Analisis dan Implementasi Pencarian Ayat Al-Quran Berbasis Fonetis Menggunakan Metode N-gram yang Digabungkan Dengan Pengodean Fonetis

Muhammad Fakhri Ar-Razi[1], Ir. Moch. Arif Bijaksana, MTech[2], Shaufiah S.T, M.T[3]

[1]Prodi S1 Teknik Informatika, Fakultas Informatika, Universitas Telkom
[2]Fakultas Informatika, Universitas Telkom
[3]Fakultas Ilmu Terapan, Universitas Telkom
[1]fakhriarrazi@gmail.com, [2]shaufiah@gmail.com, [3]arifbijaksana@gmail.com

Abstrak

Mencari ayat di Al-Qur'an tidak mudah bagi pengguna yang tidak memiliki cukup pengetahuan dan kemampuan dalam bahasa Arab. Oleh karena itu, pencarian fonetis dapat digunakan untuk mempermudah pengguna untuk mencari ayat dalam Al-Qur'an sesuai dengan pengucapan dan penulisan pengguna. Tugas akhir ini bertujuan untuk membangun system pencarian tersebut, khusus untuk penutur Bahasa Indonesia. Sebuah metode n-gram yang digabungkan dengan pengodean fonetis mengenai aturan bacaan Quran diusulkan untuk mencocokkan antara teks Al-Qur'an transliterasi yang sudah diubah ke dalam aksara latin (sesuai penuturan Bahasa Indonesia) dan *query* pengguna dalam aksara latin. Dilakukan pengindeksan dari trigram yang digunakan untuk perkiraan pencocokan string. Sistem ini menggunakan 2 skema pencarian yaitu pencarian dengan huruf vokal dan tanpa vokal yang sudah dibandingkan keduanya dan pencarian dengan vokal yang lebih baik; 2 metode pemeringkatan yaitu jumlah trigram dan letak posisi trigram. Dari hasil yang sudah diuji didapatkan presisi yang cukup baik dengan skema pencarian menggunakan vokal sebesar 0.746, sedangkan dengan skema pencarian tanpa vokal sebesar 0.515. Setelah menggabungkan 2 metode pemeringkatan dan menggunakan skema pencarian dengan vokal didapatkan nilai *recall* sebesar 0.79, serta didapatkan nilai korelasi yang cukup besar yaitu 0.907 dan sistem juga dapat menerima berbagai macam variasi *query* dengan baik.

1. Pendahuluan

Al-Quran adalah kitab suci yang menjadi rujukan utama bagi umat Islam di seluruh dunia. Secara statistik, Al-Quran terdiri atas 114 surat, 6236 ayat, dan 77 845 kata. Dengan jumlah surat, ayat, dan kata yang cukup banyak, pencarian kata pada teks Al-Quran secara manual sulit dilakukan. Oleh karena itu komputer dapat digunakan untuk membantu melakukan pencarian di dalam Al-Quran. Penelitian tentang pengembangan sistem untuk membantu pencarian ayat Al-Quran telah dilakukan sejak lama. Saat ini juga telah banyak dikembangkan aplikasi perangkat lunak untuk mempelajari Al-Quran.

Pada aplikasi-aplikasi perangkat lunak Al-Quran yang telah ada, tersedia fasilitas pencarian ayat yang mengharuskan pengguna untuk memasukkan kata kunci pencarian dalam bahasa dan aksara Arab. Hal ini cukup menyulitkan pengguna yang tidak bisa berbahasa Arab maupun menulis dengan aksara Arab. Selain itu, untuk mengetik aksara Arab di komputer dibutuhkan keyboard khusus atau perangkat lunak tambahan dan juga pada umumnya perangkat lunak pencarian ayat yang ada menggunakan teknik exact string matching, yaitu teknik pencarian ayat yang sesuai dengan kata inputan secara tepat. Teknik tersebut sangat sesuai jika pemakai perangkat lunak mengetikkan kata atau frase yang akan dicari dengan benar. Tetapi jika pemakai salah dalam mengetikkan kata inputan, perangkat lunak tidak memberikan solusi atau kemungkinan-kemungkinan dari ayat yang dimaksud. Sebagai contoh apabila potongan ayat yang ingin dicari adalah '*Bismilahirahmanirahim*' maka teknik exact string matching tidak dapat menemukan, karena tidak ada potongan ayat dalam Al-Quran transliterasi latin atau yang dilatinkan yang secara tepat mengandung potongan teks ayat tersebut, yang ada adalah '*Bismillahirrahmanirrahim*'. Kesalahan pengetikan tersebut disebabkan karena pengucapan/lafadz yang sama tetapi penulisannya

berbeda. Oleh karena itu, untuk mengatasi kesulitan pengguna dalam melakukan pencarian dengan aksara Arab, perlu dikembangkan sistem pencarian berbasis kemiripan fonetis untuk teks Al-Quran transliterasi latin. Pencocokan fonetis biasanya digunakan dalam pencarian nama orang yang ejaannya bisa berbeda meskipun pelafalannya serupa.

Dengan pencocokan fonetis, pencarian pada teks Al-Quran dapat dilakukan dengan kata kunci pencarian berupa pelafalan kata dalam aksara Latin. Selain itu, pencarian bersifat toleran terhadap perbedaan cara pelafalan atau penulisan lafal yang mungkin terjadi. Untuk metode pencocokan string (dalam hal ini ayat Al-Quran) secara tidak tepat sama, dapat digunakan metode n-gram yang digabungkan dengan pengodean fonetis. Beberapa pendekatan statistik untuk temu kembali informasi bahasa Arab telah diuji, dan trigram adalah jumlah gram terbaik untuk mengindeks teks tersebut. Keuntungan menggunakan trigram atau n-gram secamum ialah dapat dilakukan pengindeksan dengan tokenisasi n-gram pada korpus untuk membangun struktur data inverted index sehingga pencarian term tertentu dapat dilakukan dengan cepat.

Oleh karena itu, pada penelitian ini dibangun sistem pencarian ayat Al-Quran berbasis kemiripan fonetis (phonetic string) yang lebih sesuai dengan representasi pelafalan orang Indonesia. Untuk tujuan itu, dikembangkan metode pengodean fonetis yang didasarkan pada pemadanan aksara Arab-Latin yang digunakan di Indonesia serta kemiripan cara pelafalan huruf-huruf dalam Al-Quran. Metode pencarian yang digunakan adalah pencarian dengan trigram yang diterapkan pada kode fonetis dengan ukuran kesamaan yang ditentukan.

2. Landasan Teori dan Perancangan

2.1. Pedoman Alih Aksara Arab ke Latin

Banyak digunakan kata yang berasal dari bahasa Arab dengan aneka ragam lafal dan tulisan walaupun berasal dari kata yang sama. Pedoman ini disusun untuk menunjukkan perbedaan itu agar perbedaan tersebut dapat dipahami. Walaupun banyak variasi dalam penulisan kata dari bahasa Arab, hendaknya kata yang populer diutamakan penggunaannya.

2.2. Pencocokan String

Pengertian string menurut Dictionary of Algorithms and Data Structures, National Instituteof Standards and Technology (NIST) adalah susunan dari karakter-karakter (angka, alfabet atau karakteryang lain) dan biasanya direpresentasikan sebagaistruktur data array. String dapat berupa kata, frase,atau kalimat. Pencocokan string merupakan bagian penting dari sebuah proses pencarian string (string searching) dalam sebuah dokumen. Hasil dari pencarian sebuah string dalam dokumen tergantung dari teknik atau cara pencocokan string yang digunakan. Pencocokan string (string matching) menurut Dictionary of Algorithms and Data Structures, National Institute of Standards and Technology (NIST), diartikan sebagai sebuah permasalahan untuk menemukan pola susunan karakter string di dalam string lain atau bagian dari isi teks.

2.3. Pengodean Fonetis

Algoritme pengodean fonetis yang telah disebutkan tidak dapat digunakan secara langsung karena ditujukan untuk nama orang dalam aksara Arab, bukan teks berbahasa Arab secara umum [4]. Oleh karena itu, dibuat suatu algoritme serupa yang dapat memetakan aksara Arab dan penulisan lafal dalam aksara Latin ke suatu kode yang sama bila pelafalannya serupa. Algoritme pengodean fonetis yang dibuat juga mempertimbangkan cara membaca AlQuran (tajwid) yang sedikit berbeda dengan cara membaca teks berbahasa Arab biasa. Selain itu, untuk mengantisipasi kesalahan cara pelafalan Al-Quran yang sering terjadi. Contoh kesalahan tersebut ialah pelafalan huruf ش (syin) dengan lafal س (sin). Pengodean fonetis diterapkan baik pada teks Al-Quran transliterasi latin maupun pada *query* yang dimasukkan oleh pengguna. Hasil dari proses pengodean adalah string kode fonetis.

2.4. N-Gram

N-gram (Markov Chain) adalah rangkaian karakter (alfabet) atau kata yang diekstrak dari dari sebuah teks. Metode N-gram merupakan suatu metode yang sering digunakan untuk mengenali kesalahan-kesalahan yang sering terjadi pada suatu dokumen. N-gram dapat dibedakan menjadi dua kategori, yaitu basis karakter dan basis kata. Sebuah karakter N-gram merupakan rangkaian dari n karakter yang berurutan. Tujuan utama dibalik pendekatan ini adalah menentukan kata-kata yang mirip dengan rangkaian N-gram secara umum. Pada umumnya

N-gram mengekstrak dokumen atau kata-kata menjadi dua atau tiga rangkaian yang terurut (sering disebut bigrams dan trigrams). Sebagai contoh susunan trigram dari kata 'computer' adalah 'COM', 'OMP', 'MPU', 'PUT', 'UTE', 'TER' [5]. Contoh skema penerapan N-Gram dapat dilihat pada Table 2.1.

Tabel 2.1 Skema pada N-Gram

Data	Software
Etimologi	Software
N-gram pada etimologi From 1 to N-1 (N=5)	2-gram- {so, of, ft, tw, wa, ar, re} 3-gram- {sof, oft, ftw, twa, war, are} 4-gram- {soft, oftw, ftwa, twar, ware} 5-gram- {softw, oftwa, ftwar, tware}

2.4.1. Tokenisasi Trigram

Pada string kode fonetis yang dihasilkan, baik dari *query* maupun teks Al-Quran, dilakukan tokenisasi untuk mengambil trigram. Trigram yang diambil tidak memerlukan penanda awal atau akhir string karena *query* dapat berupa substring dari teks Al-Quran. Proses tokenisasi menggunakan overlapping window sepanjang 3 karakter [4]. Sebagai contoh, trigram dari string "ARABIC" adalah "ARA", "RAB", "ABI", dan "BIC".

2.4.2. Pengindeksan Trigram

Pada kode fonetis teks Al-Quran yang telah ditokenisasi, dilakukan pembentukan inverted index. Inverted index menggunakan trigram sebagai term dan ayat Al-Quran sebagai dokumen. Satu dokumen pada indeks adalah satu ayat pada Al-Quran. Informasi yang disimpan pada indeks adalah surah, identifier dokumen, jumlah trigram pada dokumen, serta posisi kemunculan pertama trigram pada dokumen. Contoh indeks dengan trigram dapat dilihat pada Gambar 2. Misalkan terdapat sekumpulan dokumen dengan identifier 7, 12, 44, dan 97 yang mengandung string "string" dan "data". Dari "string" dapat diambil 4 trigram, yaitu "str", "tri", "rin", dan "ing", sedangkan dari "data" dapat diambil 2 trigram, yaitu "dat" dan "ata".

2.4.3. Pencocokan Trigram

Trigram dari *query* dibandingkan dengan trigram yang ada pada indeks. Pada tahap ini, dihitung jumlah trigram dari dokumen yang sama dengan trigram dari *query*. Perhitungan dilakukan dengan memanfaatkan informasi yang tersimpan dalam indeks.

2.4.4. Pemeringkatan Dokumen

Pemeringkatan dokumen dilakukan dengan 2 metode, yaitu [7]:

1. Pemeringkatan yang hanya menghitung jumlah trigram yang sama antara dokumen dan *query*. Semakin banyak jumlah trigram yang sama, semakin tinggi peringkatnya (pemeringkatan jumlah). Misal terdapat *query* yang berisi 6 trigram (BIS, ISM, SMI, MIL, ILA, LAH) dari ke 6 trigram tersebut terdapat 4 trigram yang ada pada dokumen 2, maka didapatkan skor m.
2. Pemeringkatan yang menghitung jumlah trigram yang sama antara dokumen dengan *query* serta memperhitungkan posisi kemunculan term dari *query* pada dokumen. Semakin terurut dan rapat posisi kemunculan term, semakin tinggi peringkatnya (pemeringkatan posisi). Untuk mencari keterurutan term penulis menggunakan Longest Increasing Subsequence (LIS).

2.5. Longest Increasing Subsequence

Longest Increasing Subsequence adalah untuk menemukan subsequence dari urutan tertentu dimana unsur-unsur subsequence adalah rangka urutan, terendah ke tertinggi dan dimana subsequence selama mungkin. Subsequence ini tidak selalu berdekatan atau unik. Longest Increasing Subsequence dipelajari dalam konteks berbagai disiplin ilmu yang terkait dengan Matematika, Algorithmics, teori matriks acak, teori representasi dan Fisika. Masalah Longest Increasing Subsequence dipecahkan dalam waktu O (n log n), dimana n menandakan panjang urutan masukan [8].

Example:

Pada bagian pertama 16 term dari biner

0, 8, 4, 12, 2, 10, 6, 14, 1, 9, 5, 13, 3, 11, 7, 15

Longest Increasing Subsequence adalah

0, 2, 6, 9, 11, 15

Subsequence ini memiliki panjang 6; urutan masukan tidak memiliki tujuh anggota increasing subsequence. Longest Increasing Subsequence pada contoh berikut tidak unik: misalnya,

0, 4, 6, 9, 11, 15 atau 0, 4, 6, 9, 13, 15
Increasing Subsequence yang lain dengan panjang yang sama dalam urutan input yang sama.

Longest Increasing Subsequence (LIS) ini digunakan penulis untuk memberi skor keterurutan yang di terapkan pada posisi kemunculan trigram. LIS dari sebuah sekuens S adalah subsekuens monoton naik dari S dengan panjang maksimum [7]. Sekuens posisi kemunculan

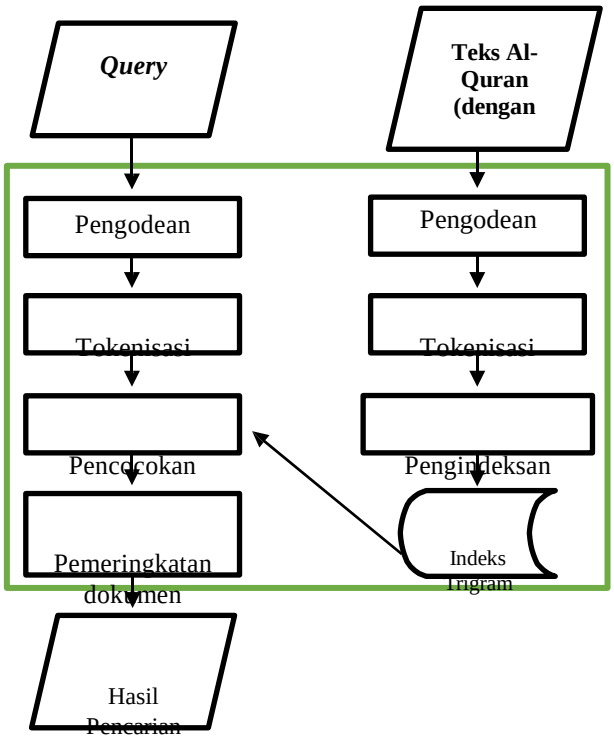
trigram yang terurut sempurna akan bernilai maksimum, yaitu panjang LIS sama dengan panjang sekuens. Algoritme yang efisien untuk mencari LIS terlampir pada

Lampiran 7. Untuk memberi skor kerapatan, dihitung rata-rata dari invers dari selisih antar elemen berdampingan pada LIS. Misalkan suatu LIS sepanjang n adalah {s1, s2, …, s^n }, maka skor kerapatannya (c) adalah:

$$c = \frac{1}{n-1} \sum_{i=1}^{n-1} \frac{1}{si + 1 - si}$$

2.6. Perancangan Sistem

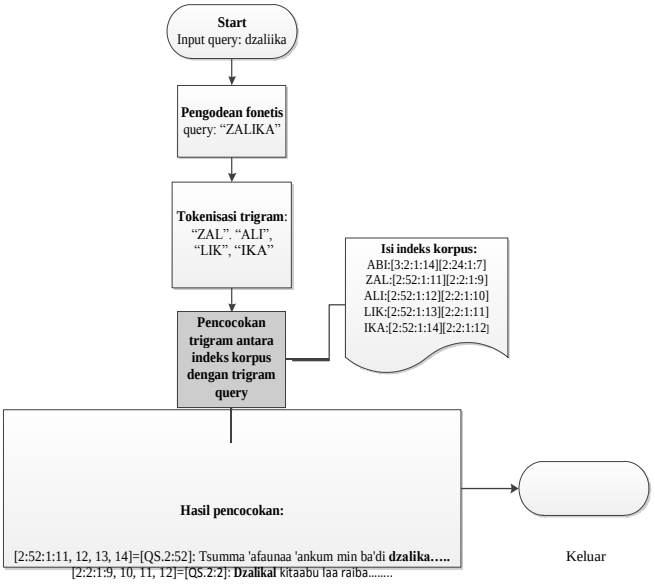
Dalam penelitian tugas akhir ini, dibangun sebuah sistem untuk memenuhi serta mencapai tujuan dari penelitian tugas akhir ini, salah satunya adalah membangun sistem pencarian ayat Al-Quran berbasis kemiripan fonetis. Sistem yang dibangun merupakan sistem yang mampu mengimplementasikan metode n-gram yang digabungkan dengan pengodean fonetis untuk pencarian ayat Al-Quran berdasarkan kemiripan fonetis. Pencocokan string (*string matching*) dari pengubahan string menjadi kode fonetis yang telah didapat melalui proses *pengodean fonetis*, sehingga dihasilkan kode fonetis bagi setiap data yang ada pada dalam database/korpus dan *query*. Selanjutnya dilakukan tokenisasi trigram baik teks Al-Quran maupun *query*, kemudian dilakukan pemrosesan pengindeksan trigram hanya pada teks Al-Quran/korpus, lalu proses yang terakhir adalah pencocokan trigram antara trigram yang dihasilkan oleh *query* dengan trigram hasil pengindeksan yang sudah dilakukan pada teks Al-Quran/korpus. Adapun gambaran umum sistem yang dapat dilihat pada Gambar 2.2.



Gambar 2.2 Gambaran umum sistem

2.6.1 Ilustrasi Keseluruhan Sistem

Dapat dilihat secara lengkap bagaimana ilustrasi proses kerja keseluruhan sistem pada Gambar 2.3.



Gambar 2.2 Ilustrasi keseluruhan sistem

2.7. Dataset / Korpus

Data teks Al-Quran transliterasi sebagai korpus diperoleh melalui hasil penulisan responden yang dibandingkan dengan Al-Quran transliterasi yang sudah ada dan aturan alih aksara arab ke latin. Contoh hasil transliterasi latin dapat dilihat pada Tabel 2.2 yang berisi ayat Surah Al-Baqarah 1-3. Dataset yang digunakan pada tugas akhir ini adalah surah Al-Fatihah, Al-Baqarah, Ali-Imran, dan seluruh surah pada Juz 30. Hasil dari Al-Quran transliterasi latin sebagai korpus sudah di tunjukkan ke 10 responden dan seluruh responden tersebut setuju dengan Al-Quran transliterasi latin yang dibuat. Terdapat sifat pada dataset yaitu seperti penambahan tanda baca (‘) sebelum huruf “A” yang menandakan dibaca seperti huruf “ع” pada huruf aksara arab.

Tabel 2.2 Contoh ayat transliterasi latin pada korpus

الٓمٓ
Alif laam miim
ذٰلِكَ الْكِتَابُ لَا رَيْبَ ۛ فِيهِ ۛ هُدًى لِّلْمُتَّقِينَ
Dzalikal kitaabu laa raiba fiihi hudal(n)-lilmuttaqiin(a)
الَّذِينَ يُؤْمِنُونَ بِالْغَيْبِ وَيُقِيمُونَ الصَّلَاةَ وَمِمَّا رَزَقْنَاهُمْ يُنفِقُونَ
Al-ladziina yu'minuuna bilghaibi wayuqiimuunash-shalaata wamimmaa razaqnaahum yunfiquun

3. Pengujian dan Analisis

3.1 Tujuan Pengujian

Pengujian yang dilakukan dalam penelitian tugas akhir ini merupakan pengujian terhadap program aplikasi sistem yang akan menguji inputan daru user yang berupa ayat Al-Quran transliterasi latin. Diharapkan dengan adanya pengujian didapatkan nilai presisi, recall, dan korelasi dari sistem yang dibuat.

3.1 Hasil Pengujian dan Analisis Skenario Pertama

Hasil penulisan dari responden didapatkan sebanyak 240 variasi penulisan *query* yang sudah disebarkan. Sebagai contoh, untuk *query* dengan kode Q2, beberapa variasi penulisan lafal oleh responden antara lain:

- Lilmuttaqiina,
- Liilmutakina,
- Lillmuttaqinna,
- Liilmutaqiinna, dan
- Lilmuttakkinna

Sebelumnya penulis menentukan terlebih dahulu nilai threshold yaitu 0.55, apabila nilai output pada sistem dari *query-query* tersebut ≥ 0.55 maka dianggap relevan dan sebaliknya apabila nilai output dari *query-query* tersebut < 0.55 maka dianggap tidak relevan. Nilai 0.55 dipilih penulis karena pada titik tersebut masih terdapat ayat-ayat yang relevan pada *query* sedangkan dibawah 0.55 sudah banyak ayat-ayat yang tidak relevan dengan *query*. Hasil perhitungan average precision (AVP) masing-masing *query* untuk setiap unit percobaan tercantum pada Tabel 3.1.

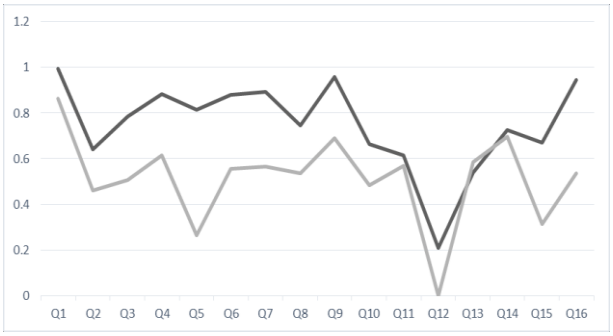
Tabel 3.1 Nilai AVP per *query*

Kode *Query*	VJ	VP	NJ	NP
Q1	0.996	0.992	0.840	0.885
Q2	0.524	0.760	0.388	0.536
Q3	0.64	0.732	0.432	0.584
Q4	0.84	0.924	0.436	0.796
Q5	0.832	0.7954	0.3596	0.168
Q6	0.76	1	0.4178	0.696
Q7	1	0.7874	0.38	0.7486
Q8	0.668	0.824	0.346	0.7248
Q9	0.916	1	0.696	0.6838
Q10	0.388	0.94	0.788	0.176
Q11	0.536	0.696	0.704	0.432
Q12	0.196	0.222	0	0
Q13	0.216	0.86	0.592	0.58
Q14	0.78	0.672	0.8608	0.532
Q15	0.512	0.8314	0.4166	0.212
Q16	1	0.892	0.3338	0.736

Dari hasil perhitungan AVP, terlihat bahwa masing-masing *query* memberikan kinerja yang berbeda untuk tiap unit percobaan. Hal ini disebabkan oleh karakteristik setiap *query* yang juga berbeda. Berdasarkan panjang, ada *query* yang pendek (misalnya Q12), ada pula yang panjang (misalnya Q3). Berdasarkan letak kemunculan, ada *query* yang biasanya terletak pada awal dokumen atau ayat (misalnya Q1), ada pula yang biasanya terletak pada akhir ayat (misalnya Q9). Berdasarkan keunikan, ada *query* yang mengandung kata yang sering muncul beberapa kali dalam satu ayat (misalnya Q14 yang mengandung kata "ALLAH"), ada pula yang mengandung kata yang biasanya hanya muncul satu kali dalam ayat (misalnya Q12).

3.1.1. Perbandingan Pencarian Vokal dan Tanpa Vokal

Nilai AVP untuk pencarian dengan vokal (V) sebesar 0.746, sedangkan untuk pencarian tanpa vokal (N) sebesar 0.515. Dengan demikian, dapat dilihat bahwa pencarian dengan memperhitungkan vokal memiliki kinerja yang lebih baik daripada pencarian tanpa memperhitungkan vokal yang hasilnya cukup signifikan. Dari hasil ini dapat diketahui bahwa pengguna dapat membedakan huruf vokal dengan baik untuk pencarian dalam Al-Quran. Oleh karena itu penulis menggunakan indeks dengan vokal untuk menggabungkan 2 metode pemeringkatan dokumen, yaitu pemeringkatan jumlah dan pemerigkatan posisi.



Gambar 3.1 Pengujian perbandingan vokal dan tanpa vokal

3.2 Hasil Pengujian dan Analisis Skenario Kedua

Pada pengujian ini didapatkan sekiranya kata-kata yang relevan sebanyak 48 dari 16 *query* yang dapat dilihat pada Tabel 4.2. Penulis menyebarkan kepada 25 responden yang mana responden dapat mengurutkan kata yang paling mirip sampai yang paling tidak mirip dari *query* tersebut. Didapatkan urutan dari 25 responden dan urutan yang dioutpukan oleh sistem sesuai 16 *query* tersebut yang terdapat pada tabel 4.2. Dapat dilihat bahwa pada Q1, Q2, Q3, Q5, Q6, Q8, Q9, Q10, Q11, Q12, Q13, Q15, dan Q16 mempunyai korelasi yang urutannya sama semua antara responden dan output dari sistem dengan nilai 1 atau sangat baik. Pada Q4 korelasi antara urutan user dan output dari sistem mempunyai nilai 0.8 yang lumayan baik karena hanya 1 urutan saja yang tertukar, begitu juga pada Q7 mempunyai nilai 0.5 yang lumayan baik juga karena seperti pada Q4 yang hanya 1 urutan yang tidak tepat antara urutan responden dan urutan output dari sistem, sedangkan untuk Q14 mempunyai nilai 0.2 lebih rendah dari pada semuanya karena mempunyai 2 urutan yang berbeda antara urutan responden dan urutan output dari sistem, tetapi nilai 0.2 dapat dibilang baik karena untuk korelasi nilai terburuk adalah -1. Rata-rata keseluruhan kolerasi pada pengujian skenario kedua ini adalah 0.907 atau lumayan baik karena mendekati 1 (sempurna).

3.3 Hasil Pengujian dan Analisis Skenario Ketiga

Pada pengujian ini penulis mendapatkan 160 macam variasi dari 16 *query* yang dituliskan langsung oleh 10 responden melalui pendengaran responden masing-masing dengan cara ke-16 *query* tersebut langsung dibacakan kepada 10 responden tersebut. Dari 160 macam variasi tersebut ada beberapa *query* yang sangat mirip bahkan sama, maka dari itu penulis hanya mengambil 112 *query* atau masing-masing 7 dalam 1 *query*. Pada hasil pengujian skenario ketiga ini dapat dilihat bahwa semua macam *query* dari Q1, Q2, Q4, Q5, Q6, Q7, Q9, Q10, Q13, Q14, dan Q16 dapat ditangani dengan baik oleh sistem aplikasi karena semua *query* dari berbagai macam responden dapat ditemukan. Pada Q4 terdapat 1 *query* yang salah atau tidak ditemukan yaitu "ANFUSAKHUM" karena jika "KH" dikodekan menjadi "H" makan yang akan ditemukan pertama kali oleh sistem aplikasi adalah "ANFUSAHUM". Pada Q11 juga terdapat 1 *query* yang tidak dapat ditemukan yaitu "AZYZZU" karena setelah dikodekan menjadi "AZYZU" yang tidak ditemukan kecocokan sama sekali dengan dengan korpus atau indeks Al-Quran yang ada sehingga sistem aplikasi memberi notifikasi 'The text is not found'. Begitu juga pada Q15 terdapat 1 *query* yang tidak bisa ditemukan yaitu *query* "BHATILYY" karena apabila "BH" dikodekan tetap menjadi "BH" sedangkan pada korpus adalah "BA" dan juga pada pengodean

"LYY" yang menjadi "LY" tetap tidak ditemukan karena pada korpus adalah "LI". Sedangkan untuk Q12 tidak ada satupun *query* yang cocok atau ditemukan karena memang pada *query* Q12 tidak tercantum dalam korpus yang ada sehingga tidak ada yang cocok dengan *query* tersebut.

3.4 Hasil Pengujian dan Analisis Skenario Keempat

Pada skenario pengujian terakhir ini penulis sudah menentukan satu kata yang akan dicari secara manual pada surah Ali-Imran dan yang akan menjadi input sistem yang nantinya akan dibandingkan jumlahnya, kata atau *query* tersebut adalah "FIIHI" (ﻪﻴﻓ). Hasil pengujian *query* tersebut dengan sistem mendapatkan output seperti pada Tabel 3.2, sedangkan hasil pencarian manual dapat dilihat pada Tabel 3.3.

Tabel 3.2 Hasil output pengujian pada sistem

Query	Ayat	Hasil Output Sistem	String Matching
Fiihi	[QS.3:9]	Rabbanaa innaka jaami'unnaasi liyaumin laa raiba **fiihi** innallaha laa yukhliful mii'aad(a)	fiihi
Fiihi	[QS.3:25]	Fakaifa idzaa jama'naahum liyaumin laa raiba **fiihi** wawuffiyat kullu nafsin maa kasabat wahum laa yuzhlamuun(a)	fiihi

Tabel 3.3 Hasil pencarian manual

Query	Ayat	Hasil Output Sistem	String Matching
Fiihi	[QS.3:198]	Lakinil-ladziina-attaqau rabbahum lahum jannaatun tajrii min tahtihaal anhaaru khaalidiina fiihaa nuzulaa min 'indillahi wamaa 'indallahi khairul(n)-lil-abraar(i)	fiihaa

Dari hasil pengujian yang sudah dilakukan, didapatkan output dari sistem sebanyak 11 ayat yang relevan dengan *query* yang terdapat pada Surah Ali-Imran ayat 9, 15, 25, 49, 55, 57, 61, 97, 88, 116, dan 164 (lampiran 5). Sedangkan pada pencarian manual yang dilakukan oleh penulis, ditemukan 14 ayat yang relevan dengan *query* yang sama (lampiran 6). Dari 14 ayat yang ditemukan secara manual terdapat 3 ayat yang tidak terdeteksi atau ditemukan oleh sistem yaitu pada ayat 117, 136 dan 198. Tidak ditemukannya 3 ayat tersebut karena posisi ayat-ayat tersebut mempunyai nilai yang kecil dan kalah prioritas karena berada pada urutan ayat belakang, karena pada sistem yang sudah dibuat ini akan mengoutpukan berdasarkan urutan ayat walaupun nilai pemeringkatan dari ayat tersebut sama dan mempunyai batas maksimal output yaitu 50. Mungkin ayat yang tidak ditemukan tersebut ada di urutan 51 keatas karena di urutan atasnya terdapat ayat-ayat dari seluruh korpus yang relevan dengan *query* "FIIHI". Pada pengujian skenario keempat ini didapatkan nilai *recall* sebesar 0.79 dari hasil pengujian menggunakan sistem dan hasil pencarian manual.

4. Kesimpulan

Berdasarkan hasil pengujian yang diperoleh dan analisis yang telah dilakukan dapat diberikan kesimpulan dari tujuan yang ada sebagai berikut:

1. Membangun sistem pencarian ayat Al-Quran berbasis kemiripan fonetis. Telah dihasilkan aplikasi sistem pencarian ayat Al-Quran berbasis kemiripan fonetis untuk Al-Quran transliterasi latin yang sesuai pelafalan orang Indonesia dengan metode n-gram dan pengodean fonetis.
2. Mengembangkan metode pengodean fonetis untuk teks Al-Quran yang sesuai untuk pembicara bahasa Indonesia. Telah dikembangkan metode pengodean fonetis untuk teks Al-Quran transliterasi latin yang sesuai pembicara orang Indonesia.
3. Mengukur kinerja sistem pencarian ayat Al-Quran berbasis kemiripan fonetis. Didapatkan kinerja dari sistem yang sudah dibuat dan yang paling baik adalah menggunakan indeks Al-Quran yang menggunakan vokal dengan nilai presisi 0.746 dan menggabungkan metode pemeringkatan jumlah dan posisi.

5. Daftar Pustaka

[1] "Wikipedia," [Online]. Available: http://id.wikipedia.org/wiki/Al-Qur'an. [Accessed 4 Juni 2015].

[2] "Wikipedia," [Online]. Available: http://id.wikipedia.org/wiki/Wikipedia:Pedoman_alih_aksara_Arab_ke_Latin. [Accessed 5 Juni 2015].

[3] M. Syaroni and R. Munir, "PENCOCOKAN STRING BERDASARKAN KEMIRIPAN UCAPAN (PHONETIC STRING MATCHING) DALAM BAHASA INGGRIS," pp. 2-5, 2005.

[4] A. F. A. Nwesri, "Effective Retrieval Techniques for Arabic Text," 2008.

[5] F. Rahmawan, "IMPLEMENTASI QUESTION ANSWERING SYSTEM PADA DOKUMEN BAHASA INDONESIA MENGGUNAKAN," p. 10, 2011.

[6] M. Davis and M. N. Karthik, "Search Using N-gram Technique Based Statistical Analysis for Knowledge Extraction in Case Based," p. 4.

[7] M. A. Istiadi, "Sistem Pencarian Ayat Al-Quran Berbasis Kemiripan Fonetis," pp. 2-4, 2012.

[8] Wikipedia, "Wikipedia," [Online]. Available: http://en.wikipedia.org/wiki/Longest_increasing_subsequence. [Accessed 5 Juni 2015].

[9] S. Muharim. [Online]. Available: https://islamagamauniversal.wordpress.com/db_cover/e_qs_002/. [Accessed 1 Juni 2015].

[10] C. D. Manning, "Introduction to Information Retrieval".

[11] B. L. L. ph.D., "Similarity as a risk factor in drug name confusion errors".

[12] [Online]. Available: http://www.pengertianahli.com/2014/07/pengertian-korelasi-apa-itu-korelasi.html#_. [Accessed 7 July 2015].

[13] [Online]. Available: http://ulumulislam.blogspot.com/2014/04/pengertian-al-quran-menurut-bahasa.html#.VZUf7vmqqko.