PENGENALAN TOKEN KALIMAT MENGGUNAKAN FINITE STATE AUTOMATA PADA NATURAL LANGUAGE PROCESSING PEMBELAJARAN ILMU NAHWU

Faiz In'amurrohman[1], Andri Pranolo[2*], dan Arief Hermawan[2]
[1]*Teknik Informatika, Universitas Teknologi Yogyakarta*
[2]*Teknik Informatika, Universitas Ahmad Dahlan Yogyakarta*
*E-Mail: andri.pranolo@tif.uad.ac.id

Abstrak

Penting bagi kaum Muslim untuk mempelajari Ilmu Nahwu yang menjadi dasar dalam memahami Al-Qur'an dan Hadits agar tidak terjadi kesalahan dalam penafsiran. Namun, pada kenyataannya mempelajari Ilmu Nahwu masih dianggap tidak mudah oleh mayoritas kaum Muslim. Sehingga, perlu alternative solusi untuk mempermudah dalam mempelajari Ilmu Nahwu. Penelitian ini bertujuan untuk memanfaatkan Finite State Automata (FSA) dalam mengenali token kalimat pada Natural Language Processing (NLP) pembelajaran ilmu Nahwu. Kalimat yang dimaksud adalah berupa kalimat tanya atau perintah. Kalimat tanya atau perintah dipecah menjadi token-token, kemudian FSA mengenali token-token tersebut diterima atau ditolak. Pada penelitian ini dihasilkan model FSA dengan 24 state untuk mengenali token kalimat pada NLP pembelajaran ilmu nahwu. FSA dapat mengenali semua token yang dibaca, baik token yang diterima maupun ditolak oleh FSA.

***Kata kunci**: finite state automata*, ilmu nahwu, *NLP*, pengenalan token, token

1. PENDAHULUAN

Pada prinsipnya setiap kaum Muslim mengetahui bahwa bahasa Arab merupakan bahasa *Al-Qur'an*. Setiap orang Muslim yang bermaksud mendalami ajaran Islam yang sebenarnya, tidak ada jalan lain kecuali harus mampu menggali dari sumber asalnya yaitu *Al-Qur'an* dan *sunnah Rasulullah sollallohu 'alaihi wasallam*. Oleh karena itu, menurut kaidah hukum Islam mengerti akan ilmu Nahwu bagi seseorang yang ingin memahami *Al-Qur'an* hukumnya *fardu 'ain*. Pelajaran Ilmu Nahwu merupakan pelajaran pertama yang dikaji dan biasanya kitab yang digunakan adalah kitab *Jurumiyyah* sebagai kitab pertama, dan *Imrithi* kitab yang dipelajari selanjutnya. Namun dibeberapa pondok pesantren lain, pelajaran Ilmu Nahwu sudah dikemas sedemikian rupa yang mengacu kepada kitab-kitab dimaksud, maupun kitab atau sumber referensi lainnya. Memahami kitab *Jurumiyyah* secara mendalam terutama dengan menghafalnya di luar kepala merupakan tugas berat bagi para santri, bahkan kadang-kadang memerlukan waktu lama, padahal selain mengkaji kitab *Jurumiyyah* para santri pun perlu mengkaji kitab-kitab lainnya (Anwar 1995).

Berdasarkan penelitian kualitatif yang dilakukan, banyaknya materi-materi yang dikaji di pondok pesantren di antaranya ilmu Tafsir, Tarikh, Lughot, Shorof, Fiqih, Tauhid, dan Akhlak menjadikan sekitar 83% santri kurang meminati disiplin ilmu Nahwu. Beberapa alasan yang mendasari kurangnya minat tersebut dinyatakan oleh para santri antara lain metode pembelajaran yang kurang menarik, banyaknya kaidah/hukum yang terdapat pada setiap bab dalam ilmu Nahwu, materi yang susah dan rumit untuk dipahami, serta penjelasan dari pengajar/*ustadz* yang kurang tepat dan membingungkan santri. Kondisi tersebut pada akhirnya menjadikan ada sekitar 54% santri yang kurang bisa memahami materi Nahwu yang disampaikan oleh pengajar, 33% santri yang cukup bisa memahami materi, dan hanya ada 13% santri saja yang dianggap sudah bisa menerapkan pada contoh permasalahan dalam disiplin ilmu Nahwu. Prosentase tersebut dapat dibuktikan dari perolehan hasil ujian pelajaran Nahwu dengan rata-rata nilai hanya 54.69 untuk santri kelas *Jurumiyyah* dan rata-rata nilai 54 untuk santri kelas *'Imrithi* ('Inayatullah 2014).

Oleh karena persoalan dimaksud, maka diperlukan solusi alternatif pembelajaran Ilmu Nahwu. Penelitian ini mengembangkan sistem pembelajaran untuk membantu santri agar dapat lebih mudah dan ringkas dalam mengkaji materi Nahwu, menggunakan metode tanya jawab. Adapun proses bertanya menggunakan bahasa alami yang dilakukan terhadap sistem yang dikembangkan, kemudian sistem akan melakukan proses dan memberikan jawaban sesuai pertanyaan dan kalimat perintah yang dimasukkan. Pada paper ini akan dipaparkan bagaimana Finite State Automata mengenali token dari kalimat tanya atau perintah sebagai input pada sistem.

FSA berfungsi *recognizer* untuk membaca *string* masukan dan memberikan kesimpulan diterima atau ditolak (Qutsiyah et al. 2015).

Penelitian lain yang telah dilakukan sebelumnya yaitu, penggunaan informasi sintaksis di dalam teks dan mengenali kesamaan kata dengan WordNet dan database leksikal (Vaishnavi et al. 2013), system CRF untuk mengenali error segmen dengan berdasarkan kepercayaan, fitur leksikal dan sintaksis ASR (Bechet & Favre 2013), menggunakan pendekatan data mining untuk token protein yang tepat dari literature (Lin et al. 2014), memanfaatkan FSA untuk mengidentifikasi perbedaan tipe jatuh (Hsieh et al. 2012), aplikasi pemenggalan kata menggunakan finite state automata dan pemotongan imbuhan menggunakan algoritma jelita asian (Erikson et al. 2013), dan aplikasi konversi aksara latin ke aksara jawa menggunakan finite state automata (Aji et al. 2012).

2. METODE

Dalam NLP, pengenalan token dilakukan pada tahap analisis leksikal yang bertujuan untuk mendapatkan daftar token (kata/leksikon) yang diperoleh melalui kalimat tanya atau perintah yang dimasukkan. Berikut ini merupakan jenis pertanyaan/perintah yang dapat diterima oleh sistem:

a. Jelaskan, contoh:
   1) Jelaskan pengertian *kalam*!
   2) Jelaskan definisi *i'rab*!
b. Definisikan, contoh:
   1) Definisikan maksud dari *isim*!
   2) Definiskan pengertian tentang *fi'il*!
c. Apa/apakah, contoh:
   1) Apa saja syarat dari *isim*?
   2) Apakah yang disebut dengan *musnad*?
d. Sebutkan, contoh:
   1) Sebutkan pembagian *i'rab*!
   2) Sebutkan tanda dari *rafa'*!
e. Berikan/berilah, contoh:
   1) Berikan contoh *jama' taksir*!
   2) Berilah dalil mengenai *i'rab*!
f. Berapa/berapakah, contoh:
   1) Ada berapa alamat *nashab*?
   2) Berapakah macam *fi'il*?
g. Di mana/di manakah, contoh:
   1) Dimana saja tempat *dhammah*?
   2) Dimanakah tanda untuk *asmaul khamsah*?

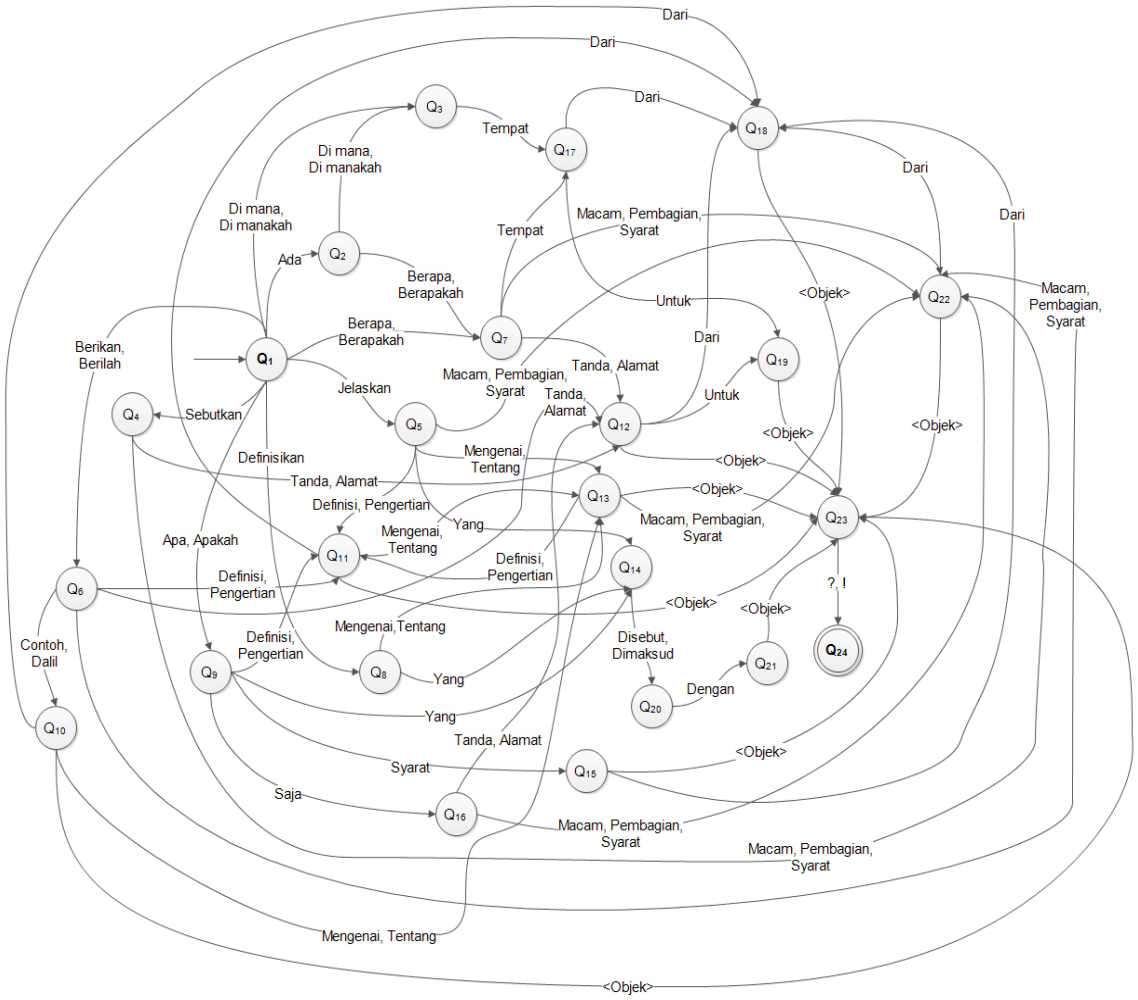
Struktur pertanyaan/perintah di atas memiliki pola keteraturan yang dapat disusun menjadi *grammar* (aturan produksi) menggunakan notasi dari aturan *Backus Naur Form* (Hariyanto 2004) sebagai berikut:

<Kalimat> ::= <FraseTanya><FraseAtribut> <Objek> <Tanda>
<FraseTanya> ::= <Pelengkap> <Jenis> <Pelengkap>
<FraseAtribut> ::= <Sambung> <Atribut> <Sambung>
<Objek> ::= materi yang diinginkan
<Tanda> ::= ? | !
<Pelengkap> ::= ada | saja | ε
<Jenis> ::= jelaskan | definisikan | apa | apakah | sebutkan | berikan | berilah | berapa | berapakah | di mana | di manakah
<Sambung> ::= dengan | yang | dari | tentang | mengenai | untuk | ε
<Atribut> ::= pengertian | definisi | syarat | macam | pembagian | contoh | tanda | alamat | dimaksud | disebut | dalil | tempat

Keterangan:

< > : Notasi untuk non-terminal (kelas sintaks)
::= : Notasi untuk menggantikan "→" (menghasilkan)
| : Notasi untuk merepresentasikan "atau"
ε : Notasi kosong

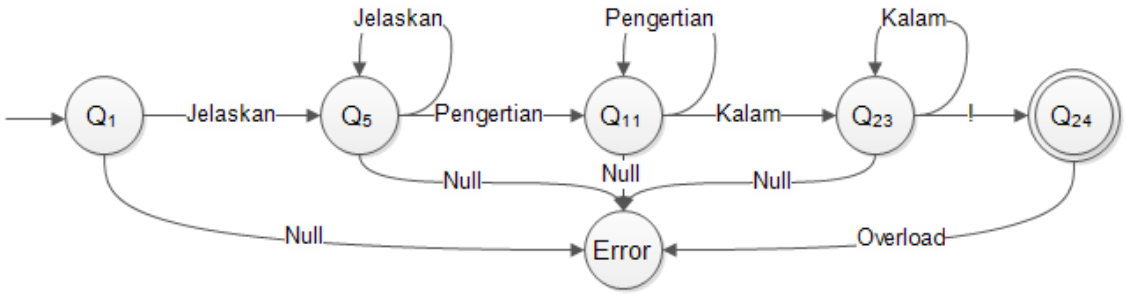
Gambar 1 menunjukkan diagram FSA dari struktur pertanyaan/perintah yang dapat dibentuk berdasarkan aturan produksi yang telah dibuat.



Gambar 1 Diagram FSA struktur pertanyaan/perintah

3. HASIL DAN PEMBAHASAN

Sebagai contoh pertanyaan yang dimasukkan adalah “Jelaskan pengertian *kalam*!”, kemudian *scanner* akan memecah kalimat tersebut menjadi daftar token-token serta mengubah *letter case* token tersebut menjadi huruf kecil untuk mempermudah dalam mengenali leksikon, sehingga didapatkan hasil tokenisasi: [“jelaskan”,“pengertian”,“kalam”, “!”]. Pada tahap ini, *scanner* akan mengambil token-token dari bahasa/*string* yang dimasukkan (*lexeme*) untuk dikenali morfologinya (bentuk kata) serta mengecek kesesuaian ejaan dari token-token yang telah didapatkan. Pengecekan ejaan ini sangat bergantung pada daftar *keyword*/atribut yang telah ditentukan, sehingga *scanner* dapat mengenalinya. Mengacu pada hasil tokenisasi di atas, terlihat tidak terdapat kesalahejaan pada token (bila terjadi kesalahan dalam ejaan, *scanner* akan menampilkan pesan kesalahan beserta saran pembetulan kata).



Gambar 2. Diagram FSA contoh pertanyaan/perintah

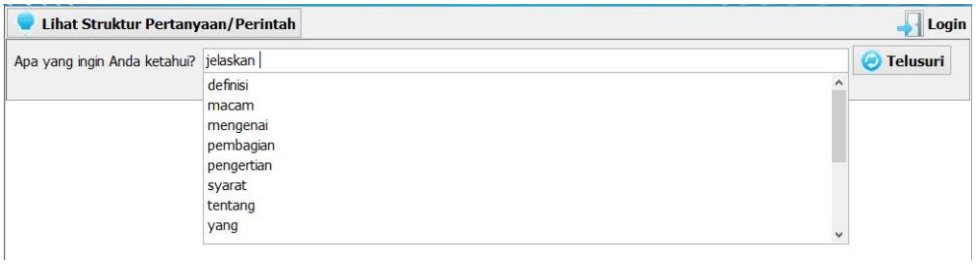
Konfigurasi diagram FSA pada Gambar 2 sebagai berikut:
$M = (Q, V_T, \delta, Q_0, F)$
$Q = \{Q_1, Q_5, Q_{11}, Q_{23}, Q_{24}, \text{Error}\}$
$V_T = \{\text{Jelaskan, Pengertian, Kalam, !, Null, Overload}\}$
δ = dapat dilihat pada Tabel 3.1
$Q_0 = Q_1$
$F = \{Q_{24}\}$

Berdasarkan Gambar 2, terlihat bahwa *lexeme* dapat mencapai *state* akhir (Q_{24}), sehingga kalimat pertanyaan/perintah dapat diterima oleh *scanner* dan akan dilanjutkan untuk proses berikutnya yaitu *parsing*. Apabila *lexeme* tidak dapat mencapai *state* akhir (dikarenakan morfologinya tidak dapat dikenali atau terdapat kesalahan ejaan), maka akan menuju ke *state error* yang berarti *scanner* menolak *lexeme* tersebut.

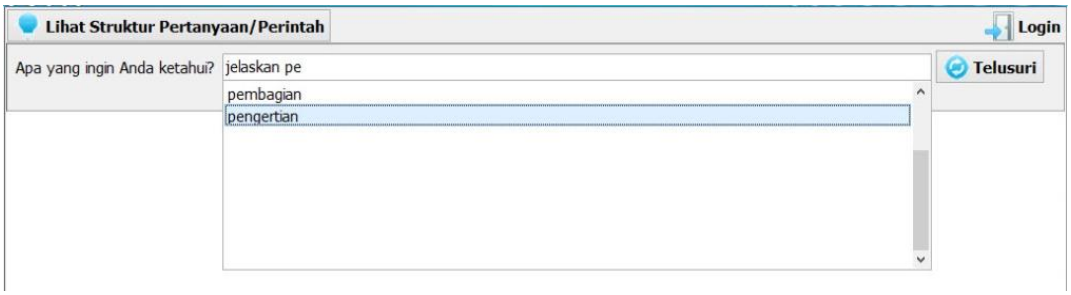
Tabel 1. Tabel transisi diagram FSA contoh pertanyaan/perintah

δ	**Masukan**					
	Jelaskan	**Pengertian**	**Kalam**	**!**	**Null**	**Overload**
Q_1	Q_5				Error	
Q_5	Q_5	Q_{11}			Error	
Q_{11}		Q_{11}	Q_{23}		Error	
Q_{23}			Q_{23}	Q_{24}	Error	
Q_{24}						Error

Tabel 1 menunjukkan tabel transisi diagram FSA pada Gambar 2. Pada saat pengguna mengetikkan pertanyaan, sistem akan memberikan beberapa saran kata (atribut) yang memiliki kesesuaian dengan kata sebelumnya. Saran kata tersebut akan muncul pada saat pengguna menekan spasi pada *keyboard* (Gambar 3) sedangkan pada saat pengguna mengetikkan kata tersebut, sistem hanya akan menampilkan daftar kata yang mengandung karakter dari kata yang diketikkan (Gambar 4).

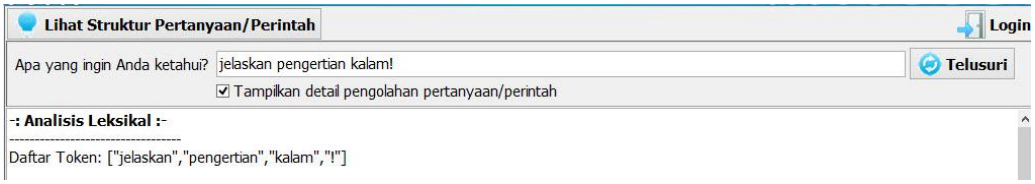


Gambar 3. Saran kata yang sesuai dengan kata sebelumnya



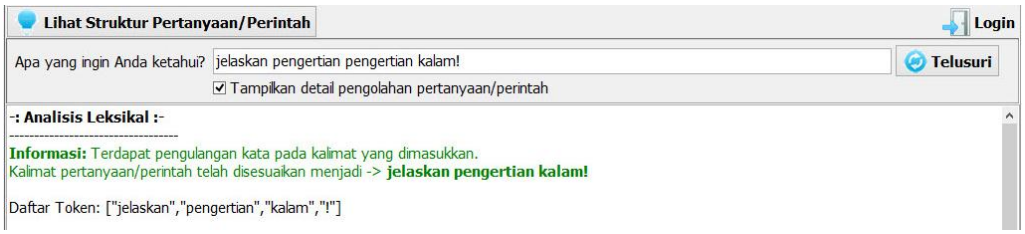
Gambar 4. Kata yang mengandung karakter yang diketikkan

Setelah kalimat pertanyaan tersebut dimasukkan, maka selanjutnya akan dilakukan proses analisis leksikal yang akan menghasilkan daftar token seperti pada Gambar 5.



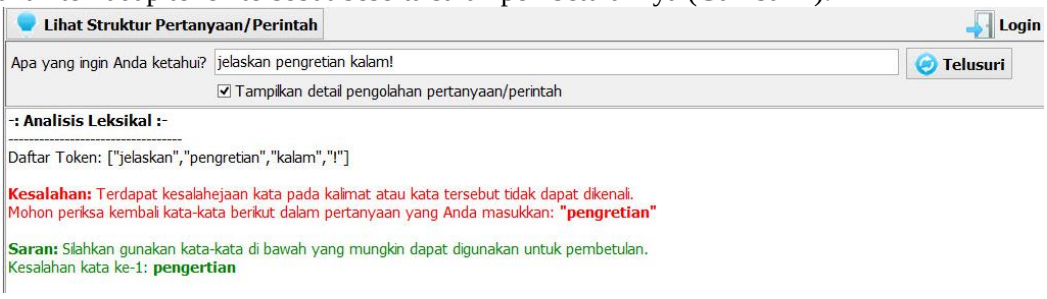
Gambar 5. Analisis leksikal (tokenisasi)

Misalkan apabila terjadi pengulangan kata pada kalimat yang dimasukkan (Gambar 6), maka sistem akan menghilangkan kata yang kedua dan kalimat tersebut akan disesuaikan menjadi bentuk kalimat yang telah dibenarkan.



Gambar 6. Pengulangan kata pada kalimat

Jika dalam proses ini terdapat kesalahejaan pada token, maka sistem akan memberikan pesan kesalahan terhadap token tersebut beserta saran pembetulannya (Gambar 7).



Gambar 7. Kesalahejaan pada token beserta saran pembetulannya

4. SIMPULAN

Hasil penelitian ini dapat disimpulkan bahwa:

1) Model diagram transisi atau *state machine* (FSA) menghasilkan 24 *state*.
2) FSA dapat mengenali token kalimat pada NLP dengan keputusan diterima atau ditolak

DAFTAR PUSTAKA

Aji, C.S., Sarwoko, E.A. & Saputra, R., 2012. APLIKASI KONVERSI AKSARA LATIN KE AKSARA JAWA MENGGUNAKAN FINITE STATE AUTOMATA DENGAN VISUAL BASIC. *Journal of Informatics and Technology*, 1(3).

Anwar, M., 1995. *Ilmu Nahwu: Terjemahan Matan Al-Ajurumiyyah dan 'Imrithi Berikut Penjelasannya*, Bandung: Sinar Baru Algensindo.

Bechet, F. & Favre, B., 2013. ASR error segment localization for spoken recovery strategy. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. pp. 6837–6841.

Erikson, T., Christanti, V. & Tony, 2013. Perancangan Aplikasi Pemenggalan Kata Menggunakan Finite State Automata dan Pemotongan Imbuhan Menggunakan Algoritma Jelita Asian. *Jurnal Ilmu Komputer dan Sistem Informasi*, 1(2).

Hariyanto, B., 2004. *Teori Bahasa, Otomata, dan Komputasi serta Terapannya*, Bandung: Penerbit Informatika.

Hsieh, S.-L. et al., 2012. A Finite State Machine-Based Fall Detection Mechanism on Smartphones. In *Ubiquitous Intelligence Computing and 9th International Conference on Autonomic Trusted Computing (UIC/ATC), 2012 9th International Conference on*. pp. 735–739.

'Inayatullah, P., 2014. *Laporan Rekapitulasi Hasil Ujian Madrasah*, Yogyakarta.

Lin, S.H., Ding, S.H. & Zeng, W.S., 2014. *Algorithmic Aspects in Information and Management* Q. Gu, P. Hell, & B. Yang, eds., Cham: Springer International Publishing. Available at: http://www.scopus.com/inward/record.url?eid=2-s2.0-84904020089&partnerID=tZOtx3y1 [Accessed June 19, 2015].

Qutsiyah, Rachman, F.. & Solihin, F., 2015. Aplikasi Teks to Speech Dalam Sistem Penerjemah Bahasa Indonesia-Madura Menggunakan Metode FSA (Finite State Automata). *Jurnal Sarjana Teknik Informatika*, 1(1), pp.1–10.

Vaishnavi, V., Saritha, M. & Milton, R.S., 2013. Paraphrase identification in short texts using grammar patterns. In *2013 3rd International Conference on Recent Trends in Information Technology, ICRTIT 2013*. Chennai; India: IEEE, pp. 472–477.