

Penggunaan *Hidden Markov Model* untuk Kompresi Kalimat

TESIS

**Karya tulis sebagai salah satu syarat
Untuk memperoleh gelar Magister dari
Institut Teknologi Bandung**

Oleh

YUDI WIBISONO

NIM: 23505023

Program Studi Informatika



INSTITUT TEKNOLOGI BANDUNG

2008

ABSTRAK

Penggunaan *Hidden Markov Model* untuk Kompresi Kalimat

Oleh

Yudi Wibisono

NIM: 23505023

Masalah utama dalam penggunaan *handheld device* adalah ukuran layar yang kecil sehingga mempersulit pengguna untuk mencari dan memperoleh informasi tekstual. Penggunaan peringkasan dokumen dapat membantu memecahkan masalah ini, tetapi kalimat yang dihasilkan masih terlalu panjang. Kompresi kalimat dapat diaplikasikan untuk mengurangi panjang kalimat tanpa menghilangkan informasi yang penting.

Kompresi kalimat pada tesis ini menggunakan Hidden Markov Model (HMM) yang diadaptasi dari model *statistical translation* dan HMM-Hedge. Algoritma Viterbi digunakan untuk mencari susunan kata yang paling optimal.

Eksperimen dilakukan untuk mengkaji pengaruh penambahan tag simbol numerik dan tag entitas pada *preprocessing*, bobot probabilitas pada model HMM, dan *bigram smoothing*. Selain itu kinerja metode HMM ini dibandingkan dengan metode Knight-Marcu Noisy Channel. Eksperimen dalam tesis ini menggunakan koleksi kalimat Ziff-Davis yang terdiri atas 1067 pasang kalimat.

Hasil eksperimen memperlihatkan bahwa HMM terbaik dibangun dengan penambahan tag simbol numerik pada *preprocessing*, *Jelinek Mercer smoothing* dengan $\gamma = 0.1$, dan bobot probabilitas = 0.1. Setelah dibandingkan dengan metode Knight-Marcu Noisy Channel, ternyata kinerja HMM masih lebih rendah.

Kata kunci: kompresi kalimat, Hidden Markov Model, algoritma viterbi, *bigram smoothing*

ABSTRACT
Hidden Markov Model
for Sentence Compression

by

Yudi Wibisono

NIM: 23505023

Main problem in using handheld device is the small screen size so that it makes difficult for user to find and get textual information. Document summarization can solve this problem, but sometimes the summary is still too long. Sentence compression can be applied to that summary to reduce sentence length without removing important information.

Sentence compression in this thesis uses Hidden Markov Model (HMM) adapted from statistical translation model and HMM-Hedge. Viterbi algorithm is also used to find optimal word sequence.

There are two experiment goals. First, explore the influences of three parameters to quality of sentence compression. Second, compare our system performance with Knight-Marcu Noisy Channel performance. This experiment used Ziff-Davis corpus, that consists of 1067 pairs of sentences.

Best HMM is constructed by adding numerical tags in preprocessing, Jelinek Mercer smoothing with $\gamma = 0.1$, and probability weight 0.1. This system is still worse than Knight Marcu Noisy Channel.

Keyword: sentence compression, Hidden Markov Model, viterbi algorithm, bigram smoothing

Penggunaan *Hidden Markov Model* untuk Kompresi Kalimat

Oleh

Yudi Wibisono

NIM: 23505023

Program Studi Informatika
Institut Teknologi Bandung

Menyetujui
Tim Pembimbing

Tanggal 26 Juni 2008

Ketua

Ir. Dwi Hendratmo Widyantoro, M.Sc., Ph.D

PEDOMAN PENGGUNAAN TESIS

Tesis S2 yang tidak dipublikasikan terdaftar dan tersedia di Perpustakaan Institut Teknologi Bandung, dan terbuka untuk umum dengan ketentuan bahwa hak cipta ada pada pengarang dengan mengikuti aturan HaKI yang berlaku di Institut Teknologi Bandung. Referensi kepustakaan diperkenankan dicatat, tetapi pengutipan atau peringkasan hanya dapat dilakukan seizin pengarang dan harus disertai dengan kebiasaan ilmiah untuk menyebutkan sumbernya.

Memperbanyak atau menerbitkan sebagian atau seluruh tesis haruslah seizin Direktur Program Pascasarjana Institut Teknologi Bandung.

KATA PENGANTAR

Puji dan syukur penulis panjatkan kepada Allah SWT sehingga penulisan tesis ini dapat diselesaikan dengan baik. Penulis sangat berterima kasih kepada:

1. Bapak Ir. Dwi Hendratmo Widyantoro, M.Sc., Ph.D, sebagai pembimbing tesis atas segala saran, bimbingan, dan petunjuknya selama penelitian berlangsung dan selama penulisan tesis ini dilakukan.
2. Bapak Dr. Ir. Benhard Sitohang sebagai wali angkatan 2005 S2-IF.
3. Ibu Ir. Sri Purwanti, M.Sc (alm) sebagai penguji presentasi proposal dan seminar.
4. Ibu Dr. Eng. Ayu Purwarianti, ST, MT sebagai penguji pra sidang dan sidang.
5. Ibu Harlili, M.Sc sebagai penguji sidang.
6. Para staf pengajar Program Studi Teknik Informatika ITB terutama Ketua dan Sekretaris Program Studi Teknik Informatika.
7. Para staf tata usaha akademik S2-IF.
8. Masayu Leylia Khodra, MT sebagai istri dan teman diskusi yang banyak memberikan perhatian, saran, dan komentar.
9. Teman-teman angkatan 2005 S2-IF

Bandung, 26 Juni 2008

Penulis

DAFTAR ISI

ABSTRAK.....	ii
ABSTRACT.....	iii
PEDOMAN PENGGUNAAN TESIS.....	v
KATA PENGANTAR.....	vi
DAFTAR ISI.....	vii
DAFTAR GAMBAR.....	viii
DAFTAR TABEL.....	ix
Bab I Pendahuluan.....	1
I.1 Latar Belakang.....	1
I.2 Rumusan Masalah.....	2
I.3 Ruang Lingkup dan Batasan Masalah.....	3
I.4 Tujuan Tesis.....	3
I.5 Metodologi Penelitian.....	3
I.6 Sistematika Pembahasan.....	4
Bab II Dasar Teori.....	5
II.1 Hidden Markov Model (HMM).....	5
II.2 Decoding dengan Algoritma Viterbi.....	8
II.3 Kompresi Kalimat.....	8
II.3.1 Teknik yang Bergantung ada Bahasa Tertentu.....	9
II.3.2 Teknik yang Tidak Bergantung pada Bahasa.....	11
II.3.3 Evaluasi Kinerja.....	11
II.4 Aplikasi HMM untuk Kompresi Kalimat.....	12
II.4.1 Statistical Translation Model.....	12
II.4.2 HMM-Hedge.....	14
II.5 Bigram Smoothing.....	15
Bab III Analisis dan Rancangan Sistem Kompresi Kalimat.....	18
III.1 Model Probabilitas.....	18
III.2 Hidden Markov Model (HMM).....	20
III.3 Decode Kompresi Kalimat.....	22
III.4 Preprocessing.....	24
III.5 Arsitektur Sistem.....	25
Bab IV Eksperimen.....	27
IV.1 Skenario Eksperimen.....	28
IV.2 Hasil Eksperimen Skenario Pertama.....	28
IV.2.1 Pengaruh Penambahan Tag Simbol Numerik dan Entitas pada <i>Preprocessing</i>	29
IV.2.2 Pengaruh <i>Bigram Smoothing</i>	30
IV.2.3 Pengaruh Bobot α	31
IV.3 Hasil Eksperimen Skenario Kedua.....	32
IV.4 Hasil Eksperimen Skenario Ketiga.....	32
Bab V Kesimpulan dan Saran.....	36
V.1 Kesimpulan.....	36
V.2 Saran.....	36
Daftar Pustaka.....	37

DAFTAR GAMBAR

Gambar I-1 Contoh Pemotongan Ringkasan Artikel pada Google News [GOO08]	1
Gambar I-2 Contoh kompresi kalimat pada koleksi dokumen Ziff-Davis	2
Gambar II-1 Contoh Markov Chain	5
Gambar II-2 Contoh HMM untuk part of speech tagger [JUR06].....	7
Gambar II-3 Contoh pembuangan subordinat [COR99]	10
Gambar II-4 Topologi HMM-Hedge.....	15
Gambar III-1 Topologi Pertama HMM	21
Gambar III-2 Topologi Kedua HMM.....	21
Gambar III-3 Diagram trellis untuk $t=1$	23
Gambar III-4 Diagram trellis untuk $t=2$	23
Gambar III-5 Arsitektur Sistem Kompresi Kalimat.....	26
Gambar IV-1 Contoh pasangan kalimat pada koleksi Ziff Davis	27
Gambar IV-2 Contoh perbandingan hasil HMM dengan Knight-Marcu Noisy Channel .	33
Gambar IV-3 Contoh hasil HMM yang lebih unggul dari Knight-Marcu Noisy Channel	34
Gambar IV-3 Tiga kalimat hasil kompresi HMM dengan skor ROGUE-2 tertinggi.....	35
Gambar IV-4 Tiga kalimat hasil kompresi HMM dengan skor ROGUE-2 terendah	35

DAFTAR TABEL

Tabel III-1 Contoh pemberian tag simbol numerik.....	25
Tabel III-2 Contoh pemberian tag simbol entitas	25
Tabel IV-1 Parameter dan nilai yang diteliti dalam eksperimen	28
Tabel IV-2 Frekuensi tag simbol numerik pada data latihan.....	29
Tabel IV-3 ROUGE-2 untuk preprocessing	30
Tabel IV-4 ROUGE-2 Zue Smoothing.....	30
Tabel IV-5 Nilai ROUGE-2 untuk J-M Smoothing.....	31
Tabel IV-6 ROUGE-2 untuk berbagai nilai α	31
Tabel IV-7 ROUGE-2 untuk kedua topologi.....	32
Tabel IV-8 ROUGE-2 antara HMM dan K-M Noisy Channel.....	33

Bab I Pendahuluan

Bab ini menguraikan latar belakang, rumusan masalah, ruang lingkup dan batasan masalah, tujuan, metodologi, dan sistematika pembahasan.

1.1 Latar Belakang

Beberapa tahun terakhir, akses internet melalui *handheld device* (PDA, telepon selular) semakin umum dilakukan. Masalah utama pada penggunaan *handheld device* adalah ukuran layar yang relatif kecil yaitu sekitar 2 sampai dengan 3.5 inci. Hal ini membuat pengguna lebih sulit mencari dan memperoleh informasi tekstual secara efisien.

Aplikasi peringkasan teks dapat digunakan untuk membantu mengatasi permasalahan ini. Peringkasan teks adalah proses otomatis yang menghasilkan versi dokumen yang lebih kecil 50% atau kurang tetapi tetap berguna bagi pengguna [RAD02]. Teknik yang umum digunakan dalam peringkasan teks adalah mengambil kalimat yang paling penting dari sebuah dokumen.

Berger et. al. [BUY01] memperlihatkan bahwa kombinasi ekstraksi kata kunci dan peringkasan teks memberikan kinerja yang tertinggi untuk penemuan halaman web di PDA. Namun, kalimat hasil peringkasan terkadang masih terlalu panjang untuk ditampilkan di *handheld device* sehingga memenuhi layar. Selain itu pemotongan secara paksa terhadap kalimat dapat ikut menghilangkan bagian penting dari kalimat. Gambar I-1 memperlihatkan contoh pemotongan ringkasan artikel pada *Google News*.



Gambar I-1 Contoh Pemotongan Ringkasan Artikel pada *Google News* [GOO08]

Kompresi kalimat dapat digunakan untuk menyelesaikan permasalahan pemotongan kalimat ini. Kompresi kalimat adalah pemilihan kata yang penting dalam satu kalimat dan menyusun ulang kata tersebut menjadi kalimat yang lebih ringkas, atau sebaliknya, menghilangkan kata yang tidak penting dan menyusun kata sisanya. Contoh kompresi kalimat dapat dilihat pada Gambar I-2.

Kalimat lengkap:	All of our design goals were achieved and the delivered performance matches the speed of the underlying device
Kalimat kompresi:	All design goals were achieved
Kalimat lengkap:	The files are stored in a temporary directory on the VAX disks, where they are converted to VMS Backup format
Kalimat kompresi:	Files are stored in a temporary directory on the VAX disks .

Gambar I-2 Contoh kompresi kalimat pada koleksi dokumen Ziff-Davis

Berdasarkan sudut pandang ketergantungan terhadap bahasa, ada dua pendekatan di dalam teknik kompresi kalimat, yaitu teknik yang bergantung kepada sintaks dan tatabahasa bahasa tertentu, dan teknik yang tidak bergantung kepada bahasa tertentu. Teknik yang tidak bergantung kepada bahasa umumnya berbasis statistik dan dapat digunakan secara *cross-lingual* yaitu kalimat asal dan kalimat yang akan dikompresi menggunakan bahasa yang berbeda.

Beberapa peneliti [NGU04][ZAJ02][DOR03][WIT99][BAN00] menggunakan HMM untuk kompresi kalimat. Teknik ini tidak bergantung kepada bahasa, *cross lingual* dan lebih tahan terhadap *noise* (kalimat yang tidak memenuhi secara sempurna aturan sintaks dan tatabahasa). Namun, penelitian mengenai HMM [ZAJ02][DOR03] belum membahas secara rinci parameter apa saja yang mempengaruhi kualitas kompresi kalimat.

1.2 Rumusan Masalah

Rumusan masalah pada tesis ini adalah bagaimana melakukan kompresi kalimat dengan menggunakan HMM, mengkaji faktor yang mempengaruhi kualitas kompresi, dan membandingkan kinerja HMM dibandingkan teknik kompresi lainnya.

1.3 Ruang Lingkup dan Batasan Masalah

Tesis ini mencakup kajian mengenai kompresi kalimat dengan HMM. Beberapa parameter HMM diujicoba untuk melihat pengaruhnya terhadap kualitas kompresi. Metode HMM kemudian dibandingkan dengan metode Knight-Marcu (K-M) Noisy Channel [KNI00] sebagai salah satu metode terbaik untuk kompresi kalimat.

1.4 Tujuan Tesis

Tujuan dari tesis ini adalah:

1. Meneliti sistem kompresi kalimat dengan HMM.
2. Meneliti parameter apa saja yang mempengaruhi kinerja sistem.
3. Membandingkan kinerja HMM dengan metode K-M Noisy Channel [KNI07].

1.5 Metodologi Penelitian

Tahapan-tahapan yang akan dilalui adalah sebagai berikut :

1. Studi literatur

Pada tahap ini, dilakukan kajian mengenai kompresi kalimat, HMM, aplikasi HMM untuk kompresi kalimat, dan *bigram smoothing*.

2. Analisis dan Perancangan

Pada tahap ini dilakukan analisis dan perancangan sistem kompresi kalimat termasuk parameter-parameter yang perlu dipertimbangkan dalam kompresi kalimat.

3. Implementasi Sistem Kompresi Kalimat

Pada tahap ini dilakukan implementasi berdasarkan hasil analisis dan rancangan pada tahap sebelumnya dengan bahasa pemrograman Java.

4. Eksperimen dan Evaluasi

Pada tahap ini dilakukan eksperimen sesuai tujuan tesis. Kemudian dilakukan evaluasi hasil eksperimen. .

5. Kesimpulan

Dari semua tahapan, ditarik kesimpulan dan saran untuk penelitian selanjutnya.

1.6 Sistematika Pembahasan

Pembahasan laporan tesis ini terdiri dari lima bab dengan perincian sebagaimana yang dijelaskan berikut ini.

Bab I Pendahuluan, menguraikan tentang latar belakang, rumusan masalah, ruang lingkup dan batasan masalah, tujuan, metodologi, dan sistematika pembahasan.

Bab II Dasar Teori, membahas landasan teori mengenai HMM dan kompresi kalimat. Selain itu dibahas penelitian terkait yang berkaitan dengan HMM.

Bab III Analisis, berisi penjelasan dan analisis terhadap sistem HMM yang dikembangkan, meliputi arsitektur dan deskripsi sistem.

Bab IV Eksperimen dan Evaluasi, berisi penjelasan mengenai eksperimen yang dilakukan meliputi tujuan eksperimen, skenario eksperimen, dan hasil eksperimen yang dilengkapi dengan pembahasannya.

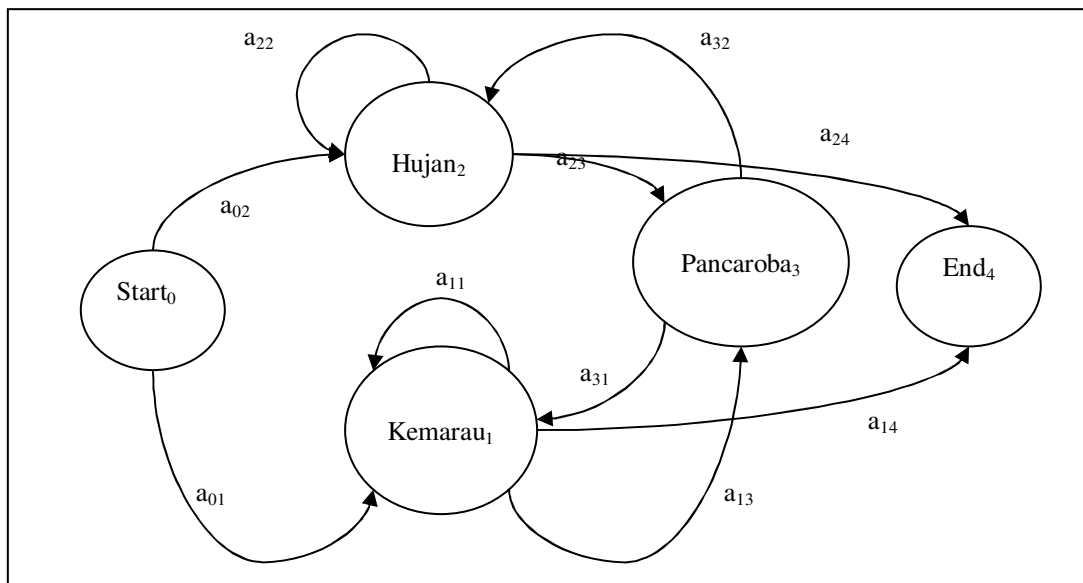
Bab V Kesimpulan dan Saran, berisi kesimpulan yang dapat diambil dari pelaksanaan tesis ini beserta saran pengembangan penelitian ini lebih lanjut.

Bab II Dasar Teori

Bab ini membahas landasan teori mengenai kompresi kalimat. Selain itu dibahas konsep Hidden Markov Model (HMM) dan penelitian-penelitian mengenai kompresi kalimat yang menggunakan HMM.

II.1 Hidden Markov Model (HMM)

Sebelum mendefinisikan HMM, perlu dibahas terlebih dulu mengenai *Markov Chain*. *Markov Chain* merupakan perluasan dari *finite automaton*. *Finite automaton* sendiri adalah kumpulan *state* yang transisi antar *state*-nya dilakukan berdasarkan masukan observasi. Pada Markov Chain, setiap busur antar *state* berisi probabilitas yang mengindikasikan kemungkinan jalur tersebut akan diambil. Jumlah probabilitas semua busur yang keluar dari sebuah simpul adalah satu. Gambar II-1 memperlihatkan contoh Markov Chain yang menggambarkan kondisi cuaca. Pada gambar ini, a_{ij} adalah probabilitas transisi dari *state* i ke *state* j .



Gambar II-1 Contoh Markov Chain

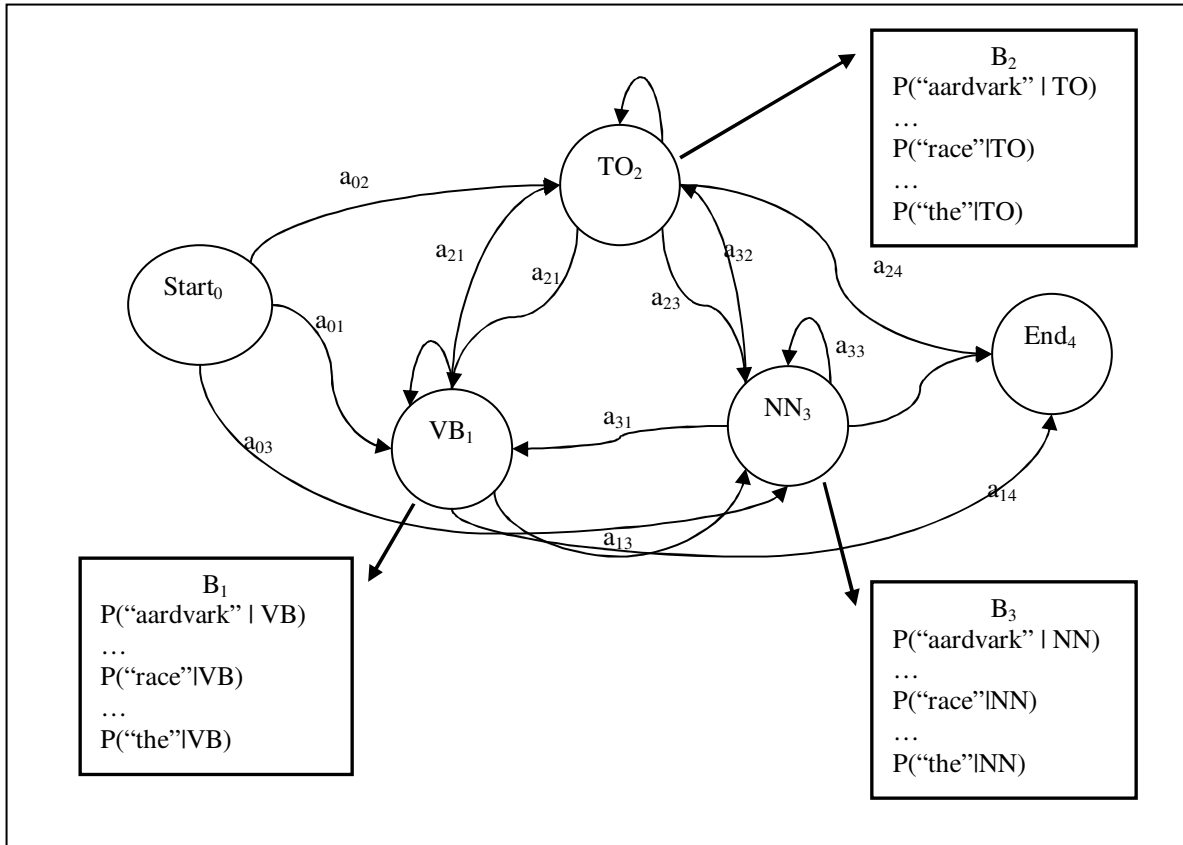
Markov Chain bermanfaat untuk menghitung probabilitas urutan kejadian yang dapat diamati. Masalahnya, terkadang ada urutan kejadian yang ingin diketahui tetapi tidak dapat diamati. Contohnya pada kasus *part-of-speech tagging (POS tagging)*. *POS tagging* adalah pemberian *tag* kepada kata: kata benda, kata kerja, kata sifat dan lain-lain. Pada *POS tagging*, urutan *tag* tidak dapat diamati secara langsung. Pengamatan secara langsung hanya dapat dilakukan terhadap urutan kata. Dari urutan kata tersebut harus dicari urutan *tag* yang paling tepat. Untuk contoh ini, *tag* adalah bagian yang tersembunyi.

Dalam HMM, bagian yang dapat diamati disebut *observed state* sedangkan bagian yang tersembunyi disebut *hidden state*. HMM memungkinkan pemodelan sistem yang mengandung *observed state* dan *hidden state* yang saling terkait. Pada kasus *POS tagging*, *observed state* adalah urutan kata sedangkan *hidden state* adalah urutan *tag*. Selain itu, HMM dapat digunakan dalam pengenalan suara, *parsing/chunking*, ekstraksi informasi, dan peringkasan teks [JUR06].

HMM terdiri atas lima komponen, yaitu:

1. Himpunan *observed state*: $O = o_1, o_2, \dots, o_N$.
2. Himpunan *hidden state*: $Q = q_1, q_2, \dots, q_N$
3. Probabilitas transisi: $A = a_{01}, a_{02}, \dots, a_{n1} \dots a_{nm}$; a_{ij} adalah probabilitas untuk pindah dari *state* i ke *state* j .
4. Probabilitas emisi atau *observation likelihood*: $B = b_i(o_t)$, merupakan probabilitas observasi o_t dibangkitkan oleh *state* i .
5. *State* awal dan akhir: q_0, q_{end} , yang tidak terkait dengan observasi.

Gambar II-2 memperlihatkan contoh topologi HMM untuk *POS tagger* untuk bahasa Inggris [JUR06].



Gambar II-2 Contoh HMM untuk part of speech tagger [JUR06]

Pada Gambar II-2, kata yang dicari tag-nya adalah “aardvark”, “race”, dan “the”. Ketiga kata tersebut menjadi *observed state*. Sedangkan *hidden state* adalah “TO” (*to infinitive*), “VB” (*verb base form*), dan “NN” (*mass noun*). B_i himpunan semua probabilitas emisi untuk *hidden state* i , sedangkan a_{ij} adalah probabilitas transisi dari *state* i ke *state* j .

Menurut Rabiner [RAB89], salah satu masalah yang dapat diselesaikan oleh HMM adalah masalah optimasi urutan *hidden state* berdasarkan urutan kejadian yang dapat diamati. Urutan *hidden state* optimal adalah urutan yang paling tepat yang “menjelaskan” kejadian yang dapat diamati. Masalah ini disebut juga masalah *decoding*.

II.2 Decoding dengan Algoritma Viterbi

Decoding adalah proses mencari urutan *hidden state* yang paling optimal berdasarkan kejadian yang dapat diamati. Algoritma Viterbi dapat digunakan untuk menyelesaikan masalah ini.

Algoritma Viterbi adalah algoritma dinamik untuk mencari jalur *hidden state* yang paling optimal. Algoritma ini menggunakan *viterbi trellis* yang dihitung secara rekursif. *Viterbi trellis* $v_t(j)$ adalah probabilitas HMM di *state* j setelah melewati t observasi dan melewati *most likely sequence* $q_1..q_{t-1}$ [JUR06]. Untuk setiap *state* q_j pada waktu t , nilai $v_t(j)$ dihitung sebagai berikut:

$$v_t(j) = \max_{1 \leq i \leq N-1} v_{t-1}(i) a_{ij} b_j(O_t) \quad \text{II-1}$$

dengan

$v_{t-1}(i)$ adalah *viterbi trellis* pada $t-1$

a_{ij} adalah probabilitas transisi dari *state* q_i ke *state* q_j

$b_j(O_t)$ adalah probabilitas emisi untuk *observation state* o_t pada *state* j

Pseudocode Algoritma Viterbi adalah sebagai berikut [JUR06]:

```
function Viterbi (observasi dengan panjang T, state-graph) return best-path
  num-states  $\leftarrow$  NUMOFSTATES(state-graph)
  Buat matrix probabilitas path viterbi[num-states+2, T+2]
  viterbi[0,0]  $\leftarrow$  1.0
  for each time step t from 1 to T do
    for each state s from 1 to num_states do
      viterbi [s,t]  $\leftarrow$  max(viterbi [s',t-1] * as's * bs(Ot))
        1<=s'<=num-states

      backpointer[s,t]  $\leftarrow$  argmax(viterbi [s',t-1] * as's )
        1<=s'<=num-states

  Backtrace dari probabilitas tertinggi di final state (kolom tertinggi) viterbi[] dan return path
end function
```

II.3 Kompresi Kalimat

Kompresi kalimat adalah pemilihan kata atau frasa yang penting dalam satu kalimat dan menyusun ulang kata tersebut menjadi kalimat yang lebih ringkas. Sebaliknya, kompresi

kalimat juga dapat dipandang sebagai proses pembuangan kata atau frasa yang tidak penting.

Kompresi kalimat erat kaitannya dengan peringkasan dokumen, keduanya memiliki tujuan yang sama yaitu menghasilkan teks yang lebih pendek tanpa kehilangan informasi penting. Perbedaannya terletak pada masukan yang diberikan. Peringkasan dokumen menerima masukan berupa teks dokumen sedangkan kompresi kalimat berupa kalimat.

Penelitian mengenai kompresi kalimat telah dilakukan sebelumnya. Berdasarkan faktor ketergantungan terhadap bahasa, teknik kompresi kalimat dapat dibagi menjadi dua bagian, yaitu teknik yang bergantung kepada bahasa tertentu dan teknik yang tidak bergantung kepada bahasa.

II.3.1 Teknik yang Bergantung ada Bahasa Tertentu

Teknik ini memanfaatkan struktur sintaks dan tatabahasa dari kalimat di dalam proses kompresi. Beberapa peneliti [KNI00] [JING00] mengkombinasikan elemen bahasa dengan model statistik, sedangkan peneliti yang lain menggunakan pendekatan berbasis *rule* [COR99].

Knight et al. [KNI00] memanfaatkan *parse tree* untuk mengkompresi kalimat. Contoh *parse tree* dari kalimat “John saw Mary” adalah sebagai berikut:

$$S = \begin{array}{l} (NP \text{ John}) \\ (VP \text{ (VB saw)} \\ (NP \text{ Mary})) \end{array}$$

Pencarian model probabilitas kalimat panjang jika diketahui kalimat pendek (*expansion probability*) dilakukan dengan cara menghitung *join event* antara pasangan *parse tree* kalimat panjang dan pendek. Misalnya $S \rightarrow NP VP PP$ di kalimat panjang dan $S \rightarrow NP VP$ di kalimat pendek dihitung menjadi satu *join event*.

Kemudian dibuat *parse forest* yang berisi semua kemungkinan kompresi yang secara tatabahasa sesuai dengan kumpulan dokumen *Penn Treebank*. *Tree extractor* kemudian digunakan untuk mengambil *parse tree* dengan *expansion probability* tertinggi.

Corston-Oliver [COR99] melakukan analisis linguistik dan menghilangkan kata yang muncul di *subordinat clauses*. Subordinat tersebut adalah *abbreviated clause*, *complement clause*, *adverbial clause*, *infinitival clause*, *relative clause* dan *present participial clause*. Gambar II-3 memperlihatkan contoh setiap *subordinat* yang dapat dibuang.

Abbreviated Clause:

Until further indicated, lunch will be served at 1 p.m.

Complement Clause:

I told the telemarketer that you weren't home

Adverbial Clause:

After John went home, he ate dinner.

Infinitival Clause:

Jon decided to go home

Relative Clause:

I saw the man, who was wearing a green hat

Present Participial Clause

Napoleon attacked the fleet, completely destroying it.

Gambar II-3 Contoh pembuangan subordinat [COR99]

Jing [JING00] menggunakan *syntactic knowledge*, *context information* dan statistik untuk menentukan pembuangan frasa pada kalimat. Tahapan dari sistem ini adalah *syntactic parsing* untuk membuat *parse tree*, *grammar checking* untuk menentukan frasa yang tidak boleh dibuang, *context information* untuk melihat frasa yang tidak berhubungan dengan topik kalimat, dan *corpus evidence* untuk menghitung *threshold* probabilitas apakah sebuah *subtree* dibuang.

Tahap akhir adalah pencarian dan pembuangan *subtree* yang memenuhi kriteria sebagai berikut: tidak wajib secara tatabahasa, tidak terkait dengan topik utama, dan memiliki probabilitas yang berada di atas batas *threshold*.

II.3.2 Teknik yang Tidak Bergantung pada Bahasa

Teknik ini menggunakan kumpulan dokumen berupa pasangan kalimat asal dan kalimat terkompresi untuk melatih model statistik. Setelah model dihasilkan, model ini digunakan untuk mengkompresi kalimat. Elemen seperti struktur sintaks dan tatabahasa tidak digunakan di dalam proses kompresi sehingga kelemahan utama dari teknik ini adalah kalimat yang dihasilkan dapat menyalahi aturan tatabahasa. Beberapa peneliti [NGU04][ZAJ02][DOR03][WIT99][BAN00] menggunakan HMM untuk melakukan kompresi kalimat.

Nguyen et al. [NGU04] menggunakan *template based technique* untuk mengkompresi kalimat. Teknik ini mencari pola hubungan antara kalimat panjang dan kalimat ringkas tanpa melalui proses parsing.

Zajic et al [ZAJ02] menggunakan HMM (yang disebut HMM-Hedge) untuk membuat judul berita bahasa Inggris. Sedangkan Dorr et. al [DOR03] menggunakan HMM Hedge untuk *cross language task* yaitu membuat judul bahasa Inggris dari berita berbahasa Hindi.

Witbrock et al. [WIT99], Banko et al.[BAN00] menggunakan prinsip *statistical translation*, dengan first order Markov dan viterbi untuk *decoding*.

HMM-Hedge dan teknik *statistical translation* akan dibahas pada bab berikutnya.

II.3.3 Evaluasi Kinerja

Kinerja sistem kompresi kalimat dapat dievaluasi dengan ROGUE-2 [LIN03] [LIN04]. ROGUE (*Recall-Oriented Understudy for Gisting Evaluation*) adalah besaran untuk menentukan secara otomatis kualitas sebuah ringkasan dengan membandingkannya dengan ringkasan ideal buatan dari manusia. ROGUE telah digunakan dalam *Document Understanding Conference* (DUC) sejak tahun 2004.

ROGUE-N mengukur jumlah *overlapping unit* n-gram antara ringkasan yang dihasilkan oleh mesin dengan ringkasan referensi yang biasanya dibuat oleh manusia. Dapat

dikatakan ROGUE-N mengukur nilai recall n-gram antara kalimat ringkasan dengan kalimat referensi. ROGUE-N didefinisikan sebagai berikut:

$$ROUGE - N = \frac{\sum_{S \in \{ref\ summaries\}} \sum_{gram_n \in S} count_{match}(gram_n)}{\sum_{S \in \{ref\ summaries\}} \sum_{gram_n \in S} count(gram_n)} \quad \text{II-2}$$

Jika hanya terdapat satu ringkasan ideal sebagai referensi dan n adalah dua (bigram) maka persamaan II-2 dapat ditulis sebagai berikut:

$$ROUGE - 2 = \frac{\sum count_{match}(gram_2)}{\sum count(gram_2)} \quad \text{II-3}$$

II.4 Aplikasi HMM untuk Kompresi Kalimat

Dalam subbab ini dibahas secara lebih rinci dua penelitian yang mengaplikasikan model statistik yang menjadi dasar penggunaan HMM pada tesis ini, yaitu *statistical translation* dan HMM-Hedge.

II.4.1 Statistical Translation Model

Statistical translation adalah model matematika yang secara statistik memodelkan proses translasi oleh manusia [YAM01]. Penelitian Witbrock et al. [WIT99] dan Banko et al. [BAN00] menggunakan *statistical translation* untuk peringkasan dokumen karena proses peringkasan dapat dianggap sebagai proses “menerjemahkan” dari satu “bahasa” yang mengandung banyak kata menjadi “bahasa” ringkasan.

Penelitian ini menggunakan dua model, yaitu model translasi untuk pemilihan kata penting dan model bahasa untuk pemilihan urutan kata. Kedua model tersebut dibentuk dengan proses pembelajaran. Selanjutnya proses pencarian ringkasan yang memaksimalkan probabilitas kedua model tersebut dilakukan dengan *viterbi beam search*

Berikut adalah rincian setiap model

1. Model Translasi

Model translasi merupakan model relasi antara kemunculan suatu fitur di dalam dokumen dan kemunculan fitur tersebut pada ringkasan. Model translasi berfungsi mengidentifikasi kata yang penting. Model ini dihitung dengan persamaan:

$$P(w_i \in H | w_i \in D) = \frac{P(w_i \in D | w_i \in H) \cdot P(w_i \in H)}{P(w_i \in D)} \quad \text{II-4}$$

H dan D merepresentasikan kumpulan kata dari ringkasan dan dokumen. $P(w_i \in H | w_i \in D)$ merupakan *conditional probability* suatu kata akan muncul di dalam ringkasan jika diketahui probabilitas kata itu ada di dalam dokumen.

2. Model Bahasa

Model bahasa adalah model untuk menentukan urutan kata. Probabilitas urutan kata dihipotesiskan dengan probabilitas suatu kata jika diketahui satu kata di sebelah kirinya atau lebih dikenal sebagai model bigram.

Model translasi dan model bahasa digunakan secara simultan untuk menghasilkan bobot total dari semua kandidat ringkasan. Untuk menyederhanakan model, diasumsikan setiap kata pada ringkasan independen satu sama lain. Keseluruhan kandidat ringkasan H yang terdiri dari kata (w_1, w_2, \dots, w_n) dapat dihitung sebagai hasil perkalian dari probabilitas kata yang dipilih untuk ringkasan, probabilitas bigram, dan probabilitas panjang ringkasan sepanjang n .

$$P(w_1, \dots, w_n) = \prod_{i=1}^n P(w_i \in H | w_i \in D) \cdot \prod_{i=2}^n P(w_i | w_{i-1}) \cdot P(\text{len}(H) = n) \quad \text{II-5}$$

Nilai probabilitas untuk kedua model ini umumnya sangat kecil sehingga dapat mengakibatkan *underflow*. Oleh karena itu nilai probabilitas disimpan dalam bentuk nilai log.

Persamaan II-5 dapat ditulis sebagai *log probability* dengan α, β, γ merupakan parameter sistem rangkuman. Ketiga parameter ini merupakan bobot untuk probabilitas pemilihan kata, probabilitas panjang ringkasan, dan probabilitas urutan kata.

$$P(w_1, \dots, w_n) = \arg \max_H (\alpha \cdot \sum_{i=1}^n \log(P(w_i \in H \mid w_i \in D)) \\ + \gamma \cdot \sum_{i=2}^n \log(P(w_i \mid w_{i-1}))) \\ + \beta \cdot \log(P(\text{len}(H) = n))$$

II-6

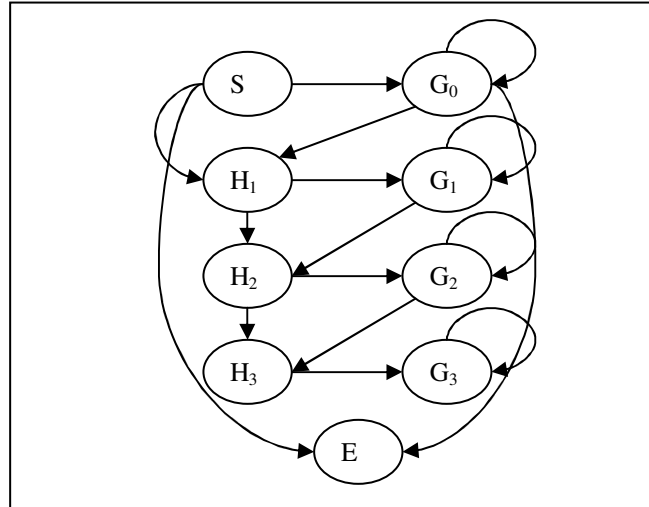
Untuk membangkitkan ringkasan, perlu ditemukan urutan kata yang memaksimalkan probabilitas berdasarkan persamaan II-6. Untuk itu digunakan *viterbi beam search* untuk menemukan urutan kata yang optimal.

Witbrock tidak menjelaskan topologi HMM yang digunakan, tetapi dari dua model yang digunakan dapat dilihat bahwa model translasi dapat digunakan sebagai probabilitas emisi dan model bahasa menjadi probabilitas transisi pada HMM.

II.4.2 HMM-Hedge

HMM-Hedge [ZAJ02] [DOR03] adalah sebuah sistem berbasis Hidden Markov Model yang digunakan untuk menghasilkan judul dari sebuah berita. Gambar II-4 memperlihatkan topologi HMM-Hedge. S dan E adalah *start* dan *end state*, H adalah kata pada judul berita, dan G adalah kata yang akan dibuang. *State H* hanya dapat mengeluarkan emisi satu kata tertentu sedangkan *state G* dapat menghapus kata apapun.

Contoh, jika kalimat masukan adalah “After months of debating following the Sept.11 terrorist hijacking the Transportation Department decided that airline pilots will not allowed to have guns in the cockpits.”, maka urutan pemrosesan adalah sebagai berikut: mulai dari *state S*, mengeluarkan simbol S , lalu pindah ke *state G₀*. HMM akan tetap di *state G₀* dan mengeluarkan kata “after”, “month” ... “airline”. Kemudian pindah ke H_{pilot} dan mengeluarkan kata “pilot”. Dilanjutkan G_{will} dan H_{not} dan seterusnya sampai $H_{cockpit}$ yang dilanjutkan ke E . Hasil dari emisi H berupa: “pilots” “not” “allowed” “to” “have” “guns” “in” “cockpits”.



Gambar II-4 Topologi HMM-Hedge

Untuk probabilitas emisi, HMM-Hedge menggunakan nilai satu untuk *state H* dan $p(w)$ untuk *state G*, sedangkan probabilitas bigram digunakan sebagai probabilitas transisi. HMM-Hedge menggunakan empat *decoding parameter*, yaitu:

1. Penalti ukuran untuk mengatur agar jumlah kata pada ringkasan antara 5 sampai dengan 15.
2. Penalti posisi yang memberikan keuntungan kata yang muncul di awal dokumen.
3. *String penalty* digunakan untuk mengatur keterikatan antara kata yang berurutan pada kalimat asal. Semakin tinggi *string penalty* semakin sulit perpindahan dari *state H* ke *state G*, sehingga *state H* akan cenderung pindah ke *state H* lagi.
4. Penalti-jarak digunakan untuk mencegah jarak yang terlalu besar antara kumpulan kata. *Penalty* ini mencegah sistem berada terlalu lama di *state G*.

II.5 Bigram Smoothing

Baik *statistical translation model* maupun HMM-Hedge menggunakan probabilitas bigram sebagai probabilitas transisi. Jika terdapat pasangan kata C_{i-1} C_i , dan $count(x)$

adalah jumlah kemunculan x di dokumen pelatihan, maka probabilitas bigram pasangan tersebut $P(C_i|C_{i-1})$ dapat dihipotesis dengan persamaan berikut ini.

$$P(C_i | C_{i-1}) = \frac{\text{count}(C_{i-1}C_i)}{\text{count}(C_i)} \quad \text{II-7}$$

Masalah dalam penghitungan probabilitas bigram adalah jika terdapat data pasangan bigram yang tidak ada di dokumen latihan, yaitu $\text{count}(c_i|c_{i-1}) = 0$ sehingga nilai probabilitas bigram menjadi nol. Untuk mengurangi masalah ini, dilakukan *smoothing* dalam perhitungan bigram. [CHE96].

Ada dua jenis *smoothing* bigram yang akan diujicoba dalam tesis ini: Jelinek-Mercer [JEL80] dan Zue[ZUE92]. Kedua jenis *smoothing* ini menambahkan nilai unigram ke dalam perhitungan probabilitas bigram, sehingga walaupun pasangan bigram tidak ditemukan di dalam data latihan, nilai probabilitasnya akan dihipotesis oleh probabilitas unigram.

Jelinek-Mercer *smoothing* mengganti persamaan II-7 menjadi persamaan II-8.

$$P(C_i | C_{i-1}) = \lambda \hat{p}(C_i | C_{i-1}) + (1 - \lambda) \hat{p}(C_i) \quad \text{II-8}$$

dimana λ adalah konstanta yang bernilai antara 0 sampai dengan 1,

$$\hat{p}(C_i | C_{i-1}) = \frac{\text{count}(C_{i-1}C_i)}{\text{count}(C_i)} \quad \text{II-9}$$

$$\hat{p}(C_i) = \frac{\text{count}(C_i)}{\text{jumlah_kata}} \quad \text{II-10}$$

Zue *smoothing* merupakan modifikasi *smoothing* Jelinek-Mercer dan didefinisikan sebagai berikut:

$$P(C_i | C_{i-1}) = \lambda(C_{i-1}) \hat{p}(C_i | C_{i-1}) + (1 - \lambda(C_{i-1})) \hat{p}(C_i) \quad \text{II-11}$$

$$\hat{p}(C_i | C_{i-1}) = \frac{\text{count}(C_{i-1}C_i)}{\text{count}(C_i)} \quad \text{II-12}$$

$$\hat{p}(C_i) = \frac{\text{count}(C_i)}{\text{count}(\text{allwords})} \quad \text{II-13}$$

$$\lambda(C_{i-1}) = \frac{\text{count}(C_{i-1})}{\text{count}(C_{i-1}) + K} \quad \text{II-14}$$

dengan K adalah konstanta yang merupakan bilangan positif integer.

Sebagai contoh perhitungan bigram pasangan kata "mesin pencari", diketahui data berikut ini dari dokumen pelatihan:

1. total semua kata berjumlah 100
2. kemunculan kata "mesin" adalah 10
3. kemunculan kata "pencari" adalah 2,
4. kemunculan kata "mesin pencari" adalah 0

maka hasil perhitungan probabilitas bigram adalah:

1. probabilitas bigram tanpa smoothing (persamaan II-8II-7) = $0/2 = 0$
2. probabilitas bigram *Jelinek-Mercer Smoothing* (persamaan II-8) = $0.5*0+(1-0.5)*2/100=0.01$, dengan $\lambda=0.5$ dan $\hat{p}(C_i | C_{i-1})=0/2=0$.
3. probabilitas bigram *Zue Smoothing* (persamaan II-8 II-11) = $0.1*0+(1-0.1)*2/100=0.18$, dengan $K=90$, $\lambda(\text{mesin})=10/(10+90)=0.1$, dan $\hat{p}(C_i | C_{i-1})=0/2=0$.

Bab III Analisis dan Rancangan Sistem Kompresi Kalimat

Bab ini berisi penjelasan dan analisis terhadap sistem kompresi kalimat yang dikembangkan di dalam tesis ini.

Penelitian ini menggunakan pendekatan *statistical translation* yang digunakan oleh Witbrock et al. [WIT99] dan Banko et al. [BAN00]. Sedangkan Hidden Markov Model yang digunakan diadaptasi dari HMM-Hedge yang digunakan oleh Zajic et al [ZAJ02] dan Dorr et al. [DOR04]. Adaptasi dilakukan pada topologi HMM, probabilitas emisi, probabilitas transisi, dan *preprocessing*.

III.1 Model Probabilitas

Secara formal, kompresi kalimat merupakan pencarian kompresi yang memaksimalkan $P(C|S)$.

$$\hat{C} = \arg \max_C P(C|S) \quad \text{III-1}$$

dengan,

S adalah kalimat asal, terdiri atas urutan kata $S_1, S_2, S_3, \dots, S_N$

C adalah kalimat hasil kompresi kalimat, terdiri atas urutan kata $C_1, C_2, C_3, \dots, C_N$. Kata C_i dapat berupa S_i atau S_i yang dihapus ($\#S_i\#$).

\hat{C} adalah hasil kompresi kalimat yang paling optimal.

Misalnya jika S adalah “finally another advantage of broadband is distance” maka salah satu kandidat C yang terbaik adalah “#finally# another advantage #of# #broadband# is distance”. Kata yang ditandai dengan #, adalah kata yang dihapus sehingga C dapat dibaca “another advantage is distance”.

Jika menggunakan teorema Bayes, persamaan III-1 dapat ditulis kembali sebagai berikut:

$$\hat{C} = \arg \max_c \frac{P(S | C)P(C)}{P(S)} \quad \text{III-2}$$

Karena $P(S)$ bernilai sama untuk setiap kombinasi C maka persamaan III-2 dapat ditulis kembali menjadi:

$$\hat{C} = \arg \max_c P(S | C)P(C) \quad \text{III-3}$$

Karena $P(S|C)$ dan $P(C)$ masih sulit dihitung, maka digunakan dua asumsi. Asumsi pertama adalah probabilitas kemunculan suatu kata di kalimat asal hanya bergantung kepada pasangan kata ini di kalimat yang terkompresi. Oleh karena itu $P(S|C)$ dapat dihampiri dengan:

$$P(S | C) \approx \prod_{i=1}^n P(S_i | C_i) \quad \text{III-4}$$

dengan S_i adalah kata ke- i di kalimat asal, C_i adalah kata ke- i di kalimat terkompresi $P(S_i | C_i)$ adalah probabilitas kemunculan suatu kata S_i di kalimat asli jika diketahui kata C_i muncul di kompresi dan dihitung dengan cara sebagai berikut:

$$P(S_i | C_i) = \frac{\text{count}(C_i, S_i)}{\text{count_all}(C_i)} \quad \text{III-5}$$

Asumsi kedua adalah kata pada kalimat terkompresi hanya bergantung kepada satu kata sebelum kata tersebut, sehingga $P(C)$ dihitung sebagai berikut:

$$P(C) \approx \prod_{i=1}^n P(C_i | C_{i-1}) \quad \text{III-6}$$

Dapat dilihat bahwa $P(C)$ dihitung dengan probabilitas bigram.

Karena nilai probabilitas yang dihasilkan cenderung sangat kecil, maka digunakan *log probability*. Jika persamaan III-4 dan III-6 disubstitusikan ke persamaan III-3 dan ditambahkan bobot probabilitas α , maka persamaannya menjadi:

$$\hat{C} = \arg \max_H ((1 - \alpha) \sum_{i=1}^n \log(P(S_i | C_i)) + \alpha \sum_{i=1}^n P(C_i | C_{i-1})) \quad \text{III-7}$$

Nilai $P(S_i | C_i)$ dan $P(C_i | C_{i-1})$ dihitung dari dokumen latih yang terdiri atas pasangan kalimat asli dan kalimat yang terkompresi.

III.2 Hidden Markov Model (HMM)

Berdasarkan model probabilitas yang menggunakan persamaan III-7, dapat digunakan Hidden Markov Model (HMM) untuk mencari susunan kata yang paling mungkin menjadi kalimat terkompresi. Berikut akan dibahas secara lebih detail setiap komponen HMM yang digunakan dalam task kompresi kalimat.

1. Observed State

Observed state ($S_1, S_2, .. S_N$) pada HMM ini adalah urutan kata kalimat asal yang akan dikompresi. S_1 adalah kata pertama, S_2 kata kedua dan seterusnya.

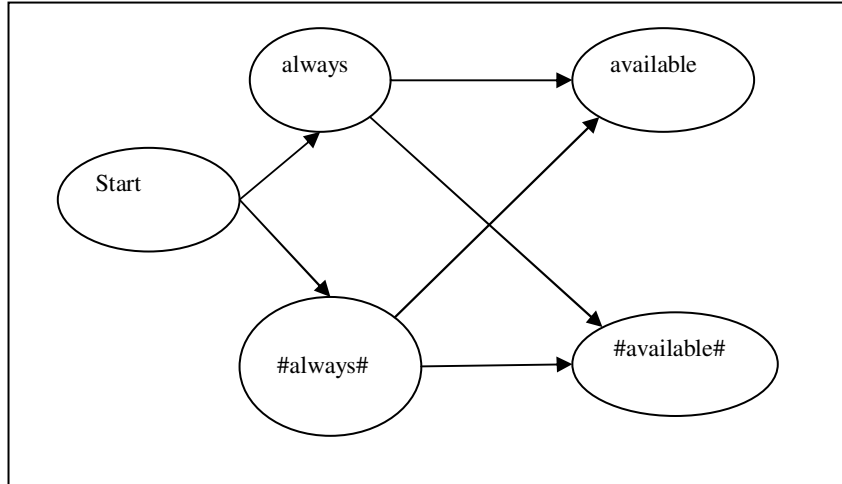
2. Start State

HMM untuk kompresi kalimat ini mempunyai *start-state*. *End-state* tidak digunakan karena proses akan dihentikan setelah semua *observed state* diproses.

3. Hidden state

Untuk sejumlah N kata unik dalam kalimat, terdapat $2N$ *hidden state*. Untuk setiap *observed state* terdapat dua *hidden state*, yaitu satu *hidden state* yang akan menampilkan kata dan satu *hidden state* yang menandakan kata itu dihapus. Sebagai contoh, jika *observed state* adalah "always", "available" maka ada empat *hidden state* yaitu "always", "#always#", "available" dan "#available#". Gambar III-1 memperlihatkan topologi HMM untuk *observed state* tersebut.

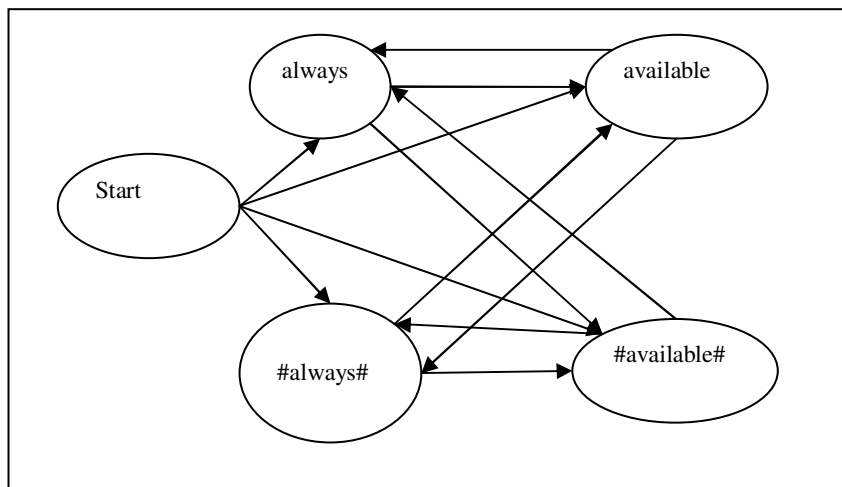
Setiap *hidden state* hanya terhubung dengan *hidden state* kata berikutnya tanpa *self-loop*. Hal ini untuk menjamin urutan kata hasil kompresi akan sesuai dengan urutan kata pada kalimat asli, sehingga mengurangi terbentuknya kalimat yang tidak valid secara tatabahasa.



Gambar III-1 Topologi Pertama HMM

Perbedaan topologi HMM ini dengan topologi HMM-Hedge ditunjukkan oleh Gambar II-4. Perbedaan utama terletak pada representasi kata yang dihapus dan penggunaan *self loop*. Pada HMM ini, setiap kata memiliki pasangan kata yang akan dihapus, sedangkan pada HMM-Hedge, *state G* yang merepresentasikan kata yang dihapus dapat digunakan untuk kata manapun sehingga membutuhkan *self loop*.

Untuk melihat pengaruh topologi terhadap kinerja HMM, dilakukan ujicoba terhadap topologi yang lain. Gambar III-2 memperlihatkan topologi kedua.



Gambar III-2 Topologi Kedua HMM

Pada topologi kedua, setiap *hidden state* saling berhubungan. Dengan topologi ini urutan kalimat yang dihasilkan dapat berbeda dengan kalimat asal.

Kedua topologi ini digunakan karena lebih sederhana dan lebih mudah dijelaskan dengan model probabilitas yang digunakan.

4. *Probabilitas Transisi*

Probabilitas transisi adalah probabilitas perpindahan dari suatu *hidden state* ke *hidden state* lainnya. Probabilitas ini disimpan dalam bentuk log untuk mempermudah perhitungan dan mencegah *underflow*. Pada HMM ini, probabilitas bigram digunakan sebagai probabilitas transisi dan dihitung dari dokumen pelatihan.

5. *Probabilitas Emisi*

Probabilitas emisi adalah probabilitas suatu *observed state* dihasilkan dari sebuah *hidden state*. Dalam HMM ini, probabilitas emisi dihitung dengan $P(S_i | C_i)$ dari dokumen pelatihan. $P(S_i | C_i)$ sendiri dihitung menggunakan persamaan III-5.

6. *Probabilitas Awal*

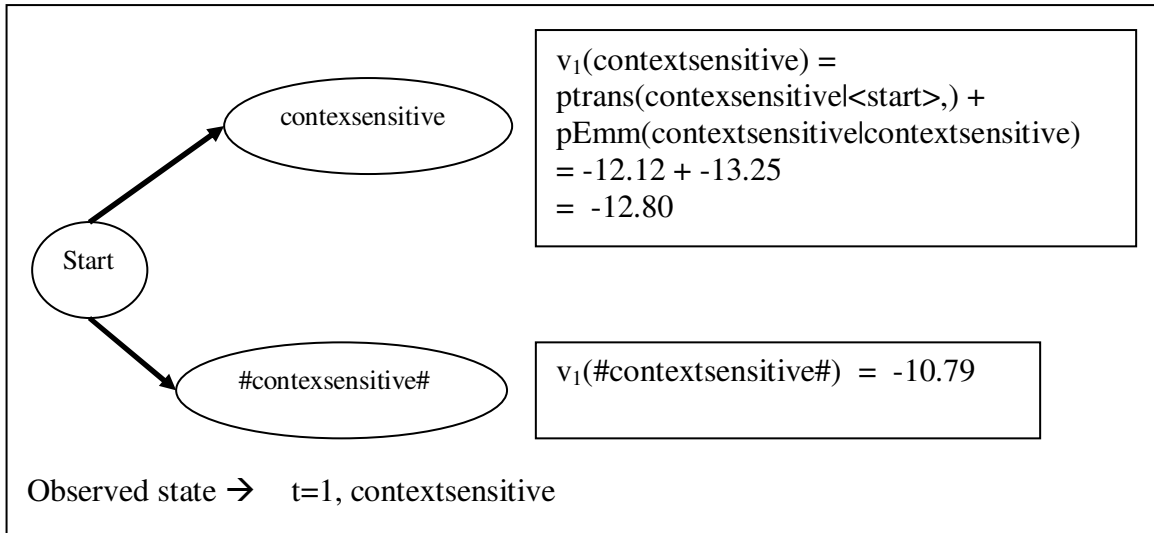
Probabilitas awal Π_i menyatakan probabilitas suatu ringkasan akan dimulai oleh *state* i . Probabilitas awal suatu kata C_i dihitung dengan probabilitas bigram $P(C_i | awal_dokumen)$.

III.3 Decode Kompresi Kalimat

Algoritma Viterbi digunakan untuk mencari urutan *hidden state* yang optimal di dalam HMM. Masukan dari algoritma ini adalah urutan kata kalimat $(S_1, S_2 .. S_n)$ dan outputnya adalah urutan *hidden state* $S = (C_1, C_2 ... C_n)$. Sebelum proses *decode*, dilakukan pembelajaran terhadap dokumen pelatihan untuk mendapatkan probabilitas bigram dan probabilitas emisi $P(S_i | C_i)$.

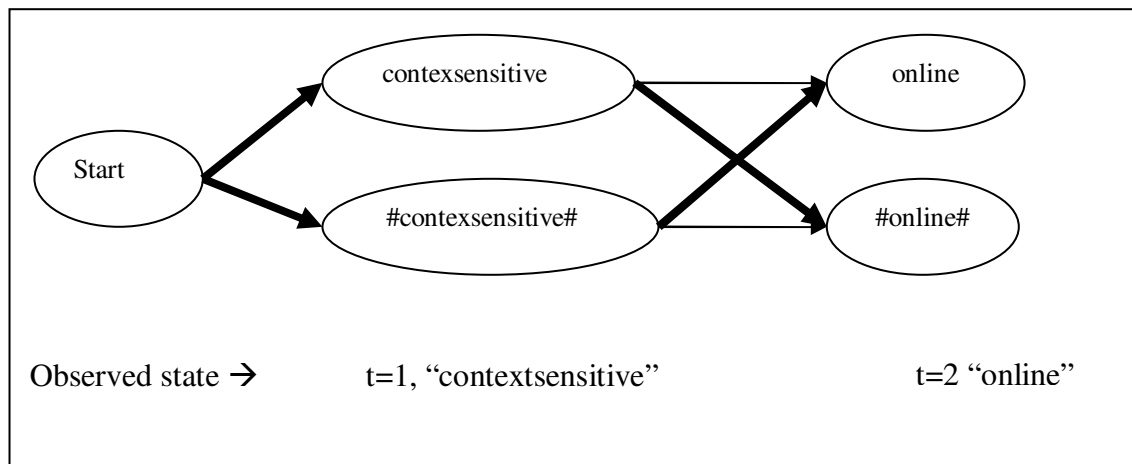
Berikut adalah contoh proses *decode* untuk kalimat “context-sensitive online help is always available”, menggunakan topologi pertama (Gambar III-1) dengan asumsi probabilitas bigram dan probabilitas emisi sudah dihitung terlebih dahulu.

Proses *decode* menghitung nilai *viterbi trellis* secara rekursif. *Viterbi trellis*, $v_t(i)$ adalah probabilitas viterbi path pada *state* ke- i dan saat t . Gambar III-3 memperlihatkan diagram trellis pada saat $t=1$ dengan *observed state* “context-sensitive”. p_{Trans} adalah probabilitas transisi, p_{Emm} adalah probabilitas emisi. Karena semua probabilitas disimpan dalam log maka operator yang digunakan adalah penjumlahan dan dapat bernilai negatif.



Gambar III-3 Diagram trellis untuk $t=1$

Langkah berikutnya digambarkan pada diagram trellis berikut, untuk $t=2$ dengan *observed state* “online”



Gambar III-4 Diagram trellis untuk $t=2$

Nilai $v_2(\text{online})$ dihitung dengan cara sebagai berikut

$$v_2(\text{online}) = \max(\begin{aligned} &v_1(\text{context-sensitive}) + p_{\text{Trans}}(\text{online}|\text{context-sensitive}) + \\ &p_{\text{Emmi}}(\text{online}|\text{online}), \\ &v_1(\#\text{context-sensitive}\#) + p_{\text{Trans}}(\text{online}|\#\text{context-sensitive}\#) + \\ &p_{\text{Emmi}}(\text{online}|\text{online}) \end{aligned})$$

lalu, substitusikan dengan nilai setiap variabel:

$$v_2(\text{online}) = \max(\begin{aligned} &-12.80 + -7.09 + -11.05 = -30.94, \\ &-10.79 + -5.02 + -11.05 = -26.86 \end{aligned})$$

Terlihat bahwa nilai terbesar menuju *state* “online” pada $t=2$ adalah dari *state* “#context-sensitive#”. *State* ini disimpan sebagai *backpointer* agar dapat ditelusuri kembali. Dengan cara yang sama, nilai terbesar menuju *state* “#online#” pada $t=2$ adalah “context-sensitive”.

Proses dilakukan sampai pada *observed state* terakhir yaitu $t=6$, “available”, kemudian dihitung nilai $v_6(i)$ yang terbesar.

Dari hasil perhitungan, nilai probabilitas terbesar yaitu -7.76, diperoleh dari *state* “available”. Penelusuran balik mendapatkan path sebagai berikut: “ <start> #context-sensitive# online help is #always# available” sehingga hasil akhir adalah “online help is available”.

III.4 Preprocessing

Preprocessing dilakukan terhadap data uji coba dan data latihan sebelum proses kompresi dan proses training dilakukan. Selain *casefolding* dan pembuatan huruf non-alphanumerik, diujicobakan dua perlakuan untuk *preprocessing*:

1. Pemberian tag simbol numerik, terdiri atas dua tag: uang {MON}, angka {NUM} dan campuran {MIX}. Hal ini disebabkan *corpus* yang digunakan banyak mengandung angka, baik merupakan uang maupun nama produk. Contoh:

Tabel III-1 Contoh pemberian tag simbol numerik

Sebelum preprocessing	Sesudah preprocessing
The system is priced at \$26,995	the system is priced at {MON}
Dataviews 8.0 also supports Ada	dataviews {NUM} also supports ada
Compaq 386 users awarded the 386/20e and the 386/20 high marks for CPU speed	compaq {NUM} users awarded the {MIX} and the {MIX} high marks for CPU speed

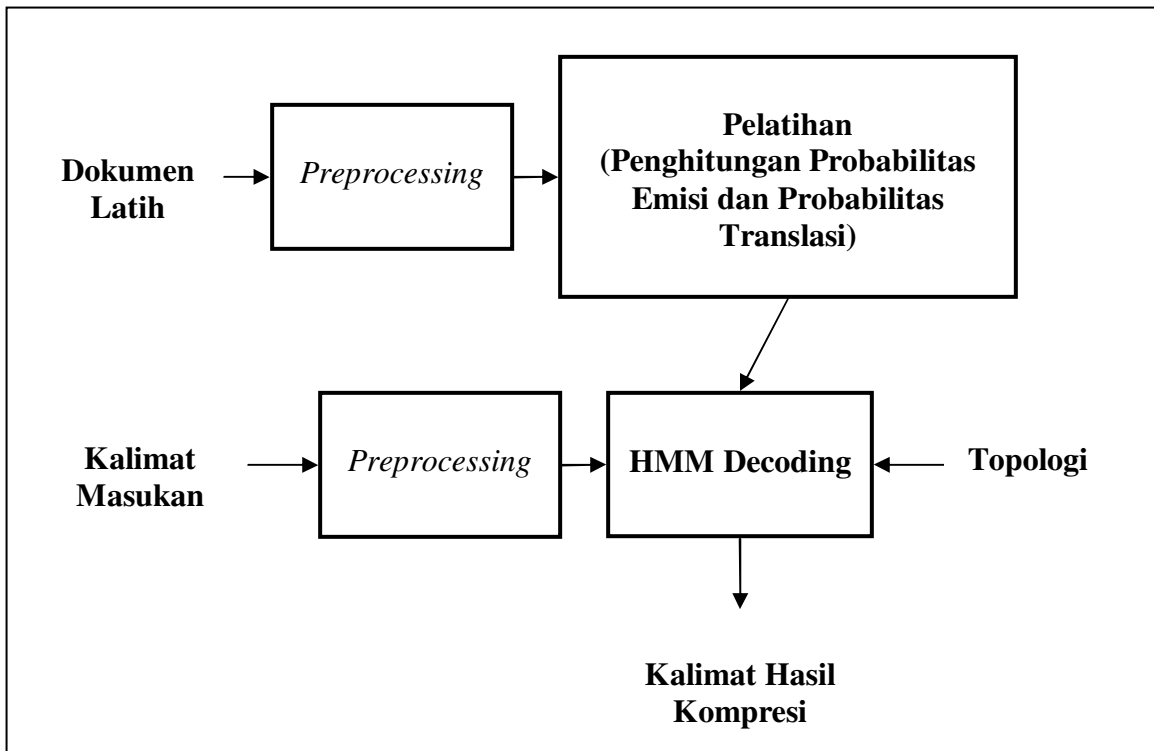
2. Pemberian tag untuk entitas. Kata yang merupakan entitas didefinisikan sebagai kata yang diawali huruf kapital dan berada di tengah kalimat atau kata yang berada di depan kalimat dan seluruh hurufnya terdiri atas huruf kapital. Dua kata entitas yang berurutan akan digabung menjadi satu. Hal ini dilakukan karena banyak nama produk dan isitilah yang hanya muncul di satu kalimat saja, sehingga pola bigramnya tidak akan tertangkap oleh model pada saat pelatihan. Contoh:

Tabel III-2 Contoh pemberian tag simbol entitas

Sebelum preprocessing	Sesudah preprocessing
Much of ATM 's performance depends on the underlying application	much of {NAME} performance depends on the underlying application\
ESRI will develop an interface to Sybase 's SQL Server .	{NAME} will develop an interface to {NAME} server

III.5 Arsitektur Sistem

Arsitektur sistem kompresi HMM ditunjukkan oleh Gambar III-5. Dokumen latihan yang telah di-*preprocessing* digunakan dalam tahap pelatihan untuk menghitung probabilitas emisi dan probabilitas transisi. Probabilitas tersebut bersama dengan arsitektur topologi digunakan dalam proses *decoding* untuk mencari kalimat yang terkompresi berdasarkan kalimat masukan.



Gambar III-5 Arsitektur Sistem Kompresi Kalimat

Proses pelatihan cukup dilakukan satu kali. Setelah probabilitas emisi dan probabilitas transisi diperoleh, HMM *decoding* dapat digunakan untuk berbagai kalimat masukan tanpa perlu melakukan pelatihan ulang.

Bab IV Eksperimen

Bab ini membahas tujuan eksperimen, skenario eksperimen dan hasil eksperimen kompresi kalimat dengan HMM.

Tujuan eksperimen ini adalah:

1. Meneliti pengaruh *preprocessing*, *bigram smoothing*, dan pengaruh bobot α dalam probabilitas terhadap kinerja sistem kompresi kalimat.
2. Meneliti pengaruh topologi terhadap kinerja sistem kompresi kalimat.
3. Membandingkan metode HMM ini dengan metode Knight-Marcu *Noisy Channel* [KNI00].

Eksperimen pada tesis ini menggunakan koleksi Ziff-Davis, yang terdiri atas 1067 kalimat. Koleksi ini dikumpulkan oleh Knight et al. [KNI00] dan berisi pengumuman barang-barang teknologi informasi baik software maupun hardware. Gambar IV-1 memperlihatkan contoh isi koleksi ini. Setiap bagian terdiri atas dua kalimat, kalimat pertama adalah versi kompresi manusia sedangkan di bagian bawah adalah kalimat versi lengkap.

```
The JetForm line includes JetForm Design , JetForm Filler , JetForm
Merger and JetForm Server .
The JetForm product line includes JetForm Design , JetForm Filler ,
JetForm Merger and JetFormServer .

Much of ATM 's performance depends on the underlying application .
Like FaceLift , much of ATM 's screen performance depends on the
underlying application .

Multi-Link offers only a one-year warranty .
Multi-Link offers only a one-year warranty on all parts and labor ,
with unlimited technical support on all of its fax-line-sharing devices
via a toll-free number , from 8:00 to 6:00 E.S.T. , but no after-hours
answering support .
```

Gambar IV-1 Contoh pasangan kalimat pada koleksi Ziff Davis

IV.1 Skenario Eksperimen

Sesuai dengan tujuan eksperimen, skenario eksperimen dibagi menjadi dua bagian.

1. Skenario pertama, meneliti pengaruh *preprocessing*, *bigram smoothing*, dan pengaruh bobot α dalam probabilitas. Pada Tabel IV-1 ditunjukkan nilai dari setiap parameter yang diamati.

Tabel IV-1 Parameter dan nilai yang diteliti dalam eksperimen

No	Parameter	Nilai Parameter
1	<i>Preprocessing</i>	Penambahan tag simbol numerik, penambahan tag entitas
2	<i>Bigram Smoothing</i>	Zue Smoothing ($K=20,40,80,160$) dan Jelinek-Mercer Smoothing ($\gamma = 0.1, 0.5, 0.8$)
3	<i>Bobot probabilitas (α)</i>	0.001, 0.01, 0.05, 0.1, 0.2,0.3,0.4,0.5,0.6,0.7, 0.8, 0.9

2. Skenario kedua, membandingkan kinerja antara topologi pertama dan topologi kedua. Skenario ini menggunakan konfigurasi terbaik dari hasil eksperimen skenario pertama.
3. Skenario ketiga, membandingkan sistem HMM dengan metode Knight-Marcu Noisy Channel [KNI00]. Skenario ini menggunakan konfigurasi terbaik dari hasil eksperimen skenario pertama.

Metode *hold-out* digunakan untuk ketiga skenario tersebut. Digunakan 32 kalimat sebagai data uji dan 1035 kalimat sisanya sebagai data pelatihan. Ke-32 kalimat ini merupakan data yang diambil dari penelitian Knight [KNI00] yang diambil secara acak. Kinerja sistem kompresi kemudian dihitung menggunakan besaran ROGUE-2 [LIN03][LIN04] yang membandingkan hasil kompresi kalimat dengan kalimat referensi.

IV.2 Hasil Eksperimen Skenario Pertama

Eksperimen ini dibagi menjadi tiga bagian yaitu mengamati pengaruh *preprocessing*, *bigram smoothing*, dan bobot α dalam probabilitas. Berikut akan dibahas secara lebih rinci pengaruh setiap parameter.

IV.2.1 Pengaruh Penambahan Tag Simbol Numerik dan Entitas pada *Preprocessing*

Dalam *preprocessing*, selain *casefolding* (mengubah semua karakter menjadi huruf kecil) dan pembuangan karakter non alphanumerik, dikaji penggantian simbol numerik dengan tiga *tag* yaitu: tag {NUM} menyatakan angka biasa, tag {MON} menyatakan uang dan tag {MIX} menyatakan campuran angka dan huruf.

Tabel IV-2 memperlihatkan frekuensi setiap *tag* pada data latihan. *Tag* yang diawali dengan karakter #, yaitu {#MON#}, {#NUM#} dan {#MIX#} merupakan tag simbol numerik yang dihapus dari kata yang dikompresi.

Tabel IV-2 Frekuensi tag simbol numerik pada data latihan

Tag	Jumlah
{MON}	289
{#MON#}	47
{NUM}	444
{#NUM#}	114
{MIX}	185
{#MIX#}	79

Selain tag simbol numerik, dikaji juga penggunaan *tag* entitas {NAME}, yaitu tag yang menggantikan kata yang merupakan entitas nama. Contoh kata yang diganti oleh *tag* ini adalah "IBM", "Compaq" dan seterusnya. Terdapat 2025 tag {NAME} pada dokumen latihan.

Tabel ini memperlihatkan bahwa *preprocessing* untuk tag {MON}, {NUM} dan {MIX} meningkatkan nilai ROGUE-2, tetapi penambahan tag entitas {NAME} justru memperburuk kinerja sistem. Dari pengamatan terhadap hasil kompresi, hal ini disebabkan karena beberapa pasangan memiliki probabilitas bigramnya yang terlalu tinggi. Misalnya bigram "the {NAME}".

Tabel IV-3 memperlihatkan nilai ROGUE-2 untuk kompresi yang menggunakan data tanpa *tag*, data dengan *tag angka* dan data dengan *tag entitas*. Tabel ini memperlihatkan bahwa *preprocessing* untuk tag {MON}, {NUM} dan {MIX} meningkatkan nilai ROGUE-2, tetapi penambahan tag entitas {NAME} justru memperburuk kinerja sistem.

Dari pengamatan terhadap hasil kompresi, hal ini disebabkan karena beberapa pasangan memiliki probabilitas bigramnya yang terlalu tinggi. Misalnya bigram "the {NAME}".

Tabel IV-3 ROUGE-2 untuk preprocessing

Perlakuan	ROUGE-2	Perubahan (%)
Tanpa Tag (<i>baseline</i>)	0.5005	-
Tag simbol numerik {MON} {NUM} {MIX}	0.5101	1.9181
Tag simbol numerik {MON} {NUM} {MIX} dan Tag entitas {NAME}	0.4849	-3.1169

Untuk eksperimen berikutnya, diaplikasikan *preprocessing* dengan *tag* untuk simbol numerik.

IV.2.2 Pengaruh *Bigram Smoothing*

Uji coba dilakukan untuk mengetahui sejauh mana pengaruh *bigram smoothing* pada kinerja sistem. Ada dua jenis *bigram smoothing* yang digunakan, yaitu *Zue Smoothing* [ZUE92] dan *Jelinek Mercer Smoothing* [JEL82]

Tabel IV-4 memperlihatkan uji coba sistem HMM dengan berbagai nilai K untuk *Zue Smoothing* [ZUE92].

Tabel IV-4 ROUGE-2 Zue Smoothing

K	ROGUE-2	Perubahan (%)
Tanpa <i>Smoothing</i>	0.5101	-
k=20	0.5072	-0.5685
k=40	0.5106	0.0980
k=80	0.5106	0.0980
k=160	0.5100	-0.0196

Dari Tabel IV-4 terlihat penggunaan *Zue Smoothing* tidak terlalu memperbaiki kinerja sistem.

Berikutnya dikaji menggunakan *Jelinec-Mercer (J-M) Smoothing* [JEL80]. Tabel IV-5 memperlihatkan uji coba sistem HMM dengan berbagai nilai γ .

Tabel IV-5 Nilai ROUGE-2 untuk J-M Smoothing

γ	ROGUE-2	Perubahan (%)
Tanpa Smoothing	0.5101	-
$\gamma = 0.1$	0.5478	7.3907
$\gamma = 0.5$	0.5122	0.4117
$\gamma = 0.8$	0.5062	-0.7646

Dapat dilihat pada $\gamma = 0.1$ penggunaan *J-M smoothing* meningkatkan kinerja sistem dan memiliki nilai ROGUE-2 lebih baik dibandingkan dengan teknik *Zue Smoothing*. Oleh karena itu, untuk eksperimen selanjutnya digunakan *J-M Smoothing* dengan $\gamma = 0.1$.

IV.2.3 Pengaruh Bobot α

Seperti dibahas sebelumnya, bobot α ditambahkan dalam model probabilitas sesuai persamaan III-7. Semakin tinggi nilai α berarti memberikan bobot yang semakin tinggi kepada probabilitas transisi (bigram) dan mengurangi peran probabilitas emisi. Hasil eksperimen berikut memperlihatkan nilai ROUGE-2 untuk berbagai nilai α .

Tabel IV-6 ROUGE-2 untuk berbagai nilai α

α	ROGUE-2	Perubahan (%)
$\alpha = 0.001$	0.5524	0.0084
$\alpha = 0.01$	0.5587	0.0198
$\alpha = 0.05$	0.5581	0.0188
$\alpha = 0.1$	0.5650	0.0314
$\alpha = 0.2$	0.5479	0.0002
$\alpha = 0.3$	0.5468	-0.0018
$\alpha = 0.4$	0.5480	0.0004
$\alpha = 0.5$	0.5478	-
$\alpha = 0.6$	0.5310	-0.0307
$\alpha = 0.7$	0.5310	-0.0307
$\alpha = 0.8$	0.5334	-0.0263
$\alpha = 0.9$	0.5334	-0.0263

Dari Tabel IV-6 dapat dilihat bahwa nilai ROUGE-2 mencapai maksimum pada $\alpha = 0.1$.

Berdasarkan eksperimen skenario pertama, penggunaan *preprocessing* simbol numerik, *J-M Smoothing* ($\gamma=0.1$) dan pengaturan $\alpha = 0.1$, meningkatkan kinerja sistem sebesar 12.89 persen.

IV.3 Hasil Eksperimen Skenario Kedua

Pada skenario kedua, diujicobakan dua topologi menggunakan konfigurasi terbaik dari eksperimen pertama. Topologi pertama (Gambar III-1) membatasi transisi *hidden state* sehingga urutan kata keluaran akan sesuai dengan urutan pada kalimat aslinya. Topologi kedua (Gambar III-2) lebih fleksibel, karena semua *hidden state* saling terhubung. Hasil eksperimen berikut memperlihatkan nilai ROUGE-2 untuk kedua topologi.

Tabel IV-7 ROUGE-2 untuk kedua topologi

Topologi	ROGUE-2
Topologi pertama, dengan batasan keterurutan	0.5650
Topologi kedua, semua hidden state terhubung	0.3120

Terlihat bahwa topologi pertama jauh lebih baik dibandingkan topologi kedua. Berdasarkan pengamatan terhadap hasil, topologi kedua menghasilkan nilai yang lebih rendah karena sering menghasilkan urutan kata yang tidak beraturan dan dapat menghasilkan pengulangan urutan kata yang tidak pada tempatnya, misalnya "the system the system".

IV.4 Hasil Eksperimen Skenario Ketiga

Eksperimen ini membandingkan kinerja HMM dengan Knight-Marcu (K-M) Noisy Channel [KNI00]. Dalam eksperimen ini, HMM menggunakan konfigurasi yang memberikan nilai ROGUE-2 tertinggi yaitu: *preprocessing* untuk tag angka, *J-M smoothing* dengan $\gamma = 0.1$ dan bobot probabilitas $\alpha = 0.1$ dengan topologi pertama. Hasil eksperimen ini diperlihatkan pada Tabel IV-8.

Tabel IV-8 ROUGE-2 antara HMM dan K-M Noisy Channel

Metode	ROGUE-2
K-M Noisy Channel	0.6782
HMM	0.5650

Tabel IV-8 memperlihatkan bahwa nilai ROUGE-2 HMM masih lebih rendah dibandingkan K-M Noisy Channel. Hal ini mungkin diakibatkan HMM tidak menggunakan model sintaks dan tatabahasa sehingga kalimat yang dihasilkan tidak setepat model K-M Noisy Channel.

Gambar IV-2 memperlihatkan contoh hasil HMM dibandingkan dengan K-M Noisy Channel.

asli:	like facelift much of atm screen performance depends on the underlying application
ideal:	much of atm performance depends on the underlying application
HMM:	facelift of atm performance depends on the application
KM-Noisy:	much of atm performance depends on the underlying application
asli:	also trackstar supports only the critical path method cpm of project scheduling
ideal:	trackstar supports the critical path method of project scheduling
HMM:	trackstar supports the cpm of project scheduling
KM-Noisy:	trackstar supports only the critical path method cpm of scheduling
asli:	beyond that basic level the operations of the three products vary widely
ideal:	the operations of the three products vary widely
HMM:	basic level the operations of the three products vary
KM-Noisy:	the operations of the three products vary widely
asli:	arborscan is reliable and worked accurately in testing but it produces very large dxf files
ideal:	arborscan produces very large dxf files
HMM:	arborscan is reliable and testing produces large dxf files
KM-Noisy:	arborscan is but it produces large dxf files

Gambar IV-2 Contoh perbandingan hasil HMM dengan Knight-Marcu Noisy Channel

Gambar IV-3 memperlihatkan contoh hasil HMM yang memiliki kinerja lebih baik dibandingkan dengan metode K-M Noisy Channel.

Lengkap:	the source code which is available for c fortran ada or vhdl can be compiled and executed on the same system or ported to other target platforms
Ideal:	the source code is available for c fortran ada or vhdl
Auto:	the source code is available for c fortran ada vhdl can be compiled and executed on the system ported to target platforms
Noisy C.:	the source code can be compiled and executed on the system or ported to other target platforms
Rouge-2 HMM:	0.75
Rouge-2 NOISY:	0.25
Lengkap:	the first new product atf prototype is a line of digital postscript typefaces that will be sold in packages of up to six fonts
Ideal:	atf prototype is a line of digital postscript typefaces that will be sold in packages of up to six fonts
HMM:	the atf prototype is a line of typefaces will be sold packages of up to six fonts
Noisy C.:	the new product atf prototype is a line of postscript typefaces
Rouge-2 HMM:	0.62
Rouge-2 NOISY:	0.29
Lengkap:	the utilities will be bundled with quickdex ii in a [MON] package called super quickdex which is expected to ship in late summer
Ideal:	the utilities will be bundled with quickdex ii
HMM:	the utilities will be bundled with quickdex ii a [MON] quickdex is expected to ship summer
Noisy C.:	the utilities will be bundled
Rouge-2 HMM:	0.89
Rouge-2 NOISY:	0.56
Lengkap:	the discounted package for the sparcserver [NUM] is priced at [MON] down from the regular [MON]
Ideal:	the sparcserver [NUM] is priced at [MON] down from the regular [MON]
HMM:	the discounted for the [NUM] is priced at [MON] the [MON]
Noisy C.:	the package for the [NUM] is priced
Rouge-2 HMM:	0.46
Rouge-2 NOISY:	0.23

Gambar IV-3 Contoh hasil HMM yang lebih unggul dari Knight-Marcu Noisy Channel

Gambar IV-4 dan Gambar IV-5 memperlihatkan tiga kalimat terbaik dan tiga kalimat terburuk yang dihasilkan oleh HMM.

Hasil selengkapnya, beserta perbandingan hasil kompresi dengan K-M Noisy Channel dapat dibaca pada lampiran A.

ROGUE-2: 0.8889	asli: the utilities will be bundled with quickdex ii in a [MON] package called super quickdex which is expected to ship in late summer
	ideal: the utilities will be bundled with quickdex ii
	HMM: the utilities will be bundled with quickdex ii a [MON] quickdex is expected to ship summer
ROGUE-2: 0.8750	asli: actual hardware and maintenance costs have decreased [NUM] percent over the years while the number of users supported has increased by over percent since [NUM]
	ideal: actual hardware and maintenance costs have decreased
	HMM: actual hardware and maintenance costs have decreased [NUM] percent over the years the users supported has increased over percent [NUM]
ROGUE-2: 0.8333	asli: another slight downside is that envelopes must be fed manually
	ideal: envelopes must be fed manually
	HMM: slight downside is envelopes must be fed manually

Gambar IV-4 Tiga kalimat hasil kompresi HMM dengan skor ROGUE-2 tertinggi

ROGUE-2: 0.0000	asli: many debugging features including userdefined break points and variablewatching and messagewatching windows have been added
	ideal: another advantage is distance
	HMM: advantage of broadband is
ROGUE-2: 0.1429	asli: many debugging features including userdefined break points and variablewatching and messagewatching windows have been added
	ideal: many debugging features have been added
	HMM: debugging features userdefined break points and variablewatching and messagewatching windows
ROGUE-2: 0.2500	asli: working in the score is not an intuitive process it takes a lot of practice
	ideal: working in the score is not intuitive
	HMM: the score is an intuitive process takes of practice

Gambar IV-5 Tiga kalimat hasil kompresi HMM dengan skor ROGUE-2 terendah

Bab V Kesimpulan dan Saran

Bab ini menjelaskan kesimpulan yang didapatkan selama pelaksanaan penelitian dan penulisan tesis ini. Selain itu, bagian saran menjelaskan mengenai perbaikan yang perlu dilakukan atau kemungkinan lain yang belum sempat dilakukan dalam tesis ini.

V.1 Kesimpulan

Kesimpulan yang didapatkan selama pelaksanaan penelitian dan penulisan tesis ini adalah sebagai berikut.

1. Kinerja HMM untuk kompresi kalimat dipengaruhi oleh: penambahan tag simbol numerik pada *preporcessing*, *bigram smoothing*, dan pembobotan probabilitas.
2. Topologi merupakan komponen yang penting dalam HMM karena kesalahan dalam merancang topologi dapat menurunkan kinerja sistem.
3. Kinerja sistem HMM masih lebih rendah dibandingkan dengan Knight-Marcu Noisy Channel yang menggunakan *syntatic knowledge*.

V.2 Saran

Beberapa hal yang sebaiknya dilakukan dalam penelitian selanjutnya adalah sebagai berikut.

1. Karena kelebihan utama kompresi kalimat dengan HMM adalah tidak bergantung kepada bahasa, HMM perlu diujicoba untuk dokumen bahasa lain (terutama Bahasa Indonesia) dan *cross lingual task*.
2. Melakukan pengkajian lebih lanjut antara HMM dengan metode Knight Marcu Noisy Channel untuk dokumen yang memiliki tingkat *noise* tinggi.
3. Perlu dikembangkan sistem yang dapat melakukan konfigurasi otomatis terhadap bobot probabilitas (α) melalui pembelajaran.

Daftar Pustaka

- [BAN00] Banko, M. Mittal, V. O. Witbrock, M. J. (2000), "Headline Generation Based on Statistical Translation", Annual Meeting- Association For Computational Linguistics, Vol 38; Part 1, pages 318 – 325
- [BER00] Berger, A.L., Mittal, V.O. (2000), "OCELOT: a system for summarizing Web pages", Research and Development in Information Retrieval, pages 144-151
- [BUY01] Buyukkokten O., Garcia-Molina H., Paepcke A., " Seeing the whole in parts: Text summarization for webBrowsing on handheld devices", Proceedings of the 10th International WWW Conference, HongKong, China, 2001.
- [CHE96] SF Chen, J Goodman, (1996) An Empirical Study of Smoothing Techniques for Language Modeling, Proceedings of the 34th annual meeting on Association for Computational Linguistik
- [COR99] Corston-Oliver, S. H, Dolan, W.B. Less is more: eliminating index terms from subordinate clauses Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics table of contents
- [DOR03] Dorr B, Zajic D., Schwartz R., Cross-language headline generation for Hindi ACM Transactions on Asian Language Information Processing (TALIP), 2003
- [DOR04] Doran, W., Stokes, N., Newman, E., Dunnion, J., Carthy, J., Toolan, F., (2004) "News Story Gising at University College Dublin", Document Understanding Conference, DUC 2004
- [EIS06] J. Eisner, "Assignment 5: Tagging with Hidden Markov Model", course handout.
- [FOR73] Forney, G.D., "The Viterbi Algorithm" (1973), Proceeding of The IEEE, Vol 61, No 3, page 268-277
- [GOO08] Google News, <http://news.google.com>, diakses tanggal 14 Juni 2008
- [KNI00] Knight, K., and Marcu, D. 2000. Statistics Based Summarization: Step One: Sentence Compression. Proceedings of the 17th National Conference of the American Association for Artificial Intelligence AAAI2000, Austin, Texas, July 30-August 3, 2000
- [NGU04] M Le Nguyen, S Horiguchi, A Shimazu, BT Ho (2004), Example-based sentence reduction using the hidden markov model, ACM Transactions on Asian Language Information Processing
- [JEL80] Jelinek, Frederick and Mercer, RL.. Interpolated Estimation of Markov Source Parameters from Sparse Data. In Proceedings of the Workshop on Pattern Recognition in Practice, Amsterdam, The Netherlands: North-Holland, 1980
- [JING00] Jing H, (2000), Sentence reduction for automatic text summarization - Proceedings of the 6th Applied Natural Language Processing
- [JIN00] Jin, R., Huptmann, A.G. (2000), "Headline Generation using a Training Corpus", In CICLING 2000.

- [JUR06] Jurafsky, D., Martin, J.H. (2006) *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*.
- [KAN01] Kan, M.Y., McKeown, K.R., Klavans, J. (2001), "Domain-specific informative and indicative summarization for information retrieval", *Proceedings of the Document Understanding Conference*
- [LIN03] C.Y. Lin, "ROUGE: Recall-Oriented Understudy for Gisting Evaluation," (2003)
- [LIN04] C.-Y. Lin, "Looking for a few good metrics: ROUGE and its evaluation," *Working Notes of NTCIR-4 (Vol. Supl. 2) 1-8*, (2004)
- [RAB89] Rabiner, LR (1989), *A tutorial on hidden Markov models and selected applications inspeech recognition*, *Proceedings of the IEEE*
- [RAD00] Radev, D.R, Fan,W. (2000), "Automatic summarization of search engine hit lists", *Proceedings ACL Workshop on Recent Advances in NLP and IR*
- [RAD02] Radev, D., Hovy, E., McKeown, K. (2002). "Introduction to the special issue on text summarization." *Computer Linguist*, page 28(4).
- [WIT99] Witbrock, M.J., Mittal, V.O, (1999) "Ultra-Summarization: A Statistical Approach to Generating Highly Condensed Non-Extractive Summaries (poster abstract)", *Research and Development in Information Retrieval*, pages 315-316
- [YAM01] Yamada, K., Knight K, *A syntax-based statistical translation model*, *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics 2001*
- [ZAJ02] Zajik D, Dorr B, (2002), *Automatic Headline Generation for Newspaper Stories*,
- [ZAJ07] Zajic,D.etal., (2007), *Multi-candidate reduction: Sentence Compression as a tool*, *InforMation Processing and Management*
- [ZUE92] Zue V., (1992), *The MIT ATIS System: February 1992 Progress Report 1*