Penggabungan Algoritma Markov Chain Monte Carlo dan Metode Statistik pada Named Entity Recognition Lintas Bahasa Suku di Indonesia untuk Pembelajaran Alkitab

Vania Putri Minarso, Henry Novianus Palit, Justinus Andjarwirawan
Program Studi Informatika Fakultas Teknologi Industri Universitas Kristen Petra
Jl. Siwalankerto 121 – 131 Surabaya 60236
Telp. (031) – 2983455, Fax. (031) – 8417658
E-Mail: vaniaputri430@gmail.com, hnpalit@petra.ac.id, justin@petra.ac.id

ABSTRAK

Alkitab merupakan teks *bilingual* atau *monolingual* yang penting, sehingga menjadikannya sumber pelatihan untuk beberapa tugas seperti penerjemahan, analisis *named entity*, dan transliterasi. *Named entity* adalah suatu objek yang ada di dunia nyata, seperti orang, lokasi, baik dalam bahasa Indonesia maupun bahasa Suku yang ada di Indonesia. Selain itu organisasi, produk, dll., yang dapat dilambangkan dengan nama yang tepat. Tujuan dari semuanya ini adalah memudahkan orang yang ingin belajar Alkitab dapat mengerti setiap nama-nama penting di Alkitab, digunakan untuk memudahkan misionaris yang ingin melakukan penginjilan di daerah-daerah agar dapat mengetahui secara cepat *named entity* untuk bahasa Suku yang ada di Indonesia. Dalam proses ini, ada kemungkinan terjadi kesenjangan atau perbedaan antara *named entity* Bahasa Indonesia dengan Bahasa Suku di Indonesia.

Untuk menjawab persoalan tersebut, dirancang suatu aplikasi yang menerapkan penggabungan dari algoritma *Markov Chain Monte Carlo* yang dikemas dalam *efmaral tool*, metode statistik yang dikemas dalam *giza++ tool*, dan reparameterisasi IBM Model 2 yang dikemas dalam *fast align tool*. Dari hasil ketiga *tools* yang merupakan korelasi setiap kata dari Bahasa Indonesia dan Bahasa Suku, akan diambil hasilnya dengan mencari konsensus yang tepat sehingga dapat menemukan hasil *named entity* dalam Bahasa Suku yang tepat. Penentuan *named entity* dilakukan dari penomoran strong yang berasal dari Bahasa Asli.

Berdasarkan hasil penelitian, dapat diketahui bahwa hasil akurasi *named entity* dengan menggunakan penggabungan *efmaral*, *giza++*, dan *fast align tools* lebih baik dibandingkan hasil akurasi *named entity* menggunakan *efmaral tool*. Akurasi yang dihasilkan *efmaral tool* sebesar berkisar antara 0,07 sampai 0,66. Akurasi yang dihasilkan dari *giza++ tool* sebesar berkisar antara 0,47 sampai 0,90. Akurasi yang dihasilkan dari gabungan *tools (efmaral, giza++, fast align)* berkisar antara 0,36 sampai 0,87.

Kata Kunci: *named entity recognition, Markov Chain Monte Carlo*, metode statistik, reparameterisasi IBM Model 2, *efmaral, giza++, fast align*

ABSTRACT

Bible is an important bilingual or monolingual text, making it a resource for training such as translation, named entity analysis, and transliteration. Named entity is a realworld object such as people, location, organization, product, et cetera, that can be symbolized with an exact name. The objective of the experiment is to help people learning the bible in understanding important names in the bible either in Indonesian or other local language in Indonesia. This can also help missionaries doing evangelism to quickly understand named entity in the local language of the place they are in. In this process, they may be a gap exists between the named entity in Indonesian and the local language.

To solve this issue, designed an application that integrate Markov Chain Monte Carlo algorithm that wrapped inside efmaral tool with statistical method wrapped inside giza++ tool, and IBM Model 2 reparameterization wrapped inside fast align tool. From the product of all the tools, which is correlation between every word in Indonesian with the word in the local language, the result will be decided by finding the right consensus to find the correct named entity in the local language. The named entity will be chosen based on the strong numbering from the original language.

Based on the result of the experiment, integration of efmaral, giza++, and fast align tools yields better accuracy than efmaral tool alone. Efmaral tool has the accuracy around 0,07 to 0,66. Giza++ tool has the accuracy around 0,47 to 0,90. The integrated tools (efmaral, giza++, and fast align) has the accuracy around 0,36 to 0,87.

Keywords: *named entity recognition, Markov Chain Monte Carlo, statistical method, IBM Model 2 reparameterization, efmaral, giza++, fast align*

1. PENDAHULUAN

Alkitab merupakan teks *bilingual* atau *monolingual* yang penting, sehingga menjadikannya sumber pelatihan untuk beberapa tugas seperti penerjemahan, analisis *named entity*, dan transliterasi [10]. *Named entity* adalah suatu objek yang ada di dunia nyata, seperti orang, lokasi, organisasi, produk, dll., yang dapat dilambangkan dengan nama yang tepat. Contoh dari *named entity* adalah Adam, Hawa, dan Taman Eden. Oleh karena itu, Alkitab dapat dijadikan salah satu sumber untuk mengekstrak *named entity* yang ada dalam teks dan sekaligus menerjemahkan *named entity* tersebut untuk lintas bahasa. Penerjemahan *named entity* untuk lintas bahasa dilakukan agar memudahkan mencari *named entity* sesuai bahasa yang diinginkan. Tujuan dari semuanya ini adalah memudahkan orang yang ingin belajar Alkitab dapat mengerti setiap nama-nama penting di Alkitab baik dalam bahasa Indonesia maupun bahasa Suku yang ada di Indonesia dan memudahkan misionaris yang ingin melakukan penginjilan di daerah-daerah agar dapat mengetahui secara cepat *named entity* untuk bahasa Suku yang ada di Indonesia.

Sistem yang dibuat akan menerapkan penggabungan dari algoritma *Markov Chain Monte Carlo* yang dikemas dalam *tool efmaral*, metode statistik yang dikemas dalam *tool giza++*, dan reparameterisasi IBM Model 2 yang dikemas dalam *tool fast align*. Dari hasil ketiga *tools* yang merupakan korelasi setiap kata dari Bahasa Indonesia dan Bahasa Suku, akan diambil hasilnya dengan mencari konsensus yang tepat sehingga dapat menemukan hasil

named entity dalam Bahasa Suku yang tepat. Penentuan *named entity* dilakukan dari penomoran strong yang berasal dari Bahasa Asli.

Setelah melalui beberapa proses penggabungan, sistem dapat menampilkan hasil *named entity* dengan tampilan yang *user friendly*. Melalui sistem ini, pengguna dapat mengetahui *named entity* Bahasa Suku yang terkorelasi dari *named entity* dalam Bahasa Indonesia.

2. DASAR TEORI

2.1. *Named Entity Recognition*

Named Entity Recognition (NER) merupakan turunan dari ekstraksi informasi yang bertujuan untuk memudahkan mencari informasi dengan cara pemberian nama entitas pada setiap kata dalam suatu teks. Pada umumnya, NER digunakan untuk mendeteksi nama orang, nama tempat, dan organisasi dari sebuah dokumen.

2.2. *Stemming*

Stemming adalah suatu proses untuk menemukan kata dasar dari sebuah kata. Hal yang dilakukan dalam *stemming* adalah menghilangkan semua imbuhan baik yang terdiri dari awalan, sisipan, akhiran, dan kombinasi dari awalan dan akhiran.

2.3. Algoritma *Markov Chain Monte Carlo*

Markov Chain Monte Carlo (MCMC) adalah metode pengambilan sampel dari distribusi probabilitas menggunakan rantai markov. Rantai markov adalah suatu teknik matematika yang digunakan untuk melakukan pemodelan bermacam-macam sistem dan proses bisnis. Teknik ini dapat digunakan untuk memperkirakan perubahan untuk waktu yang akan datang dalam variabel-variabel yang dinamis atas dasar perubahan dari variabel-variabel yang dinamis di waktu lalu. Metode MCMC digunakan dalam pemodelan data untuk inferensi bayesian. MCMC dapat memberikan jawaban untuk masalah simulasi yang sulit dari distribusi dengan dimensi yang tinggi, biasanya muncul dengan model yang kompleks. Teknik *monte carlo* adalah metode pengambilan sampel [4].

2.4. *Efmaral Tool*

Efmaral (Efficient Markov chain Aligner) adalah sistem *word alignment* yang menggunakan model *Bayesian* dengan inferensi *Markov Chain Monte Carlo* [7]. *Efmaral tool* dilakukan untuk melakukan *word alignment* menggunakan *gibbs sampling* dengan *Bayesian extension* dari IBM model. *Gibbs sampling* merupakan prosedur dimana nilai awal dari parameter akan diestimasi untuk menghasilkan nilai parameter selanjutnya.

2.5. *Moses*

Moses adalah implementasi dari pendekatan statistik untuk *machine translation* (MT). *Moses* adalah sistem penerjemah berbasis statistik yang dapat memudahkan untuk melatih model penerjemahan secara otomatis pasangan bahasa apapun. Dalam *statistical machine translation* (SMT), sistem terjemahan dilatih dengan sejumlah data paralel. Data paralel adalah kumpulan kalimat dalam dua bahasa yang berbeda, yang disejajarkan dengan kalimat, dimana setiap kalimat dalam satu bahasa dicocokan dengan kalimat yang sudah diterjemahkan dalam bahasa lain. *Moses* melakukan proses pelatihan dengan mengambil data paralel dan menggunakan *coocurrences* kata dan segmen (atau lebih dikenal sebagai frase) untuk menyimpulkan *correspondences* dari terjemahan antara dua Bahasa [5].

2.6. *Giza++ Tool*

Giza++ adalah toolkit untuk melatih model penyelarasan kata. *Giza++* mendukung IBM model 1 sampai 5. Proses awal mengunakan *script preprocessing* untuk mengekstrak kalimat-kalimat bilingual. Setelah mengekstrak kalimat bilingual dalam bahasa sumber dan bahasa target, membuat direktori yang digunakan untuk menjalankan *giza++*. Selanjutnya, menggunakan *script bundled* dengan sistem terjemahan mesin *Moses* untuk menandai teks dalam setiap bahasa. Hasilnya berupa file vcb (kosakata) dan file snt (kalimat), yang masing-masing berisi daftar kosakata dan kalimat yang selaras.
IBM model 4 dan 5 menggunakan *word classes* untuk memodelkan distorsi. Distorsi adalah sebuah konsep untuk memodelkan bagaimana perubahan urutan kata dalam berbacgai bahasa, sebagai contoh *'white bird'* (Bahasa Inggris) dan *'pájaro blanco'* (Bahasa Spanyol) [2].

2.7. *Fast align Tool*

Fast align adalah *aligner* yang sederhana, cepat, dan *unsupervised*. *Fast align* merupakan reparametersisasi IBM Model 2. Model yang digunakan adalah variasi dari *lexical translation model* [1].

2.6. *Levenshtein*

Jarak *levenshtein* antara 2 kata adalah jumlah minimum penyuntingan satu karakter (penyisipan, penghapusan, atau penggantian) yang diperlukan untuk mengubah satu karakter menjadi karakter yang lain. Perhitungan *levenshtein distance* ini menggunakan matriks untuk menghitung jumlah perbedaan string antara dua string. Algoritma ini berjalan mulai dari pojok kiri atas sebuah array dua dimensi yang berisi string awal dan string target beserta nilai *cost*. Nilai *cost* menjadi nilai *levenshtein distance* yang menggambarkan jumlah perbedaan dua string [11].

2.8. *Soundex*

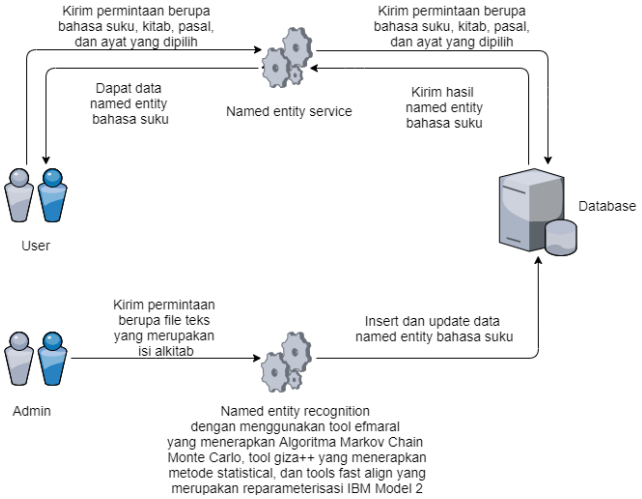
Soundex adalah fungsi yang digunakan untuk menghitung s*oundex key* dalam sebuah string. *Soundex key* memiliki properti jika kata-kata yang diucapkan sama akan menghasikan s*oundex key* yang sama (bukan ejaan). *Soundex* membagi huruf-huruf ke dalam tujuh kelas yang berbeda. Kode yang "Tidak dikodekan" untuk huruf A, I, U, E, O, H, W, Y. Kode "1" untuk huruf B, F, P, V. Kode "2" untuk huruf C, G, J, K, Q, S, X, Z. Kode "3" untuk huruf D, T. Kode "4" untuk huruf L. Kode "5" untuk huruf M, N. Kode "6" untuk huruf R [8].

2.9. *Similar text*

Similar text merupakan fungsi dalam PHP yang digunakan untuk menghitung kesamaan antara 2 string. Hasil dari s*imilar text* merupakan presentase dari kesamaan 2 string tersebut [6]. Presentase s*imilar text* didapatkan dari membagi similar_text() dengan rata-rata panjang string dikalikan 100 Misal s*imilar text* dari string "bafoobar" dan "barfoo". Kesamaan dari 2 string tersebut adalah karakter b, a, f, o, o yang panjangnya 5. Untuk rata-rata panjang stringnya adalah 7 yang didapat dari 14 ÷ 2 = 7. Selanjutnya untuk menghitung *similar text*-nya dengan cara jumlah string yang sama dibagi rata-rata panjang string. Jadi, (5÷7) × 100% =71,428%.

3. DESAIN SISTEM

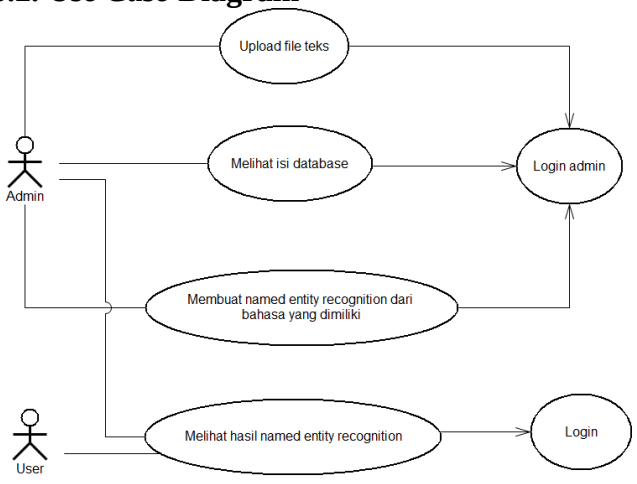
3.1 Arsitektur Sistem



Gambar 1. Arsitektur sistem dari *named entity recognition* dan *named entity service*

Dari Gambar 1 dijelaskan mengenai rancangan sistem arsitektur yang akan diimplementasikan pada aplikasi *named entity recognition* dan *named entity services*. Sistem arsitektur ini memiliki dua akses untuk admin dan *user* biasa. *User* biasa dapat mengirimkan permintaan untuk menampilkan hasil *named entity* untuk Bahasa Suku. Hasil *named entity* diambil dari *database*. Sedangkan admin dapat mengakses *named entity recognition* untuk memproses data Alkitab sehingga mendapatkan *named entity* yang sesuai dengan data tersebut. Selanjutnya, data *named entity* Bahasa Suku akan disimpan dalam database. Data Alkitab yang diproses yaitu data Alkitab dalam versi Bahasa Indonesia dan Alkitab versi Bahasa Suku.

3.2. *Use Case* Diagram

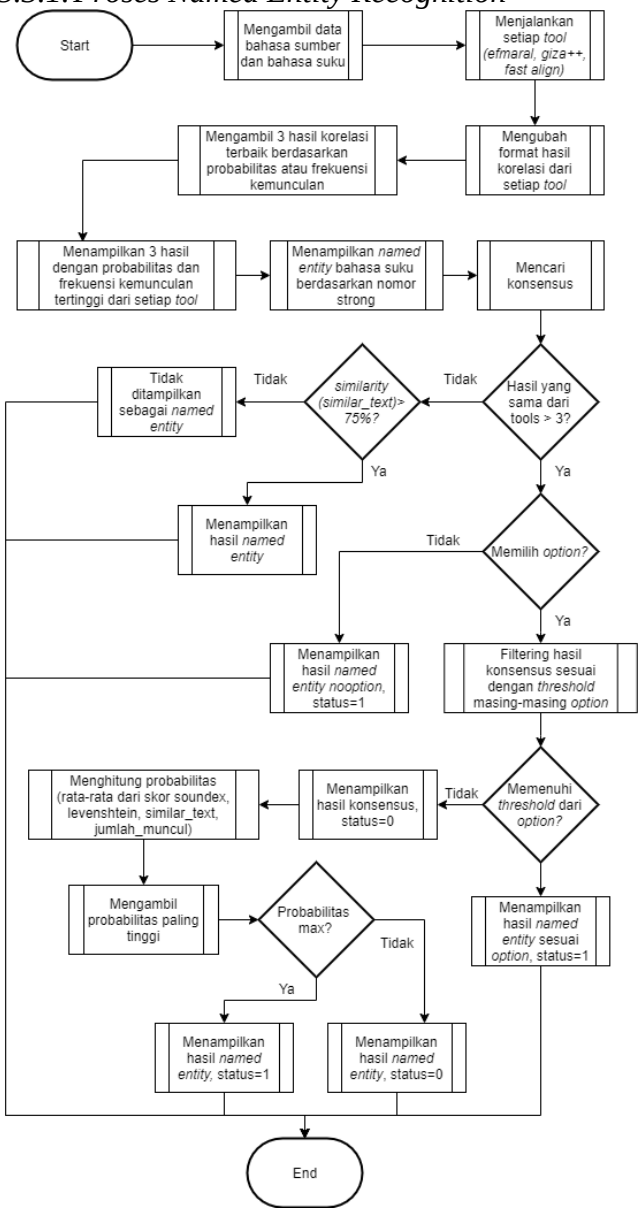


Gambar 2. *Use case diagram* sistem *builder* dan aplikasi *named entity recognition*

Usecase diagram pada Gambar 2 mendeskripsikan aktor-aktor yang ada di dalam sistem yaitu *User* Admin dan *User* biasa. Setiap aktor memiliki fungsi yang berbeda-beda. Yang pertama, *user* admin bisa melakukan *upload* file ke sistem, melihat isi database yang dihasilkan, membuat *named entity recognition* dari bahasa yang dimiliki. Yang kedua, *use case* untuk *user* biasa yang hanya bisa melihat hasil *named entity* sesuai inputan bahasa, kitab, pasal, dan ayatnya, Setiap aktor memiliki fungsi sendiri-sendiri, dan tidak saling berinteraksi dengan aktor yang lain.

3.3. *Flowchart*

3.3.1. Proses Named Entity Recognition



Gambar 3. *Flowchart* proses *named entity recognition*

Flowchart diagram proses *named entity recognition* dapat dilihat pada Gambar 3. Proses dimulai dengan mengambil data untuk teks Alkitab versi Bahasa Indonesia dan Bahasa Suku dari *database*. Data dari *database* disesuaikan dengan format setiap *tools* yang akan digunakan untuk mengkorelasikan antara bahasa sumber dan bahasa targetnya. *Tools* yang digunakan yaitu *efmaral*, *giza++*, *dan fast align*. Ketiga *tools* memiliki format input yang berbeda-beda. Format input untuk *efmaral* adalah <s num ={id ayat}>{isi ayat}</s>. Format input untuk *giza++* adalah {isi ayat}. Format input untuk *fast align* adalah {isi ayat sumber} ||| {isi ayat target}. Setelah semua data sesuai dengan formatnya masing-masing, data akan diproses oleh masing-masing *tools*. Hasil korelasi dari setiap *tools* berbeda-beda. Hasil korelasi dari *efmaral* dan *fast align*

adalah korelasi berupa index kata sumber dan index kata target. Hasil korelasi dari *giza++* adalah korelasi kata per kata dari bahasa sumber dan bahasa target. Selanjutnya, hasil dari *efmaral* dan *fast align* yang berupa index kata diubah menjadi kata untuk index tersebut. Setelah diubah menjadi kata, hasil korelasi setiap korelasi dihitung frekuensi kemunculannya dan dilakukan *sorting* secara *decsending*. Lalu, diambil 3 hasil dengan frekuensi kemunculan tertinggi. Sedangkan, hasil dari *giza++* untuk korelasi arah *reverse* (Bahasa Suku – Bahasa Indonesia) disesuaikan formatnya menjadi Bahasa Indonesia – Bahasa Suku. Selanjutnya, probabilitas dari hasil giza++ di *sorting* secara *decsending* dan diambil dengan 3 hasil dengan probabilitas tertinggi. Lalu, diambil 3 hasil dengan frekuensi kemunculan tertinggi. Hasil dari ketiga tools ditampilkan dalam 1 file teks. Selanjutnya, dari hasil tersebut dicari yang merupakan nama orang dan nama tempat berdasarkan nomor strong. Hasil dari proses tersebut, merupakan list yang merupakan *named entity.*

Selanjutnya, mencari konsensus dengan 2 cara. Yang pertama, jika menemukan hasil yang sama kurang dari 3 untuk nama orang ataupun nama tempat, serta memiliki *similar_text* lebih dari 75% maka diambil sebagai *named entity* serta memiliki status adalah 1. Yang kedua, jika menemukan hasil yang sama lebih dari 3 untuk setiap nama orang ataupun nama tempat, akan dicek kembali jika *admin* memilih *option*. *Option* terdiri dari 3 yaitu *soundex*, *levenshtein*, dan *similar_text*. Pengambilan konsensus untuk setiap *option* memiliki *threshold* yang berbeda-beda. Untuk *soundex*, jika hasil dari kata sumber dan hasil dari kata target memiliki skor *soundex* yang sama maka kata tersebut akan diambil sebagai *named entity* serta memiliki status adalah 1. Untuk *levenshtein,* jika hasil dari kata sumber dan hasil dari kata target memiliki skor *levenshtein* lebih dari 55% maka kata tersebut akan diambil sebagai *named entity* serta memiliki status adalah 1. Untuk *similar_text*, jika hasil dari kata sumber dan hasil dari kata target memiliki skor *similar_text* lebih dari 55% maka kata tersebut akan diambil sebagai *named entity* serta memiliki status adalah 1. Selain itu, kata akan tetap disimpan dengan statusnya 0. Status 1 menunjukkan hasil *named entity* dan status 0 menunjukkan bukan hasil *named entity* dari proses konsensus.

Hasil dari konsensus akan dibandingkan dengan hasil *masterlist* yang sudah ada. Hasil *masterlist* berupa nomor strong dan nama. Jika tidak ditemukan *named entity* yang sesuai dengan data dalam *masterlist*, data dari konsensus akan di proses lagi dengan mengambil probabilitas yang paling tinggi untuk dijadikan hasil *named entity*. Probabilitas didapatkan dari hasil menghitung rata-rata skor *soundex*, *levenshtein*, *similar_text*, dan jumlah kemunculan.

3.4. Formula untuk Mendapatkan Probabilitas

3.4.1. Menghitung skor soundex

$$Soundex_1 = soundex(str1)$$
$$Soundex_2 = soundex(str2)$$
$$Skor\ soundex = \frac{jum\ karakter\ yang\ sama}{4} \times 100 \qquad (1)$$

Contoh:
str1= “firaun” soundex$_1$= F**65**0
str2= “pringon” soundex$_2$= P**65**2
Contoh penggunaan rumus skor soundex:

$$Skor\ soundex = \frac{2}{4} \times 100 = 50$$

3.4.2. Menghitung skor levenshtein

$$lev = levenshtein(str1, str2)$$
$$Skor\ levenshtein = \left(1 - \frac{lev}{\max length}\right) \times 100 \qquad (2)$$

Contoh:
str1= “firaun” lev=5
str2= “pringon” maxlength=7
Contoh penggunaan rumus skor levenshtein:

$$Skor\ levenshtein = \left(1 - \frac{5}{7}\right) \times 100 = 28{,}57$$

3.4.3. Menghitung skor similar text

$$Skor\ similar\ text = similar\_text(str1, str2, perc) \qquad (3)$$

Contoh penggunaan rumus skor similar text (str1= “firaun”, str2= “pringon”):

$$Skor\ similar\ text = 30{,}77$$

3.4.4. Menghitung jumlah kemunculan

$$Skor\ jum\ kemunculan = \frac{jum\ muncul}{jum\ total} \times 100 \qquad (4)$$

Tabel 1. Hasil *Named Entity* Bahasa Suku dari 3 *Tools*

Kata indo	Efmaral	Efmaral _rev	Giza	Giza _rev	Fast align	Fast align _rev
Firaun	Lajeng, **Pringon**, Ingkang	Lajeng, ingkang, sang	**Pringon**, Mbetahake, null	**Pringon**, Kawisudha, Mbetahake	Dening, Ingkang, sang	Ingkang, Sang, putranipun

Contoh penggunaan rumus skor jumlah kemunculan (str1= “firaun”, str2= “pringon”):

$$Skor\ jumlah\ kemunculan = \frac{3}{17} \times 100 = 17{,}65$$

3.4.5. Menghitung probabilitas

$$Prob = \frac{soun + lev + sim + jum\ muncul}{4} \times 100\ \% \qquad (5)$$

Keterangan rumus:
soun=skor soundex
lev=skor levenshtein
sim=skor similar text
jum muncul=skor jumlah kemunculan
prob=probabilitas
Contoh penggunaan rumus probabilitas (str1= “firaun”, str2= “pringon”):

$$prob = \frac{50 + 28{,}57 + 30{,}77 + 17{,}65}{4} \times 100\% = 31{,}725\ \%$$

4. PENGUJIAN SISTEM

Standar yang digunakan untuk mengukur hasil *performance* dari sistem *Natural Language Processing* (NLP) yaitu *precision*, *recall*, dan *f-measure*. *Precision* adalah tingkat ketepatan antara informasi yang diminta oleh pengguna dengan jawaban yang diberikan oleh sistem. *Recall* adalah tingkat keberhasilan sistem dalam menemukan kembali sebuah informasi. *F-Measure* (F_1) merupakan salah satu perhitungan evaluasi yang mengkombinasikan *recall* dan *precision*. Pengukuran *precision* dan *recall* dapat dilakukan dengan rumus [9]:

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \quad (6)$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \quad (7)$$

$$F - measure = 2 \times \frac{precision \times recall}{precision + recall} \quad (8)$$

BLEU (*Bilingual Evaluation Understudy*) digunakan untuk mengevaluasi kualitas dari sebuah hasil terjemahan yang telah diterjemahkan oleh mesin dari satu bahasa alami ke bahasa lain. Rumus BLEU sebagai berikut [3]:

$$BP_{BLEU} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-\frac{r}{c})} & \text{if } c \leq r \end{cases} \quad (9)$$

$$P_n = \frac{\sum C_{\in corpus\ n-gram \in C} \sum count_{clip}(n-gram)}{\sum C_{\in corpus\ n-gram \in C} \sum count(n-gram)} \quad (10)$$

$$BLEU = BP_{BLEU} . e^{\sum_{n=1}^{N} w_n \log p_n} \quad (11)$$

Keterangan:
BP = *brevity penalty* (konstanta)
c = jumlah kata dari hasil terjemahan otomatis
r = jumlah kata rujukan
Pn = modified precission score
wn = 1/N (standar nilai N untuk BLEU adalah 4)
pn = jumlah n-gram hasil terjemahan yang sesuai dengan rujukan dibagi jumlah n-gram hasil terjemahan.

Untuk pengujian hasil, ada 3 macam data *test* yang digunakan untuk 1 kitab di Perjanjian Lama dan 1 kitab di Perjanjian Baru. Untuk kitab Perjanjian Lama yang digunakan adalah Kitab Hosea, sedangkan untuk Perjanjian Baru adalah Kitab Ibrani. Data *test* yang digunakan ada 3 macam yaitu data *test* yang merupakan hasil *named entity* Bahasa Suku yang menggunakan *efmaral tool*, *giza++ tool*, dan gabungan *(efmaral, giza++*, dan *fast align*). Data *test* dihasilkan berdasarkan Alkitab versi Bahasa Indonesia dan Alkitab versi Bahasa Suku. Alkitab versi Bahasa Indonesia yang digunakan untuk Perjanjian Lama adalah Terjemahan Baru (TB), sedangkan Alkitab versi Bahasa Indonesia untuk Perjanjian Baru adalah Alkitab Yang Terbuka (AYT). Alkitab versi Bahasa Suku yang digunakan adalah versi formal atau versi sehari-hari. Data *gold standard* diperoleh dari 5 narasumber yang mengerti Bahasa Suku yang digunakan dalam penelitian ini.

Pengujian dilakukan berdasarkan perhitungan akurasi dan BLEU *score* antara kata sumber (Bahasa Indonesia) dan kata target (Bahasa Suku). Pengujian dilakukan dengan menghitung jumlah *name entity* yang unik. Perhitungan akurasi dengan cara membagi jumlah yang benar dengan jumlah keseluruhan.

Tabel 2 dan Tabel 3 merupakan salah satu hasil pengujian akurasi dan BLEU *score* yang berasal dari *efmaral*, *giza++*, gabungan (*efmaral*, *giza++*, *fast align*), serta gabungan dan *stemming*. Untuk hasil pengujian akurasi dan BLEU *score* Bahasa Suku lainnya, berasal dari *efmaral*, *giza++*, dan gabungan (*efmaral*, *giza++*, *fast align*).

Hasil pengujian Bahasa Jawa 1981 Kitab Hosea dapat dilihat pada Tabel 2. Akurasi paling baik merupakan hasil *option* soun yaitu giza++ 0.72. BLEU yang paling baik merupakan hasil dari *option* soun, lev, dan sim yaitu giza++ sebesar 0.90. Hasil akurasi paling baik dari gabungan *(efmaral, giza++*, dan *fast align*) merupakan hasil dari *option* soun dan lev yaitu sebesar 0.68. BLEU *score* paling baik dari gabungan *(efmaral, giza++*, dan *fast align*) merupakan hasil dari *option* lev yaitu sebesar 0.88.

Tabel 2. Hasil Pengujian Bahasa Jawa 1981 Kitab Hosea

OPTION	AKURASI			BLEU		
	Efm-aral	Gi-za	Gab-ungan	Efma-ral	Gi-za	Gab-ung-an
Soun	0.18	0.61	0.46	0.50	0.86	0.73
Lev	0.39	**0.72**	0.68	0.65	**0.90**	0.86
Sim	0.42	0.71	0.68	0.70	**0.90**	0.88
Soun, lev, sim	0.37	0.69	0.66	0.69	**0.90**	0.86
Soun, lev, sim+ Stemming	0.39	0.71	0.67	0.70	0.89	0.87

Hasil pengujian Bahasa Jawa 1981 Kitab Ibrani dapat dilihat pada Tabel 3. Akurasi paling baik merupakan hasil *option* soun dan gabungan dari semua *option* (soun, lev, dan sim) yaitu giza++ sebesar 0.85. BLEU yang paling baik merupakan hasil gabungan dari semua *option* (soun, lev, dan sim) yaitu giza++ sebesar 0.93. Hasil akurasi paling baik dari gabungan *(efmaral, giza++*, dan *fast align*) merupakan hasil dari *option* soun dan gabungan dari semua *option* yaitu sebesar 0.83. BLEU *score* paling baik dari gabungan *(efmaral, giza++*, dan *fast align*) merupakan hasil dari *option* soun, lev, dan gabungan dari semua *option* yaitu sebesar 0.92.

Tabel 3. Hasil Pengujian Bahasa Jawa 1981 Kitab Ibrani

OPTION	AKURASI			BLEU		
	Efm-aral	Gi-za	Gab-ungan	Efma-ral	Gi-za	Gab-ung-an
Soun	0.27	0.72	0.51	0.63	0.87	0.76
Lev	0.63	**0.85**	0.83	0.86	0.92	0.92
Sim	0.64	0.83	0.81	0.88	0.92	0.92
Soun, lev, sim	0.62	0.83	0.79	0.86	0.92	0.91
Soun, lev, sim+ Stemming	0.66	**0.85**	0.83	0.87	**0.93**	0.92

Dari hasil pengujian untuk Bahasa Jawa 1981 Kitab Hosea dan Kitab Ibrani memiliki beberapa kesamaan akurasi untuk soun dan gabungan dikarenakan Bahasa Jawa memiliki kedekatan dengan Bahasa Indonesia, sehingga penyebutan (*soundex*) memiliki banyak kemiripan. Sedangkan dalam penulisan katanya tidak terlalu mirip sehingga hasil akurasi dan BLEU *score* yang didapat oleh *similar text* rendah. Selain itu, hasil terbaik didapatkan dari hasil *giza++* dikarenakan hasil korelasi *tool giza++* dapat memenuhi 2 kata dalam Bahasa Suku yang sama dan tepat untuk setiap kata dari Bahasa Indonesia. Untuk hasil gabungan juga memiliki akurasi yang cukup tinggi karena hasil konsensus yang sama lebih dari 3. Sedangkan, hasil akurasi dan BLEU *score* untuk *efmaral* rendah dikarenakan hasil korelasi dari *tool efmaral* yang tidak memenuhi konsensus sama dengan 2, sehingga tidak ditemukan kecocokan antara data *test* dan data *gold standard*.

Untuk Bahasa Jawa 1981, dilakukan percobaan untuk *stemming*. Hasil akurasi *stemming* pada Kitab Hosea yang merupakan gabungan dari semua *option* (soun, lev, dan sim) yaitu *efmaral* sebesar 0.38, *giza++* sebesar 0.62, dan gabungan sebesar 0.60. Sedangkan, hasil BLEU *score* untuk efmaral sebesar 0.71, *giza++* sebesar 0.83, dan gabungan sebesar 0.84. Hasil akurasi *stemming* pada Kitab Ibrani yang merupakan gabungan dari semua *option* (soun, lev, dan sim) yaitu *efmaral* sebesar 0.59, *giza++* sebesar 0.79, dan gabungan sebesar 0.80. Sedangkan, hasil BLEU *score* untuk *efmaral* sebesar 0.85, *giza++* sebesar 0.91, dan gabungan

sebesar 0.91. Berdasarkan tab el pengujian untuk Bahasa Jawa 1981, data berupa kata yang sudah di *stemming* dengan data berupa kata yang tidak di *stemming* hasilnya tidak terlalu signifikan, bahkan lebih rendah hasil akurasinya. Tabel 4 merupakan hasil pengujian akurasi dan BLEU *score* yang paling baik diantara *tool efmaral*, *tool giza++*, dan gabungan.

Tabel 4. Keseluruhan Hasil Pengujian

Bahasa	Perjanjian Lama		Perjanjian Baru	
	Akurasi	BLEU	Akurasi	Akurasi
Bahasa Jawa 1981	0.72	0.90	0.85	0.93
Bahasa Sunda (BIS)	0.70	0.86	0.85	0.93
Bahasa Toraja	0.64	0.86	0.87	0.95
Bahasa Batak Simalungun	0.60	0.87	0.90	0.95
Bahasa Lunbawang	0.67	0.88	0.81	0.93

Secara keseluruhan, hasil pengujian menggunakan perhitungan akurasi memiliki presentase kisaran antara 0.60 hingga 0.90. Sedangkan, hasil pengujian menggunakan BLEU *score* memiliki presentase kisaran antara 0.86 hingga 0.95. Dari hasil pengujian, tidak dapat diambil konklusi mana akurasi yang paling baik, dikarenakan pengujian hanya dilakukan pada 2 Kitab yaitu Perjanjian Lama (Hosea) dan Perjanjian Baru (Ibrani). Selanjutnya, dilakukan pengujian untuk menghitung akurasi apakah kata target (dalam Bahasa Suku) merupakan *named entity* atau bukan. Hasil pengujiannya menggunakan *confusion matrix* dilakukan hanya untuk Bahasa Jawa 1981 Perjanjian Baru dengan mengambil semua data yang unik untuk kitab Ibrani dapat dilihat pada Tabel 5.

Tabel 5. Hasil Pengujian *Confusion Matrix* Bahasa Jawa 1981 Perjanjian Baru dengan Data yang Unik

		Actual Values	
		Positive	*Negative*
Predicted Values	*Positive*	TP = 35	FP = 3
	Negative	FN = 4	TN = 1521

$$Presisi = \frac{35}{35+3} = 0{,}92$$

$$Recall = \frac{35}{35+4} = 0{,}89$$

$$f1-score = 2 \times \frac{0{,}92 \times 0{,}89}{0{,}92 + 0{,}89} = 2 \times \frac{0{,}8188}{1{,}81} = 0{,}90$$

Dari hasil pengujian menggunakan *confusion matrix* untuk Bahasa Jawa 1981 Perjanjian Baru dengan data yang unik didapatkan hasil presisi sebesar 0.92, *recall* sebesar 0.89, dan *f1-score* sebesar 0.90.

5. KESIMPULAN

Dari hasil pengujian sistem yang telah dilakukan, dapat diambil beberapa kesimpulan antara lain:

- Hasil akurasi *named entity* yang dihasilkan menggunakan *efmaral tool* lebih rendah dibandingkan menggunakan *giza++ tool*.
- Hasil akurasi *named entity* dengan menggunakan penggabungan *efmaral*, *giza++*, dan *fast align tools* lebih baik dibandingkan hasil akurasi *named entity* menggunakan *efmaral tool*. Sedangkan, hasil akurasi *named entity* dengan menggunakan penggabungan *efmaral*, *giza++*, dan *fast align tools* hampir sama dengan hasil akurasi *named entity* menggunakan *giza++ tool*.
- Berdasarkan tabel pengujian untuk Bahasa Jawa 1981, data berupa kata yang sudah di*stemming* dengan data berupa kata yang tidak di*stemming* hasilnya tidak terlalu signifikan, bahkan lebih rendah hasil akurasinya.
- Akurasi yang dihasilkan *efmaral tool* sebesar berkisar antara 0.07 sampai 0.66. Akurasi yang dihasilkan dari *giza++ tool* sebesar berkisar antara 0.47 sampai 0.90. Akurasi yang dihasilkan dari gabungan *tools (efmaral, giza++, fast align)* berkisar antara 0.36 sampai 0.87.
- BLEU *score* yang dihasilkan *efmaral tool* sebesar berkisar antara 0.24 sampai 0.88. BLEU *score* yang dihasilkan dari *giza++ tool* sebesar berkisar antara 0.80 sampai 0.95. BLEU *score* yang dihasilkan dari gabungan *tools (efmaral, giza++, fast align)* berkisar antara 0.55 sampai 0.95.
- Hasil pengujian menggunakan *confusion matrix* untuk Bahasa Jawa 1981 Perjanjian Baru dengan semua data per ayat didapatkan hasil presisi sebesar 0.33, *recall* sebesar 0.67, dan *f1-score* sebesar 0.44. Sedangkan, hasil pengujian menggunakan *confusion matrix* untuk Bahasa Jawa 1981 Perjanjian Baru dengan data yang unik didapatkan hasil presisi sebesar 0.92, *recall* sebesar 0.89, dan *f1-score* sebesar 0.90.
- Berdasarkan hasil kuesioner yang diberikan kepada orang yang bidang *biblical computing* dan mengerti permasalahan dari penelitian ini, penilaian bahwa aplikasi telah menjawab kebutuhan adalah 5 (sangat baik) dan keseluruhan aplikasi adalah 4 (baik).

Saran untuk pengembangan kedepannnya adalah:

- Penentuan bobot yang tepat dalam perhitungan probabilitas agar dapat meningkatkan hasil akurasi.
- Penggunaan *database* yang lebih tepat agar dapat menampung data *named entity*, misalnya NoSQL *database*.
- Penambahan fitur edit untuk admin, agar dapat memperbaiki kesalahan penulisan dalam teks Alkitab Bahasa Suku.
- Penentuan *named entity* bisa lebih diperdalam dengan memahami arti dari kata, karena ada beberapa kata yang sama tetapi memiliki arti yang berbeda.

6. DAFTAR PUSTAKA

[1] Dyer, C., Chahuneau, V., & Smith, N. A. 2013. A simple, fast, and effective reparameterization of IBM model 2. *NAACL HLT 2013 - 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Main Conference*, (June), 644–648.

[2] Hagiwara, M. Using GIZA++ to Obtain Word Alignment Between Bilingual Sentences. URI= http://masatohagiwara.net/using-giza-to-obtain-word-alignment-between-bilingual-sentences.html

[3] Jarob, Y., Sujaini, H., & Safriadi, N. 2017. Uji Akurasi Penerjemahan Bahasa Indonesia – Dayak Taman Dengan Penandaan Kata Dasar Dan Imbuhan. *Jurnal Edukasi Dan Penelitian Informatika (JEPIN)*, *2*(2), 78–83. DOI= https://doi.org/10.26418/jp.v2i2.16520

[4] Lee, D., Park, S., Jung, N., & Chun, M. 2007. *for Data Modeling*. 224–231.

[5] Moses - Moses/Overview. 2013. URI= http://www.statmt.org/moses/?n=Moses.Overview

[6] Oliver, I. 1993. Programming classics : implementing the world's best algorithms. Prentice Hall.

[7] Östling, R., & Tiedemann, J. 2016. The Prague Bulletin of Mathematical Linguistics NUMBER 106 OCTOBER 2016 125-146 Efficient Word Alignment with Markov Chain Monte Carlo. DOI= https://doi.org/10.1515/pralin-2016-0013

[8] PHP: soundex-Manual. URI= https://www.php.net/manual/en/function.soundex.php

[9] Ping Shung, K. Accuracy, Precision, Recall or F1? – Towards Data Science. 2018. URI= https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9

[10] Wu, W., Vyas, N., & Yarowsky, D. 2018. Creating a Translation Matrix of the Bible ' s Names Across 591 Languages. *11th Edition of Its Language Resources and Evaluation Conference (LREC 2018)*, 1659–1665.

[11] Yulaiandaru, A. 2015. Penerapan String Matching Pada Auto-Correct Berbasis Algoritma Levenshtein Distance.