BAB 2 LANDASAN TEORI

Bab ini menjelaskan tentang landasan teori yang digunakan dalam penelitian ini. Pembahasan dimulai dari sejarah *machine translation* (subbab 2.1), *statistical machine translation* (subbab 2.2), *automatic evaluation* (subbab 2.3), *word reordering* dan *phrase reordering* (subbab 2.4), POS *tagging* (subbab 2.5), dan *parsing* (subbab 2.6).

2.1 Sejarah Machine Translation

Machine translation atau mesin penerjemah merupakan alat penerjemah otomatis pada sebuah teks dari satu bahasa ke bahasa lainnya. Mesin penerjemah juga merupakan salah satu aplikasi yang penting dalam bidang *Natural Languange Processing* (NLP). *Machine translation* melakukan substitusi sederhana dari suatu kata dalam suatu bahasa ke bahasa lain. Ada beberapa pendekatan untuk *machine translation* seperti pendekatan dengan menggunakan aturan (*rule-based machine translation*), pendekatan dengan menggunakan contoh (*example-based machine translation*), dan pendekatan dengan menggunakan model statistik (*statistical machine translation*).

Rule-based machine translation merupakan strategi pendekatan pertama yang digunakan dalam penelitian *machine translation*. Keuntungan dari *Rule-based machine translation* adalah aturan – aturan bahasa dapat menganalis dalam level semantik dan sintaks secara dalam. Kelemahan dari *Rule-based machine translation* adalah membutuhkan banyaknya pengetahuan linguistik yang mendalam pada sebuah bahasa. (Charoenpornsawat et al., 2002).

Example-based machine Translation merupakan *machine translation* yang menggunakan pendekatan berdasarkan pada pasangan koleksi dokumen (korpus) paralel. *Example-based machine translation* dirancang oleh Nagao Makoto pada tahun 1984. Prinsip dasar dari penerjemahan *example-based machine Translation* adalah ide penerjemahan dengan analogi (Makoto, 1984). Berikut adalah contoh korpus paralel.

Tabel 2.1 Contoh Korpus Paralel

Inggris	Indonesia
How much is that red umbrella?	Berapa harga payung merah itu?
How much is that small camera?	Berapa harga kamera kecil itu?

Pada Tabel 2.1 terdapat sebuah contoh korpus dwibahasa dengan jumlah pasangan kalimat yang minimal, yang berarti bahwa kalimat-kalimat tersebut hanya dibedakan oleh satu elemen saja. *Example-based machine Translation* akan mempelajari tiga unit terjemahan sebagai berikut:

1. ”*How much is that* **X** ?” ⟷ ”Berapa harga **X** itu ?”
2. ”*red umbrel l a*” ⟷ ” payung merah”
3. ”*small ca mera* ” ⟷ ” kamera kecil”

Pembentukan tiga unit di atas dapat digunakan untuk menghasilkan terjemahan-terjemahan baru untuk ke depannya. Sebagai contoh jika sistem telah dilatih dengan menggunakan beberapa teks yang berisikan kalimat ”*President Kennedy was shot dead during the parade.*” dan ”*The convict escaped on July 15th.*”, maka kalimat ”*The convict was shot dead during the parade.*” dapat diterjemahkan dengan mensubstitusikan bagian-bagian yang tepat dari kalimat-kalimat tersebut. (Makoto, 1984)

Statistical machine translation diperkenalkan pertama kali oleh Warren Weaver pada tahun 1949. Setelah itu, *statistical machine translation* dipopulerkan kembali pada tahun 1980 oleh IBM pada Candide Project (Koehn & Callison-Burch, 2005). *Statistical machine translation* adalah suatu paradigma mesin penerjemah dimana hasil terjemahan dihasilkan atas dasar model statistik dengan menggunakan parameter yang didapatkan dari analisis kumpulan teks dua bahasa yang paralel. *Statistical machine translation* menggunakan korpus paralel tersebut untuk menghasilkan sebuah model penerjemahan statistik secara otomatis. Model statistik tersebut kemudian akan diterapkan pada teks-teks yang baru untuk menghasilkan terjemahan yang baik dari teks baru tersebut. Penjelasan lebih lanjut tentang *statistical machine translation* dijelaskan pada subbab 2.2.

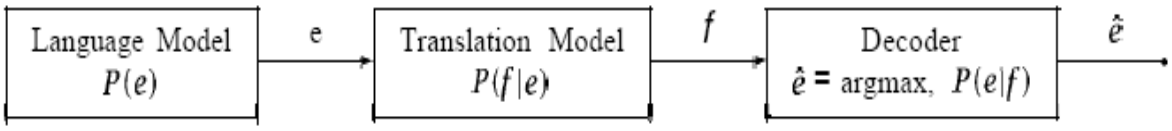
2.2 Statistical Machine Translation

Statistical machine translation atau mesin penerjemah statistik merupakan salah satu jenis mesin penerjemah dengan menggunakan pendekatan statistik. Pendekatan statistik yang digunakan adalah konsep probabilitas. Setiap pasangan kalimat (*S*,*T*) akan diberikan sebuah $P(T|S)$ yang diinterpretasikan sebagai distribusi probabilitas dimana sebuah penerjemah akan menghasilkan *T* dalam bahasa tujuan ketika diberikan *S* dalam bahasa sumber.

Sebagai contoh, proses penerjemahan dari kalimat Bahasa Prancis ke Bahasa Inggris dapat diinterpretasikan dengan $P(e|f)$. $P(e|f)$ merupakan distribusi probabilitas kalimat e (Bahasa Inggris) terhadap kalimat f (Bahasa Perancis). Dengan menggunakan teorema Bayes pada $P(e|f)$, didapatkan:

$$P(e|f) = \frac{P(f|e).P(e)}{P(f)} \quad \text{... (2.1)}$$

Menurut Christopher D Manning dan Hinrich Schutze, dalam *statistical machine translation* terdapat 3 buah komponen yang terlibat dalam proses penerjemahan kalimat dari suatu bahasa ke bahasa lain, yaitu: *language model*, *translation model*, dan *decoder* (Manning & Schutze ,2000).



Gambar 2.1 Komponen Statistical Machine Translation

Gambar 2.1 merupakan komponen *statistical machine translation*. Penjelasan lebih lanjut tentang *language model* terdapat pada subbab 2.2.1. Penjelasan lebih lanjut tentang *translation model* terdapat pada subbab 2.2.2. Penjelasan lebih lanjut tentang *decoder* terdapat pada subbab 2.2.3.

2.2.1 *Language Model*

Language model digunakan pada aplikasi *Natural Language Processing* seperti *speech recognition*, *part-of-speech tagging* dan *syntactic parsing*. Saat ini, *language model* juga telah banyak digunakan untuk bidang *Information Retrieval*. Berdasarkan pendekatan statistik, masing-masing dokumen dipandang sebagai sebuah sampel bahasa dan sebuah kueri sebagai sebuah proses generasi (*generation*). Dalam *language model* statistik, elemen-elemen kuncinya adalah probabilitas dari rangkaian-rangkaian kata yang ditunjukkan sebagai $P(w_1, w_2, \dots, w_n)$ atau singkatnya $P(w_{1,n})$. *Language model* statistik menetapkan probabilitas $P(w_{1,n})$ ke serangkaian *n* kata dengan *means* sebuah distribusi probabilitas. Rangkaian-rangkaian tersebut bisa berupa frase-frase atau kalimat-kalimat dan probabilitasnya dapat diperkirakan dari korpus dokumen-dokumen yang besar. Salah satu contoh pendekatan *language model* adalah *n-gram model*. Dalam sebuah *n-gram model*, probabilitas $P(w_{1,m})$ dari mengamati kalimat $w_1, w_2, \dots, w_m$ diaproksimasikan sebagai (Song & Croft, 1999) :

$$P(w_1, w_2, \dots, w_m) = \prod_{i=1}^{m} P(w_i | w_1, \dots, w_{i-1}) \approx \prod_{i=1}^{m} P(w_i | w_{i-(n-1)}, \dots, w_{i-1}) \quad \dots (2.2)$$

Pada persamaan (2.2) diasumsikan bahwa probabilitas dari mengamati kata w_i ke-*i* dalam sejarah konteks *i*-1 kata-kata terdahulu, dapat diaproksimasikan oleh probabilitas mengamatinya dalam sejarah konteks *n*-1 kata-kata terdahulu yang diperpendek (orde ke-*n* properti Markov).

Probabilitas bersyarat (*conditional*) dapat dihitung dari perhitungan frekuensi n-gram:

$$P(w_i | w_{i-(n-1)}, \dots, w_{i-1}) = \frac{count(w_{i-(n-1)}, w_{i-1} \dots, w_i)}{count(w_{i-(n-1)}, \dots, w_{i-1})} \quad \dots (2.3)$$

Beberapa contoh model bahasa n-gram adalah:

- Unigram (1-gram): $P(w_{1,n}) = P(w_1)P(w_2) \dots P(w_n)$.
- Bigram (2-gram): $P(w_{1,n}) = P(w_1)P(w_2|w_1) \dots P(w_n|w_{n-1})$.

- Trigram (3-gram): $P(w_{1,n}) = P(w_1)P(w_2|w_1)P(w_3|w_{1,2}) \dots P(w_n|w_{n-2,n-1})$

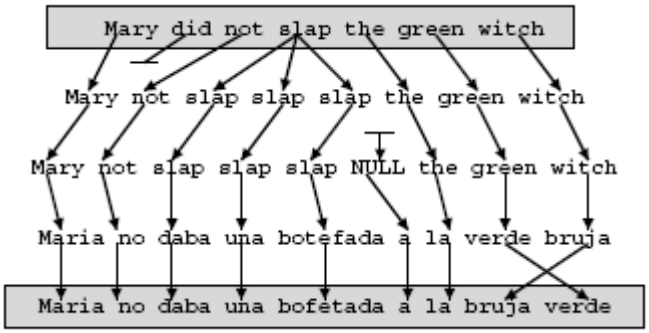
Sebagai contoh, dalam model bahasa trigram, probabilitas dari kalimat "Mary did not slap the green witch" yang diambil dari artikel Kevin Knight dan Philipp Koehn (Knight & Koehn, 2003) dihitung dengan cara :

P(Mary, did, not, slap, the, green, witch) ≈ P(Mary) P(did|Mary) P(not|Mary, did) P(slap|did, not) P(the|not, slap) P(green|slap, the) P(witch|the, green)

Merujuk pada komponen *statistical machine translation* (Gambar 2.1), simbol $P(e)$ yang merupakan sebuah *language model* adalah probabilitas dari kalimat Bahasa Inggris. *Language model* $P(e)$ berperan untuk menentukan apakah kalimat hasil terjemahan e adalah kalimat Bahasa Inggris yang natural dan memiliki aturan gramatikal yang baik. Semakin tinggi nilai probabilitas dari $P(e)$ menunjukkan bahwa kalimat hasil terjemahan merupakan kalimat yang dibentuk dengan baik.

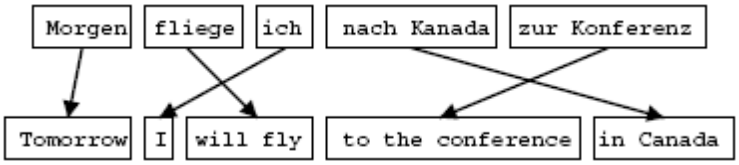
2.2.1 *Translation Model*

Translation model merupakan salah satu komponen penting pada *statistical machine translation* dalam proses penerjemahan. Terdapat 2 model penerjemahan pada *statistical machine translation*, yaitu: *word-based translation model* dan *phrase-based translation model*.



Gambar 2.2 Ilustrasi Penerjemahan dengan *Word-Based Translation Model*

Gambar 2.2 merupakan ilustrasi *word based translation model*. Proses yang terjadi pada *word based translation model* adalah: penerjemahan kata, *reordering*, *duplication*, dan *insertion* (Koehn, Phillip, 2005).



Gambar 2.3 Ilustrasi Penerjemahan dengan *Phrase-Based Translation Model*

Gambar 2.3 merupakan ilustrasi *phrase based translation model*. Proses penerjemahan dapat dipecah menjadi beberapa bagian yaitu: membagi kalimat bahasa asal menjadi barisan frase, menerjemahkan setiap frase ke bahasa tujuan, dan *reordering*. (Koehn, Phillip, 2005).

Menurut penelitian, hasil terjemahan dari *phrase-based statistical machine translation* lebih baik dan berkualitas daripada hasil terjemahan dari *word-based statistical machine translation*. (Koehn, Philipp; Franz Josef Och, and Daniel Marcu, 2003).

Merujuk pada komponen *statistical machine translation* (Gambar 2.1), Simbol $P(f|e)$ merupakan *translation model*. Tujuan dari *translation model* adalah untuk memasangkan teks *input* dalam bahasa asal (*source language*) dengan teks *output* dalam bahasa tujuan (*target language*).

Pada *word-based translation model*, merujuk pada persamaan (2.1), simbol $P(f|e)$ dapat diubah menjadi::

$$P(f|e) = \frac{1}{Z} \sum_{a_1=0}^{l} \dots \sum_{a_m=0}^{l} \prod_{j=1}^{m} P(f_j|e_{aj}) \qquad \dots (2.4)$$

Simbol e adalah kalimat Bahasa Inggris, f adalah kalimat Bahasa Perancis, l adalah panjang kalimat Bahasa Inggris, m adalah panjang kalimat Bahasa Perancis, f_j adalah kata j dalam Bahasa Perancis, a_j adalah posisi kata dalam e yang berpasangan dengan f_j, e_{aj} adalah kata yang berpasangan dengan f_j, dan Z

adalah konstanta normalisasi. Secara singkat, P(f|e) didapatkan dari penjumlahan probabilitas pencocokan setiap kata. (Manning, Christopher D. and Hinrich Schutze, 2000).

Sedangkan pada *phrase-based translation model*, merujuk pada persamaan (2.1), simbol $P(f|e)$ dapat diubah menjadi:

$$P(\bar{f}_1^I|\bar{e}_1^I) = \prod_{i=1}^{I} \phi(\bar{f}_i|\bar{e}_i) d(a_i - b_{i-1}) \quad \text{... (2.5)}$$

Pada *phrase-based translation model*, kalimat *input* f dibagi ke dalam satu barisan dari I frase – frase, $\bar{f}_1^I$. Setiap frase $\bar{f}_i$ dalam $\bar{f}_1^I$. diterjemahkan menjadi frase Bahasa Inggris $\bar{e}_i$. Frase-frase dalam Bahasa Inggris tersebut akan disusun kembali (*reorder*). Penerjemahan frase dimodelkan oleh distribusi probabilitas $\phi(\bar{f}_i|\bar{e}_i)$. Proses penyusunan ulang dari frase Bahasa Inggris dimodelkan dengan *relative distortion probability distribution* $d(a_i - b_{i-1})$, dimana a_i menunjukkan posisi awal dari frase bahasa asal (*source language*) yang diterjemahkan menjadi frase bahasa tujuan (*target language*), dan b_{i-1} menunjukkan posisi akhir dari frase bahasa asal (*source language*) yang diterjemahkan menjadi frase bahasa tujuan (*target language*) ke i-1. *Relative distortion probability distribution* $d(a_i - b_{i-1})$, dilatih dengan menggunakan sebuah model *joint probability*. (Koehn, Philipp; Franz Josef Och, and Daniel Marcu, 2003)

2.2.3 Decoder

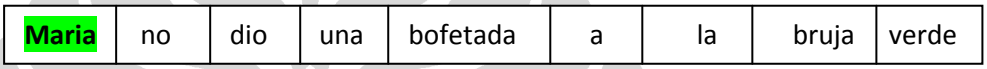
Fungsi dari *decoder* adalah untuk mencari teks dalam bahasa tujuan (*target language*) yang memiliki probabilitas paling besar dengan pertimbangan faktor *translation model* dan *language model*. Penghitungan ê dapat dituliskan sebagai berikut:

$$\hat{e} = arg_e \, max \, P(e|f) = arg_e \, max \, \frac{P(f|e).P(e)}{P(f)} = arg_e \, max \, P(f|e) \, P(e) \quad \text{...(2.6)}$$

$Arg_e\ max$ mencari e yang dapat memberikan nilai probabilitas terbesar yang diperoleh. Simbol $P(f)$ dapat dihilangkan karena f tetap (Manning, Christopher D. and Hinrich Schutze ,2000).

Secara umum, proses *decoding* membangun kalimat terjemahan dari kiri ke kanan. Proses decoding ini menggunakan algoritma *beam search*. Dua konsep penting dalam algoritma *beam search* yang digunakan adalah konsep pemangkasan (*pruning*) dan estimasi *future cost*. Berikut ini adalah contoh proses penerjemahan pada *phrase-based statistical machine translation* (Koehn, 2006) :

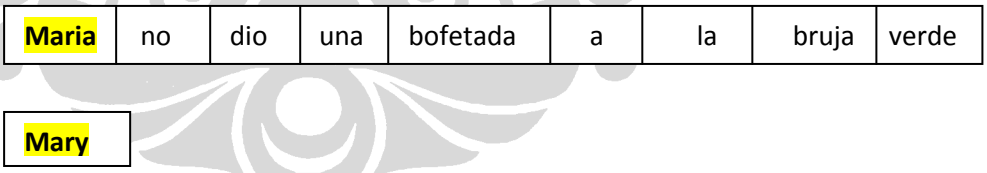
a. Pilih kata-kata asing dalam bahasa sumber (misalnya: bahasa Spanyol) untuk diterjemahkan



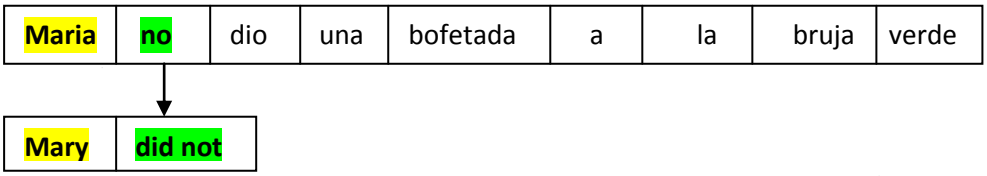
b. Temukan terjemahan frase dalam bahasa target (misalnya: bahasa Inggris)

c. Tambahkan frase bahasa Inggris ke akhir penerjemahan parsial

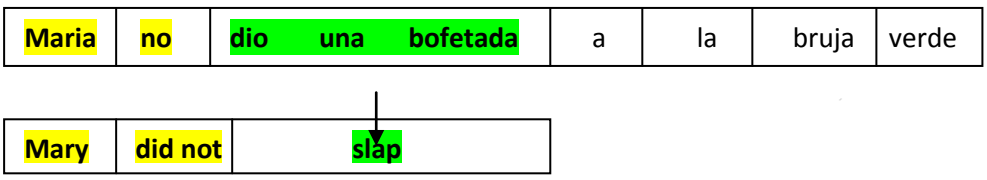
d. Tandai kata-kata asing yang telah diterjemahkan



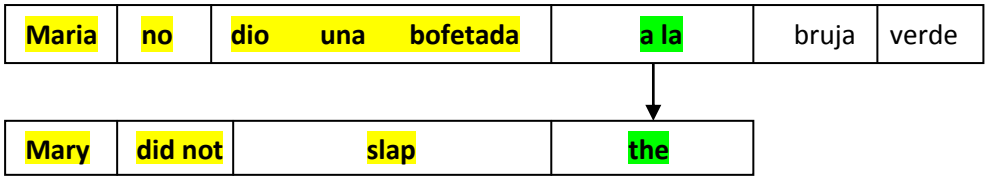
e. Penerjemahan satu-ke-banyak (*one-to-many*)



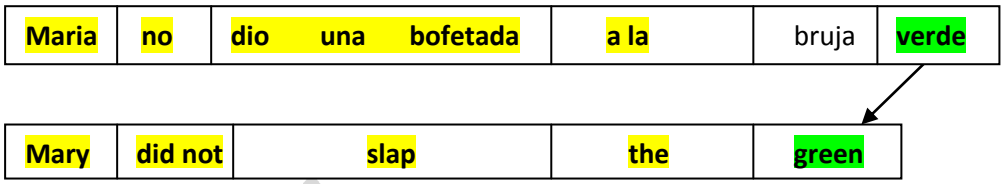
f. Penerjemahan banyak-ke-satu (*many-to-one*)



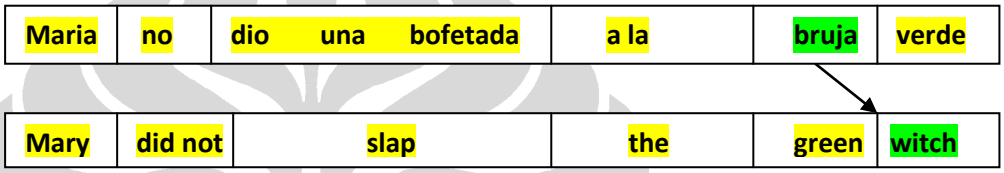
g. Penerjemahan banyak-ke-satu (*many-to-one*)



h. *Reordering* (penyusunan kembali)



i. Penerjemahan selesai



2.3 Automatic Evaluation

Akurasi dan kualitas dari sebuah *statistical machine translation* pada umumnya dinilai dari hasil terjemahan yang dihasilkan. Penilaian tersebut bisa dilakukan secara manual dan otomatis. Penilaian secara manual tentunya merupakan cara penilaian yang terbaik karena memberikan nilai akurasi yang lebih tepat. Namun penilaian secara manual merupakan cara penilaian yang lambat, mahal dan lama sebab membutuhkan sumber daya manusia yang banyak serta waktu yang cukup lama. Oleh karena itu dibutuhkan suatu sistem evaluasi otomatis untuk menentukan kualitas terjemahan yang cepat, murah, dan cukup bagus.

Grup riset dari IBM memperkenalkan suatu sistem evaluasi otomatis yang bernama BLEU (Bilingual Evaluation Understudy). BLEU mengukur *modified* n-gram *precision score* antara hasil terjemahan otomatis dengan terjemahan rujukan dan menggunakan konstanta yang dinamakan *brevity penalty*.

Cara sederhana untuk menghitung *precision* adalah dengan menghitung jumlah kata pada hasil terjemahan (unigrams) yang cocok dengan rujukan dan dibagi dengan total jumlah kata (unigrams) yang ada pada hasil terjemahan. Namun cara tersebut memiliki beberapa kelemahan. *Statistical machine translations* dapat menghasilkan dengan berlebihan kata-kata yang benar yang mengakibatkan dihasilkannya terjemahan yang mustahil namun memiliki *precision* tinggi. Dengan demikian sebuah kata rujukan harus dianggap telah muncul setelah sebuah kata hasil terjemahan yang cocok teridentifikasi. Untuk mencapai tujuan tersebut maka Papineni melakukan beberapa perubahan dan dihasilkanlah metode yang dinamakan *modified* n-gram *precision* (Papineni, Kishore; Salim Roukos; Todd Ward and Wei-Jing Zhu, 2001).

Contoh penghitungan *modified* unigram *precision*:

Hasil Terjemahan	Rujukan
the the the the the the the.	*The cat is on the mat*.

Nilai *precision* unigram adalah = 7/7. Sedangkan nilai *precision* unigram yang dimodifikasi adalah = 2/7. Jumlah maksimum kata ”the” pada referensi adalah 2 dan jumlah unigram pada hasil terjemahan adalah 7.

Brevity penalty digunakan untuk mencegah kalimat pendek memperoleh nilai yang tinggi. Dengan kata lain, BLEU diperoleh dari hasil perkalian antara *brevity penalty* dengan rata – rata geometri dari *modified precision score*. Jadi agar dapat menghasilkan nilai yang tinggi dalam BLEU, panjang kalimat hasil terjemahan harus mendekati panjang kalimat referensi dan kalimat hasil terjemahan harus dapat menghasilkan kata serta urutan yang sama dengan kalimat referensi. Nilai dari BLEU berada dalam rentang 0 sampai 1. Semakin tinggi nilai BLEU, maka semakin akurat dengan rujukan. Rumus menghitung BLEU adalah sebagai berikut (Papineni, Kishore; Salim Roukos; Todd Ward and Wei-Jing Zhu, 2001):

$$\mathrm{BP}_{\mathrm{BLEU}} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \le r \end{cases} \quad ...(2.6)$$

$$p_n = \frac{\sum_{C \in corpus} \sum_{\text{n-gram} \in C} \text{count}_{\text{clip}}(\text{n-gram})}{\sum_{C \in corpus} \sum_{\text{n-gram} \in C} \text{count}\ (\text{n-gram})} \quad ...(2.7)$$

$$\mathrm{BLEU} = \mathrm{BP}_{\mathrm{BLEU}} \cdot \mathrm{e}^{\sum_{n=1}^{N} w_n \log p_n} \quad ...(2.8)$$

Simbol BP merupakan *brevity penalty*, c merupakan jumlah kata dari hasil terjemahan otomatis, r merupakan jumlah kata dari rujukan, dan p_n merupakan *modified precision score*. Nilai w_n adalah 1/N. Standar nilai N untuk BLEU adalah 4, karena nilai presisi BLEU pada umumnya dihitung sampai 4-gram saja. Pada penelitian ini BLEU dihitung sampai 4-gram. Simbol p_n diperoleh dari jumlah n-gram pada hasil terjemahan yang cocok dengan rujukan dibagi dengan jumlah n-gram pada hasil terjemahan.

Misalkan terdapat pasangan hasil kalimat terjemahan dan rujukan sebagai berikut:

Line	Hasil Terjemahan (*Translated*)	Rujukan (*Reference*)
1	Anto suka makan nasi goreng kambing	Anto suka makan nasi goreng dan minum teh
2	Anto suka makan soto	Anto suka makan nasi sate

Tabel 2.2 Contoh Hasil Terjemahan dan Rujukan

Maka nilai BLEU untuk kalimat pertama dari contoh di atas (”Anto suka makan nasi goreng kambing”) adalah 0.6912316215478422. Berikut adalah proses penghitungannya.

- Nilai c adalah 6 dan r adalah 8.
- Oleh karena c $\leq$ r maka nilai dari *brevity penalty* $= e^{(1- r/c)}$
  $= e^{(1- 8/6)}$

$= e^{-2/8}$

$= 0.778800783.$

- Nilai *modified unigram precision* atau p_1 adalah 5/6

Translated :

Anto	suka	makan	nasi	goreng	kambing

Reference :

Anto	suka	makan	nasi	goreng	dan	minum	teh

- Nilai *modified bigram precision* atau p_2 adalah 4/5

Translated :

Anto	suka	makan	nasi	goreng	kambing

Reference :

Anto	suka	makan	nasi	goreng	dan	minum	teh

- Nilai *modified trigram precision* atau p_3 adalah 3/4

Translated :

Anto	suka	makan	nasi	goreng	kambing

Reference :

Anto	suka	makan	nasi	goreng	dan	minum	Teh

- Nilai *modified* 4gram *precision* atau p_4 adalah 2/3

Translated :

Anto	suka	makan	nasi	goreng	kambing

Reference :

Anto	suka	makan	nasi	goreng	dan	minum	teh

$$\text{BLEU} = 0.778800783 . \text{e}^{\left\{\sum_{n=1}^{4}\frac{1}{4}\log p_n\right\}}$$

$$= 0.778800783 . \text{e}^{\left\{\frac{1}{4}\left(\log\frac{5}{6} + \log\frac{4}{5} + \log\frac{3}{4} + \log\frac{2}{3}\right)\right\}}$$

$$= 0.778800783 . \text{e}^{\{-0,119280333679915\}}$$

$$= 0.6912316215478422$$

Selain BLEU, terdapat sistem evaluasi otomatis lainnya yang bernama NIST. NIST (National Institute of Standards and Technology) merupakan perkembangan dari BLEU. NIST juga menggunakan penilaian berdasarkan presisi n-grams. NIST menghitung presisi dari n-gram dengan bobot yang berbeda untuk setiap kata. Semakin langka kemunculan suatu kata n-gram maka semakin besar bobot yang diberikan. Nilai NIST didapatkan dari perkalian *brevity penalty* dengan jumlah dari *precision score*. Berbeda dengan BLEU yang memberikan bobot presisi n-grams yang sama. (Ying Zhang, Stephan Vogel, Alex Waibel, 2004). Sebagai contoh: bobot trigram kata ”oleh karena itu” lebih kecil daripada bobot trigram kata ”lampu lalu lintas”. Hal ini dikarenakan trigram ”oleh karena itu” dianggap lebih sering muncul daripada trigram “lampu lalu lintas”. Rumus untuk menghitung NIST adalah sebagai berikut (Doddington, George, 2002):

$$\text{BP}_{\text{NIST}} = e^{\left\{\beta \log^2\left[\min\left(\frac{L_{sys}}{L_{ref}},1\right)\right]\right\}} \quad ...(2.9)$$

$$\text{Info}(w_1 \ldots w_n) = \log_2\left(\frac{\text{the\#of occurences of } w_1 \ldots w_{n-1} \text{ in all reference}}{\text{the\#of occurences of } w_1 \ldots w_n \text{ in all reference}}\right) \quad ...(2.10)$$

$$\text{NIST} = \sum_{n=1}^{N}\left\{\frac{\sum_{\substack{\text{all } w_1 \ldots w_n \\ \text{that co-occur}}} \text{Info}(w_1 \ldots w_n)}{\sum_{\substack{\text{all } w_1 \ldots w_n \\ \text{in sys output}}} (1)}\right\} . \text{BP}_{\text{NIST}} \quad ...(2.11)$$

Simbol L_{sys} merupakan jumlah kata yang ada pada hasil terjemahan, L_{ref} merupakan rata – rata dari jumlah kata yang ada pada semua rujukan. Standar

nilai N untuk NIST adalah 5, karena nilai presisi NIST pada umumnya dihitung sampai 5-gram saja. Pada penelitian ini NIST dihitung sampai 5-gram. Ketika jumlah kata pada hasil terjemahan adalah 2 dan rata – rata jumlah kata pada semua rujukan adalah 3, diketahui bahwa nilai *brevity penalty* adalah 0.5.

Sehingga nilai konstanta β dapat dihitung dengan persamaan (2.9):

$$0{,}5 = e^{\left\{\beta \log 2\left[\min\left(\frac{2}{3},1\right)\right]\right\}}$$

$$0{,}5 = e^{\left\{\beta \log 2\left[\min\left(\frac{2}{3},1\right)\right]\right\}}$$

$$0{,}5 = e^{\{\beta\ 0{,}03100813\}1}$$

$$-0.69314718 = 0.031008131\ \beta\ .\ \ln e$$

$$-0.69314718 = 0.031008131\ \beta$$

$$-22.35372329 = \beta$$

Menurut penghitungan di atas dapat dikatakan bahwa β bernilai -22,35372329.

Mengacu pada tabel 2.2, maka nilai NIST untuk kalimat pertama (”Anto suka makan nasi goreng kambing”) adalah 1,810890498. Berikut adalah proses penghitungannya:

Nilai L_{sys} adalah 10 dan L_{ref} adalah 12 (karena pada tabel 2.2 hanya terdapat 1 buah berkas rujukan).

Nilai *brevity penalty* $= e^{\left\{-22{,}35372329 \log 2\left[\min\left(\frac{10}{12},1\right)\right]\right\}}$

$$= e^{\left\{-22{,}35372329 \log 2\left[\frac{5}{6}\right]\right\}}$$

$$= 0.869227439.$$

$$\text{Nilai Info(Antosuka)} = \log_2\left(\frac{\#\text{of occurences "Anto"}}{\#\text{of occurences "Antosuka"}}\right) = \log_2\left(\frac{2}{2}\right) = 0$$

$$\text{Nilai Info(sukamakan)} = \log_2\left(\frac{\#\text{of occurences "suka"}}{\#\text{of occurences "sukamakan"}}\right) = \log_2\left(\frac{2}{2}\right) = 0$$

$$\text{Nilai Info(makan nasi)} = \log_2\left(\frac{\#\text{of occurences "makan"}}{\#\text{of occurences "makan nasi"}}\right) = \log_2\left(\frac{2}{2}\right) = 0$$

$$\text{Nilai Info(nasigoreng)} = \log_2\left(\frac{\#\text{of occurences "nasi"}}{\#\text{of occurences "nasigoreng"}}\right) = \log_2\left(\frac{2}{1}\right) = 1$$

$$\text{Nilai Info(Antosukamakan)} = \log_2\left(\frac{\#\text{of occurences "Antosuka"}}{\#\text{of occurences "Antosukamakan"}}\right) = \log_2\left(\frac{2}{2}\right) = 0$$

$$\text{Nilai Info(sukamakan nasi)} = \log_2\left(\frac{\#\text{of occurences "sukamakan"}}{\#\text{of occurences "sukamakan nasi"}}\right) = \log_2\left(\frac{2}{2}\right) = 0$$

$$\text{Nilai Info(makan nasigoreng)} = \log_2\left(\frac{\#\text{of occurences "makan nasi"}}{\#\text{of occurences "makan nasigoreng"}}\right) = \log_2\left(\frac{2}{1}\right) = 1$$

$$\text{Nilai Info(Antosukamakan nasi)} = \log_2\left(\frac{\#\text{of occurences "Antosukamakan"}}{\#\text{of occurences "Antosukamakan nasi"}}\right) = \log_2\left(\frac{2}{2}\right) = 0$$

$$\text{Nilai Info(sukamakan nasigoreng)} = \log_2\left(\frac{\#\text{of occurences "sukamakan nasi"}}{\#\text{of occurences "sukamakan nasigoreng"}}\right) = \log_2\left(\frac{2}{1}\right) = 1$$

$$\text{Nilai Info(Antosukamakan nasigoreng)} = \log_2\left(\frac{\#\text{of occurences "Antosukamakan nasi"}}{\#\text{of occurences "Antosukamakan nasigoreng"}}\right) = \log_2\left(\frac{2}{1}\right) = 1$$

$$\text{NIST} = \sum_{n=1}^{N}\left\{\frac{\sum_{\substack{\text{all } w_1 \ldots w_n \\ \text{that co-occur}}} \text{Info}(w_1 \ldots w_n)}{\sum_{\substack{\text{all } w_1 \ldots w_n \\ \text{in sys output}}} (1)}\right\} . \; 0{,}869227439$$

$$= \left\{\frac{1}{1+1+1+1} + \frac{1}{1+1+1} + \frac{1}{1+1} + \frac{1}{1}\right\} . \; 0{,}869227439$$

$$= \left\{\frac{25}{12}\right\} . \; 0{,}869227439$$

$$= 1{,}810890498$$

2.4 Word reordering dan phrase reordering

Preprocessing merupakan langkah penting dalam aplikasi *natural language*. Contoh dari *preprocessing* adalah *tokenization*, *stemming*, dan lain lain. Pengaturan ulang urutan kata dan frase sebagai salah satu langkah yang diterapkan pada tahap *preprocessing* dapat menunjukkan peningkatan menurut *automatic evaluation* pada mesin penerjemah statistik. Hasil dari *preprocessing* digunakan dalam tahap *training* (pembentukan *language model* dan *translation model)* dan tahap *testing* (*decoding* kalimat).

Maja Popovic dan Hermann Ney melakukan *word reordering* berdasarkan *Part Of Speech* (Popovic, & Ney, 2006). Mereka melakukan penerjemahan dari Bahasa Spanyol ke Bahasa Inggris dan Bahasa Inggris ke Bahasa Jerman. Mereka menerapkan penukaran antara kata sifat dengan kata benda. Hasil penerjemahan dari Bahasa Spanyol ke Bahasa Inggris menunjukkan adanya peningkatan kualitas mesin penerjemah dari 0.274 menjadi 0.284 dengan menggunakan sistem penilaian BLEU. Hasil penerjemahan dari Bahasa Inggris ke Bahasa Jerman menunjukkan adanya peningkatan kualitas mesin penerjemah dari 0.196 menjadi 0.204 dengan menggunakan sistem penilaian BLEU. Contoh percobaan yang dilakukan oleh Maja Popovic dan Hermann Ney:

Bahasa Spanyol	este sistema no sea susceptible de ser usado como **arma** politica.
Word Reordering	este sistema no sea susceptible de ser usado como politica **arma**
Bahasa Inggris	the system cannot be used as a political weapon

Penerapan *phrase reordering* pada korpus paralel Bahasa Jerman dan Bahasa Inggris dengan bantuan *parser* dilakukan oleh Collins pada tahun 2005 (Collins, Koehn, & Kucerova, 2005). Collins menerapkan enam aturan yang dirancang secara manual, yaitu: Verb Initial, Verb 2and, Move Subject, Particles, Infinitives, dan Negation untuk penyusunan ulang suatu kalimat. Hasil penelitian Collins tersebut menunjukkan adanya peningkatan kualitas mesin penerjemah dari

0.252 menjadi 0.268 dengan menggunakan sistem penilaian BLEU. Contoh percobaan yang dilakukan oleh Collins dengan aturan Verb Initial:

Bahasa Jerman	Ich werde Ihnen die entsprechenden Anmerkungen **aushaendigen**
Phrase Reordering	Ich werde **aushaendigen** Ihnen die entsprechenden Anmerkungen
Bahasa Inggris	I shall be passing on to you some comments

Penerapan *phrase reordering* dengan bantuan *parser* juga dilakukan oleh Sangodkar (Sangodkar, Vasudevan, & Damani., 2008). Korpus paralel yang digunakan oleh Sangodkar adalah korpus paralel Bahasa Inggris dan Bahasa India. Sangodkar melakukan penukaran antar frase dengan aturan parent-child positioning dan relation priority. Hasil yang didapatkan dari percobaan dengan menggunakan data set IIIT adalah adanya peningkatan nilai BLEU dari 0.0815 menjadi 0.0836. Akan tetapi apabila menggunakan standar NIST maka hasil yang diperoleh menunjukkan penurunan nilai dari 3.9036 menjadi 3.7335. Contoh percobaan yang dilakukan oleh Sangodkar:

Bahasa Inggris	Many Bengali poets **have sung songs in praise of this land**
Phrase Reordering	Many Bengali poets **this land of praise in songs sung have**
Bahasa India	Kai kaviyon ne is mahaan bhoomi ki prashansa ke geet gaaye hai

Berdasarkan percobaan yang telah dilakukan, telah ditunjukkan bahwa teknik *word reordering* dan *phrase reordering* saat *preprocessing* dapat meningkatkan kualitas hasil penerjemahan dari *statistical machine translation* pada beberapa bahasa. Secara umum *preprocessing* ini terdiri dari dua tahap. Tahap pertama adalah membagi kalimat yang akan diterjemahkan menjadi barisan kata. Tahap kedua adalah melakukan *word reordering* atau *phrase reordering* pada kalimat bahasa asal (*source language*) berdasarkan struktur kalimat bahasa tujuan (*target language*) (Zwarts & Dras, 2007).

2.5 POS *Tagging*

POS tagging atau *Part Of Speech tagging* adalah proses penandaan kata – kata dalam suatu teks atau korpus sesuai dengan kelas katanya. Yang termasuk ke dalam kelas kata adalah kata sifat, kata benda, kata kerja, kata keterangan, kata depan, kata kepunyaan, kata penghubung, kata sandang. Suatu kata bisa diklasifikasikan ke dalam kelas kata yang berbeda. Sebagai contoh: kata ”dogs” dalam Bahasa Inggris biasanya diklasifikasikan menjadi *plural noun* (kata benda jamak). Akan tetapi dalam konteks yang berbeda, kata ini bisa diklasifikasikan menjadi *verb* (kata kerja) misalnya pada kalimat ”*Sailor dogs the hatch”*. Solusi untuk mengatasi masalah ambiguitas ini adalah dengan melihat kelas kata dari kata sebelumnya. (Jurafsky, & Martin, 2000)

Berikut ini adalah contoh hasil *POS tagging* pada dua kalimat berbeda yang menggunakan kata “dogs”:

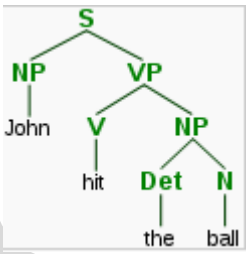
1.	I/PRP have/VBP three/CD dogs/NNS
2.	Sailor/NNP dogs/VBZ the/DT hatch/NN

Dari contoh di atas, secara statistik, jika diketahui jenis *tag* pada kata sebelum kata ”dogs” adalah /CD, maka peluang kata ”dogs” diberi *tag* /NNS lebih besar daripada peluang kata ”dogs” diberi *tag* /VBZ. Oleh karena itu pada contoh pertama kata ”dogs” diberi *tag* /NNS. Apabila diketahui jenis *tag* pada kata sebelum kata ”dogs” adalah /NNP, maka peluang kata ”dogs” diberi *tag* /VBZ lebih besar daripada peluang kata ”dogs” diberi *tag*/ NNS. Oleh karena itu pada contoh kedua kata ”dogs” diberi *tag* /VBZ.

2.6 Parsing

Parser adalah sebuah alat yang memeriksa syntax dari kalimat dan membangun struktur dari kalimat tersebut. Struktur tersebut bisa berupa *parse tree*, *abstract syntax tree*, atau struktur hierarki lainnya. *Parse Tree* merepresentasikan struktur *syntax* dari kalimat berdasarkan sebuah *grammar*

tertentu. Dalam sebuah *parse tree*, *interior nodes* ditandai dengan simbol non terminal dari *grammar*, dan *leaf nodes* ditandai dengan simbol terminal dari grammar. Berikut ini adalah salah satu contoh *parse tree* pada kalimat *John hit the ball*.



Gambar 2.4 *Parse Tree*

Struktur *parse tree* pada kalimat ”John hit the ball*”* dimulai dari S dan berakhir pada setiap leaf nodes (John, hit, the, ball). Penjelasan lebih detail mengenai struktur *parse tree* pada kalimat ”John hit the ball” adalah sebagai berikut:

- S (Sentence) menandakan kalimat, yang merupakan root atau struktur paling atas dari *parse tree*.
- NP (Noun Phrase) menandakan frase kata benda. Frase kata benda yang pertama yang terletak paling kiri adalah kata ”John”, yang menandakan subjek dari kalimat. Frase kata benda yang kedua adalah kata ”the ball”, yang menandakan objek dari kalimat.
- VP (Verb Phrase) menandakan frase kata kerja, yang merupakan predikat dari kalimat.
- V (Verb) menandakan kata kerja. Pada contoh kalimat di atas kata kerja ”hit” merupakan kata kerja transitif.
- Det (Determiner) menandakan kata sandang atau artikula. Pada contoh kalimat di atas ”the” merupakan kata sandang.
- N (Noun) menandakan kata benda. Pada contoh kalimat di atas ”ball” merupakan kata sandang.

Pada *parse tree*, setiap node dapat berupa *root node, branch node*, atau *leaf node*. Pada contoh di atas S adalah *root node*, NP dan VP adalah *branch node*, dan ”John”, ”hit”, ”the”, ”ball” adalah *leaf nodes*. Sebuah node juga dapat dipandang sebagai *parent node* dan *child node*. S adalah *parent node* dari NP dan V. “Hit” adalah *child node* dari V. (Jurafsky, & Martin, 2000)