BAB 1 LANDASAN KEPUSTAKAAN

1.1 Kajian Pustaka

Penelitian yang dilakukan oleh Chopra pada tahun 2012 dengan judul "*Hindi Named Entity Recognition by Aggregating Rule based heuristics and Hidden Markov Model*" bertujuan untuk mengenali entitas seperti lokasi, nama orang, jumlah, waktu. Penelitian ini menggunakan metode *Hidden Markov Model* (HMM) dan *Rule based heuristics* atau *Parsing technique*. Penelitian ini menghasilkan akurasi 89.78% menggunakan HMM, 47.5% menggunakan *Rule based heuristics* dan 94.61% dengan penggabungan 2 metode (Chopra, et al., 2012).

Penelitian yang dilakukan oleh Dey pada tahun 2014 dengan judul "*Named Entity Recognition for nepali language: a semi hybrid approach*" bertujuan untuk mengenali entitas seperti nama orang, lokasi, organisasi pada Bahasa Nepal. Pada penelitian ini menggunakan algoritme *lexical lookup*, HMM, *Rule base* dan *n-gram technique*. Penelitian ini menghasilkan akurasi yang mencapai 90.69% (Dey, et al., 2014).

Penelitian yang dilakukan oleh Mansouri pada tahun 2008 dengan judul "*Named Entity Recognition* Approaches" bertujuan untuk mengenali entitas-entitas pada Message Understanding Conference dataset. Penelitian ini menggunakan *Support Vector Machine* menghasilkan akurasi yang mencapai 86.40%. Kekurangan pada penelitian ini tidak disebutkan entitas apa yang dikenali dan tidak disebutkan jumlah data latih serta contoh pada pengenalan *Named Entity Recognition* untuk data latih yang digunakan (Mansouri, et al., 2008).

Penelitian selanjutnya yang dilakukan oleh Wu Sen pada tahun 2012 dengan judul "Accurate Product Name Recognition from User Generated Content" bertujuan untuk mengenali nama-nama produk secara tekstual pada katalog dengan meminimalisir abiguitas menggunakan beberapa gabungan metode. Penelitian ini menggunakan metode *Standart Matching, Rule Template* dan *Conditional Random Fields*. Penelitian ini menghasilkan pengenalan nama produk yang memiliki akurasi yang baik dengan penggabungan tiga metode (Wu, et al., 2012). Kekurangan pada penelitian ini hanya mengenali 1 entitas.

Tabel 1.1 Perbandingan Penelitian Sebelumnya dengan Penelitian Penulis

No	Penulis	Objek	Metode
1	(Chopra, et al., 2012)	Pengenalan entitas menggunakan Bahasa India	*Hidden Markov Model* dan *Rule Based Heuristics*
2	(Dey, et al., 2014)	Pengenalan entitas menggunakan Bahasa Nepal	*Hidden Markov Model* dan *Rule Based Approaches*

Tabel 1.2 Perbandingan Penelitian Sebelumnya dengan Penelitian Penulis (lanjutan)

3	(Mansouri, et al., 2008)	Pengenalan entitas pada Message Understanding Conference	*Support Vector Machine*

4	(Wu, et al., 2012)	Pengenalan entitas nama produk	*Standart Matching, Rule Template* dan *Conditional Random Field*

1.2 E-Commerce

Perdagangan elektronik atau *e-commerce* merupakan sebuah aktivitas bisnis seperti pertukaran informasi, sistem manajemen, inventaris otomatis, pengumpulan data otomatis dan transaksi yang menggunakan teknologi informasi dan komunikasi. Fasilitas internet yang memiliki layanan *get and deliver* pada website dapat digunakan untuk berdagang maupun berbelanja secara *direct selling*. Dapat diambil kesimpulan bahwa *e-commerce* adalah aktivitas penjualan dan pembelian suatu produk baik jasa maupun benda melalui jaringan internet sebagai media pertukaran informasi dan transaksi. Kehadiran *e-commerce* tentu memiliki beberapa manfaat diantaranya adalah (Sunarto, 2009):

1. Menghemat jam kerja manusia dengan adanya fasilitas internet sehingga dapat dilakukan di rumah.
2. Transaksi melalui *e-commerce* dapat dilakukan tanpa mengenal batasan waktu dan batasan wilayah geografis.
3. Menurunkan biaya operasi *(operating cost).*

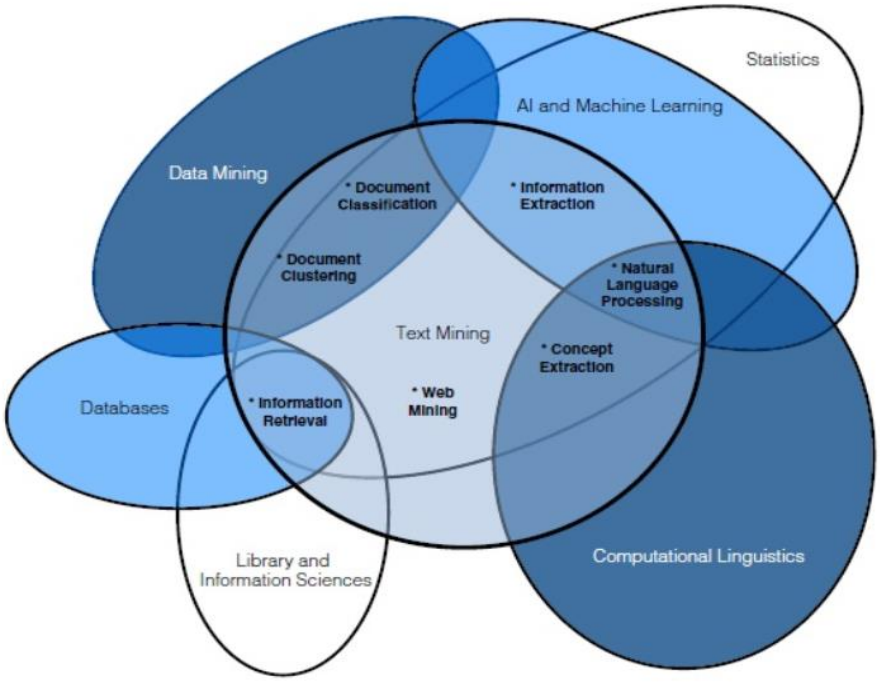
Menurut (Nickerson, 2002) fungsi atau fitur *e-commerce* secara umum meliputi *product presentation*, *electronic catalogs*, *order entry*, *order confirmation*, *electronic payment*, *order fulfillment*, and *customer service*. Pada saat sebelum, selama, atau setelah pembelian produk, pelanggan mungkin membutuhkan pelayanan khusus. Sebagai contoh, pelanggan mungkin memilliki pertanyaan mengenai sebuah produk sebelum membelinya. Selama proses pemesanan, pelanggan mungkin mengalami kesulitan dalam menggunakan sistem *e-commerce*. Setelah menerima pesanannya, pelanggan mungkin ingin menukar atau mengembalikan produk yang telah dibelinya. Situasi-situasi tersebut dapat diatasi dengan memberikan informasi detil dan jawaban pertanyaan secara elektronik meliputi FAQ, nomor telepon, *email* atau *chat*.

Setelah pelanggan menerima produk yang dibelinya, mungkin saja pelanggan masih membutuhkan dukungan bisnis seperti pengoperasian produk, perbaikan, pengembalian barang. Dukungan terhadap produk bisa disediakan oleh sistem *e-commerce* dengan mengikut sertakan informasi detil mengenai produk dalam situs *web*. Sistem *e-commerce* juga bisa menyediakan system *e-mail* sehingga pelanggan dapat mengirim pertanyaan dan menerima jawabannya melalui *e-mail*. Atau mungkin disediakan sebuah sistem *chat* bagi pelanggan untuk dapat berinteraksi dengan *customer service* (Nickerson, 2002).

1.3 Text Mining

Text mining merupakan suatu proses untuk mengekstrak pola dalam mengeksplorasi pengetahuan dari sumber data yang berbentuk teks. *Text mining* juga disebut sebagai bidang multi-disiplin berdasarkan pencarian informasi, data *mining*, pembelajaran mesin, statistik, dan linguistik komputasi. Gambar 2.1 menunjukkan diagram Venn tentang *text mining* dan interaksinya dengan bidang lainnya. Beberapa teknik pengolahan teks seperti *summarization*, klasifikasi, *clustering*, dan sebagainya dapat diterapkan untuk mengekstrak informasi. *Text*

mining berhubungan dengan teks bahasa alami yang disimpan dalam format semi terstruktur dan tidak terstruktur. Teknik pengolahan teks terus diterapkan di industri, akademisi, aplikasi web, internet dan bidang lainnya. Beberapa aplikasi seperti mesin pencari, sistem *customer service*, filter *e-mail*, analisis saran produk, deteksi kecurangan, dan analisis media sosial menggunakan *text mining* untuk opini *mining*, ekstraksi fitur, sentimen, prediksi, dan analisis tren (Talib, et al., 2016).



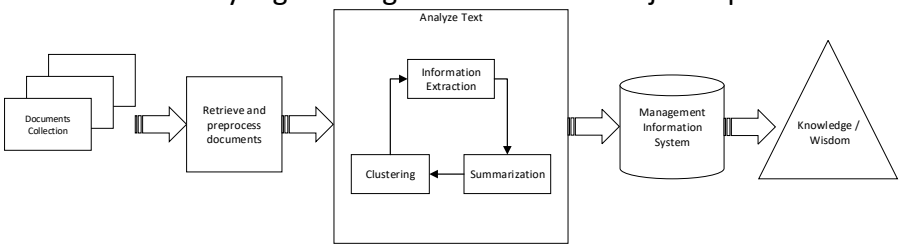
Gambar 1.1 Diagram venn text mining dan interaksinya dengan bidang lainnya

Sumber: Talib (2016)

Menurut (Talib, et al., 2016) proses generik *text minning* dilakukan dengan langkah-langkah berikut:

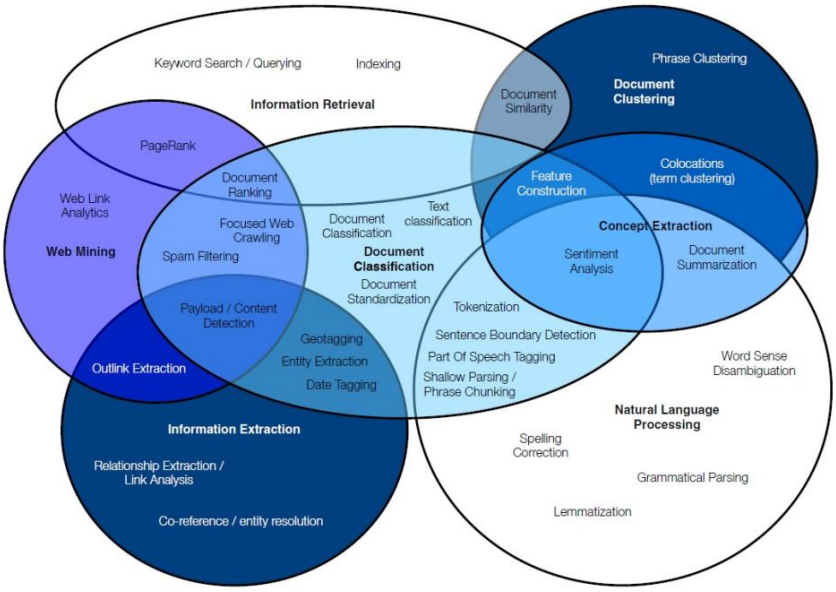
- Mengumpulkan data tidak terstruktur dari berbagai sumber yang tersedia dalam berbagai format file seperti teks biasa, halaman web, file pdf dan sebagainya.
- *Pre-processing* dan pembersihan data dilakukan untuk mendeteksi dan menghapus anomali pada data. Proses pembersihan harus memastikan untuk menangkap esensi teks sebenarnya yang tersedia, dilakukan untuk menghilangkan kata yang tidak memiliki makna atau yang tidak penting dan pengindeksan data .
- Pemrosesan dan pengendalian diterapkan untuk mengaudit kemudian membersihkan data dengan pemrosesan otomatis.
- Analisis pola diimplementasikan oleh Manajemen Sistem Informasi (MSI).

Informasi yang diproses dalam langkah-langkah di atas digunakan untuk mengekstrak informasi yang berharga dan relevan ditunjukan pada Gambar 2.2.



Gambar 1.2 Proses *Generik* Text Mining

Sumber: diadaptasi dari Talib (2016)



Gambar 1.3 Teknik *text mining* dan fungsi utamanya

Sumber: Talib (2016)

Perbedaaan teknik *text mining* diterapkan untuk menganalisis pola teks dan proses utamanya. Pada Gambar 2.3 Menunjukan diagram *Venn* keterkaitan antara Teknik *text mining* dan fungsi mereka. Terdiri dari klasifikasi dokumen, pencarian informasi, pengelompokan dokumen, pengolahan bahasa alami, ekstraksi informasi dan web *mining* (Talib, et al., 2016).

1.4 Named Entity Recognition

Named Entity Recognition (NER) merupakan bagian dari ekstraksi informasi yang bertugas untuk pengklasifikasi teks dari sebuah dokumen atau korpus yang dikategorikan seperti nama orang, lokasi, organisasi, bulan, tanggal, waktu dan sebagainya. Dan selain itu, NER digunakan pula untuk NLP (*Natural Language Processing*). Beberapa implementasinya yaitu mesin *translate*, *search engine, question answering, information retrieval,* pengindeksan dokumen dan sebagainya (Mansouri, et al., 2008).

NER yang dilakukan oleh manusia bukan hal sulit, karena banyak *named entity* kata benda dan diawali dengan huruf kapital sehingga mudah dikenali, tetapi menjadi sulit jika akan dilakukan otomatisasi dengan menggunakan mesin. Penggunaan kamus sering kali mempermudah proses pengenalan, tetapi *named entity* bukan sesuatu yang statis yang akan

berkembang jumlahnya, sehingga dengan menggunakan kamus statis akan memiliki keterbatasan. Masalah yang sering kali muncul dalam identifikasi *named entity* adalah adanya *semantic ambiguity*. NER diimplementasikan dalam banyak bidang, antara lain dalam *machine translation*, *question-answering machine system*, *indexing* pada *information retrieval*, klasifikasi dan juga dalam *automatic summarization*. Tujuan yang diharapkan dari proses dalam NER adalah untuk melakukan ekstraksi dan klasifikasi nama ke dalam beberapa kategori dengan mengacu kepada makna yang tepat (Mansouri, et al., 2008).

Terdapat dua jenis model *machine learning* atau pembelajaran mesin yang dapat digunakan untuk NER, yaitu model pembelajaran *supervised* dan *unsupervised*. *Supervised learning* atau pembelajaran terbimbing menggunakan program yang dapat belajar untuk mengklasifikasikan kumpulan data yang diberikan berdasarkan label yang telah dibuat dengan jumlah fitur yang sama. Pembelajaran ini disebut terbimbing karena data latih yang ada digunakan untuk 'mengajari' komputer agar dapat mengenali data. Pendekatan *supervised learning* membutuhkan persiapan data latih berlabel untuk membangun model statistiknya dengan jumlah yang besar guna mencapai kinerja yang baik. Model *unsupervised learning* model belajar tanpa umpan balik apapun, di dalam *unsupervised* bertujuan untuk membangun representasi data yang kumudian dapat digunakan untuk kompresi data, klasifikasi, pengambilan keputusan dan keperluan lainnya. *Unsupervised learning* ini merupakan pendekatan yang kurang popular untuk NER dan biasanya sistem yang menggunakan pendekatan ini tidak sepenuhnya *unsupervised* (Mansouri, et al., 2008).

Menurut (Dey, et al., 2014) ada tiga pendekatan NER, diantaranya adalah pendekatan *Rule based,* statistik dan pendekatan *Hybrid.* Pendekatan *Ruled based* dapat berupa pendekatan *List Lookup* atau pendekatan linguistik. Untuk deteksi NER menggunakan pendekatan *List Lookup* atau pendekatan linguistik, diperlukan banyak usaha. Daftar *Gazetteer* besar harus dibuat untuk kelas *Named Entity* (NE). Kemudian, operasi pencarian dilakukan untuk menemukan bahwa kata yang diberikan di dalam korpus berada di kategori pada NE. Pada sebuah pendekatan linguistik, menetapkan aturan dan algoritme untuk menentukan NE dalam sebuah korpus dan juga mengklasifikasikan NE ini ke dalam entitas. Dalam pendekatan statistik diperlukan jumlah tenaga manusia yang sangat sedikit dan jenisnya sebagai berikut:

1. *Hidden Markov Model* (HMM)
2. *Maximum Entropy Model* (MEM)
3. *Conditional Random Field* (CRF)
4. *Support Vector Machine* (SVM)
5. *Decision Tree* (DT)

Dalam pendekatan *Hybrid*, *hybrid* dapat meningkatkan kinerja sistem NER. *Hybrid* juga bisa menjadi kombinasi antara model linguistik dan statistik seperti daftar *Gazetteer* dan HMM, HMM dan CRF atau CRF dan MEM dan lainnya.

Sebagai contoh NER pada Bahasa Hindi:

"Mohit/PER ne/O mi road/LOC se/O kitab/O khareedi/O I/O"

Tugas dari NER adalah mengekstrak dan mengklasifikasi NE kedalam kelas tertentu. Dalam contoh ini "Mohit" memiliki kelas sebagai nama seseorang, jadi ini ditunjukan dengan *tag* PER. "mi road" merupakan nama dari lokasi, jadi dapat diberikan tanda LOC pada kata

tersebut. Tanda dari NE yang dipilih mungkin banyak variasinya. Hal ini tergantung pilihan dari individu dan konten yang ditentukan dalam NER (Chopra, et al., 2012).

1.5 Hidden Markov Model

Hidden Markov Model (HMM) merupakan model statistic untuk memodelkan data sekuensial atau data deret waktu, dan telah berhasil digunakan dalam banyak tugas seperti pengenalan ucapan, analisis urutan protein/DNA, control robot, dan ektraksi informasi dari data teks. HMM memiliki tiga utama permasalahan, yaitu (Boodidhi, 2011):

1. *Evaluation Problem*: untuk menyelesaikan permasalahan ini dapat menggunakan *forward* atau *backward* algoritme.
2. *Decoding*: Untuk menyelesaikan permasalahan ini dapat menggunakan teknik *dynamic programming* (DP). Algoritme Viterbi termasuk algoritme yang menggunakan DP.
3. *Training*: untuk *training* atau *learning* dapat menggunakan *supervised training* atau *unsupervised training*. *Maximum Likelihood Estimation* termasuk algoritme yang menggunakan *supervised training*.

HMM dapat dikatakan sebagai model urutan yang mana tugasnya menetapkan label atau kelas ke masing-masing unit secara berurutan. HMM adalah model urutan probabilistik urutan dari unit-unit (kata, huruf, morfem, kalimat dan lain sebagainya) yang kemudian dihitung nilai distribusi probabilitas untuk didapatkan urutan label yang memungkinkan dan memilih urutan label yang terbaik. HMM ditentukan oleh komponen-komponen berikut, yaitu (Jurafsky & Martin, 2017):

1. $Q = q_1q_2…q_n$, sebagai kumpulan N kondisi.
2. $A = a_{01}a_{02}…a_{n1}…a_{nn}$, A merupakan matriks *transition probability.* Setiap a_{ij} mewakili probabilitas perpindahan dari kondisi i ke kondisi j.
3. $O = o_1o_2…o_t$, merupakan urutan dari t pengamatan masing-masing diambil dari kosakata.
4. $B = b_i(o_t)$, merupakan urutan pengamatan *likelihood*, disebut juga *emission probabilities* yang mana masing-masing mewakili probabilitas dari pengamatan o_t yang dihasilkan dari kondisi i.
5. q_o,q_f, sebagai kondisi awal dan kondisi akhir (final) yang tidak berhubungan dengan pengamatan.

Maximum likelihood estimate untuk mengitung *transition probability*:

$$P(t_i|t_{i-1}) = \frac{C(t_{i-1},t_i)}{C(t_{i-1})} \quad (2.1)$$

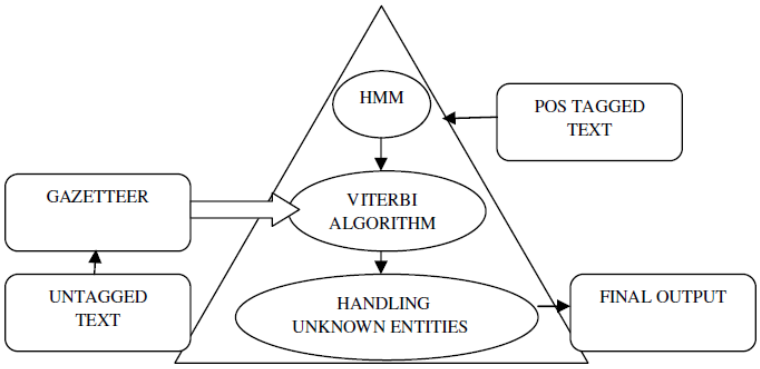
$P(t_i|t_{i-1})$ merupakan nilai *transition probability* atau probabilitas kemunculan tag ke i setelah tag ke i-1. $C(t_{i-1},t_i)$ adalah jumlah tag ke i-1 diikuti dengan tag ke i dan $C(t_{-1})$ adalah jumlah kemunculan tag ke i-1 pada data latih.

Persamaan 2.2 untuk menghitung *emision probability*:

$$P(w_i|t_i) = \frac{C(t_i,w_i)}{C(t_i)} \quad (2.2)$$

$P(w_i|t_i)$ merupakan nilai dari *emission probability* atau probabilitas kata ke i yang memiliki tag ke i, C(t_i,wi) adalah jumlah kemunculan kata ke i dengan tag ke i pada data latih dan C(t_i) adalah jumlah kemunculan tag ke i pada data latih.

Diagram HMM menurut (Chopra, et al., 2012):



Gambar 1.4 Diagram HMM

Sumber: Chopra (2012)

1.6 Algoritme Viterbi

Viterbi merupakan pemrograman dinamis yang bekerja seperti algoritme *forward*. Algoritme Viterbi juga sangat menyerupai varian pemrograman dinamis lainnya, seperti algoritme *minimum edit distance*. Algoritme Viterbi memiliki satu komponen yang tidak dimiliki Algoritme *forward* yaitu *backpointers*. Alasannya adalah bahwa algoritme *forward* perlu menghasilkan kemungkinan observasi, algoritme Viterbi harus menghasilkan probabilitas dan juga urutan kondisi yang paling baik. Perhitungan urutan kondisi terbaik ini dengan mencatat jalur *hidden state* yang menyebabkan masing-masing kondisi. Proses yang dilakukan dalam algoritme Viterbi ini ialah dengan mencari nilai *tag* optimum untuk suatu kata. Proses ini dilakukan dengan mencari nilai maksimum dari hasil perkalian nilai *transition probability* dan *emission probability* yang telah didapatkan pada pemodelan *Hidden Markov Model.* Algoritme Viterbi ini dilakukan secara rekursif sebanyak kata yang akan dikenali pada data uji. Seperti pada Persamaan 2.3 (Jurafsky & Martin, 2017):

$$\boldsymbol{V_t(j) = \max_{i=1}^{N} V_{t-1}(i) a_{ij} b_j(o_t)} \qquad (2.3)$$

Keterangan:

$V_{t-1}(i)$: probabilitas jalur Viterbi sebelumnya dari langkah waktu sebelumnya

a_{ij} : probabilitas transisi dari kondisi sebelumnya a_i ke kondisi sekarang a_j

$b_j(o_t)$: kemungkinan kondisi pengamatan dari o_t berdasarkan kondisi j sekarang

1.7 Additive Smoothing

Secara umum metode yang digunakan untuk *smoothing* dapat dibagi menjadi dua kategori. Pada kategori pertama, disebut sebagai *smoothing proper*, dimana parameter model tunggal disesuaikan untuk mengatasi kelangkaan data, biasanya dengan mengambil beberapa kemungkinan massa dari kejadian yang terlihat dan menggunakannya untuk

kejadian yang tak terlihat. Kategori ini mencakup metode seperti *Additive Smoothing*, *Good-Turing estimation*, dan berbagai metode berdasarkan data dan validasi silang. Dalam kategori kedua, disebut dengan *combinatory smoothing*, metode untuk menggabungkan perkiraan dari beberapa model. Metode yang paling terkenal dalam kategori ini adalah *back-o-smoothing* dan *linear interpolation* (Nivre, 2000).

Mungkin metode *smoothing* yang paling sederhana adalah *Additive Smoothing* terdiri dari menambahkan konstanta c untuk semua frekuensi (termasuk frekuensi nol dari kata yang tidak terlihat) dan kemudian menghasilkan estimasi *maximum likelihood* yang baru. Bergantung pada nilai c, metode ini mempunyai nama yang berbeda. Untuk c yang bernilai 1 dikenal sebagai *Laplace* dan c yang yang bernilai 0.5 disebut sebagai *Lidstone* atau *Expected Likelihood Estimation* (Nivre, 2000).

Dengan menggunakan *tagged training corpus*, kita dapat menemukan estimasi probabilitas maksimum dengan menghitung kejadian. Persamaan 2.4 menunjukan persamaan estimasi *maximum-likelihood* (Haulrich, 2009):

$$\boldsymbol{a_{km}} = \frac{|\boldsymbol{q_k q_m}|}{|\boldsymbol{q_k}|} \tag{2.4}$$

Menggunakan estimasi *maximum likelihood* dapat menyebabkan masalah karena transisi yang tidak terlihat diberi probabilitas nol. Untuk Menghindari hal tersebut menggunakan penambahan *smoothing* dalam perhitungan pada *Hidden Markov Model* ketika menghitung probabilitas. Penerapan *smoothing* pada *transition probability* dapat dilakukan menggunakan Persamaan 2.5.

$$\boldsymbol{a_{km}} = \frac{|\boldsymbol{q_k q_m}|+\boldsymbol{C}}{|\boldsymbol{q_k}|+|\boldsymbol{QA}|} \tag{2.5}$$

q_kq_m merupakan nilai *transition probability* dari kondisi q_k ke kondisi q_m dan C merupakan nilai *additive smoothing,* dalam penelitian ini menggunakan *lidstone* yaitu 0.5 sedangkan QA merupakan jumlah dari kemungkinan POS-tag. Penerapan *smoothing* pada *emission probability* dapat dilakukan menggunakan Persamaan 2.6.

$$\boldsymbol{b_i(o_t)} = \frac{|\boldsymbol{o_t, q_i}|+\boldsymbol{C}}{|\boldsymbol{q_i}|+|\boldsymbol{V}|} \tag{2.6}$$

$b_i(o_t)$ adalah nilai *emission probability*, dengan o_t adalah kata ke t dan q_i adalah kondisi ke i dan C merupakan nilai *additive smoothing* sedangkan nilai V adalah jumlah dari keseluruhan kata yang berbeda dari data latih.

1.8 Rule Template

Metode *Rule Template* bekerja dengan cara mengatur *rule* agar kandidatnya dapat dikenali. Entitas yang relevan teridentifikasi oleh setiap *rule*. Misalnya, perusahaan Nokia memiliki seri ponsel yang diberi nama 'N#' dimana '#' mewakili sebuah nomor, seperti 'N97'. Terdapat beberapa contoh *rule* yang dapat digunakan seperti *Special Words*, *Semantics Pattern* dan *General List*. *Special Words* merupakan nama unik atau kreatif yang diberikan oleh pabrik untuk hasil produksinya seperti iPhone, ThinkPad dan lain sebagainya. *Semantics Pattern* merupakan pola kebiasaan masyarakat dalam memberi nama suatu produk, yaitu terdiri dari tiga pola diantaranya:

1. Sebuah nama produk selalu diikuti oleh kata milik, preposisi, atau kuantifier seperti my MacBook, the Xbox dan lain sebagainya.

2. Terdapat kalimat yang menyebutkan beberapa produk dan mengandung preposisi '*for'* atau 'untuk' di dalamnya memiliki nilai probabilitas yang tinggi untuk menjadi kandidat nama produk, baik itu sesudah atau sebelum. Contohnya, Seidio Inocase 360 untuk BlackBerry Curve 8900.

General List merupakan metode yang digunakan untuk mengidentifikasi nama produk dengan cara mengumpulkan beberapa kategori kata yang jarang digunakan dalam pemberian nama produk. Dalam mengidentifikasi nama produk dengan tujuan *Rule Template*, model yang digunakan bersifat *cascade.* Setiap *rule* terdiri atas *classifier* dan seluruh model yang berdasarkan rangkaian dari semua *classifier*. Semua kata yang telah diklasifikasikan secara benar oleh *cascade rule* adalah simbol dari beberapa produk (Wu, et al., 2012).

1.9 Regular Expression

Salah satu keberhasilan dalam standardisasi dalam ilmu komputer adalah *regular expression* (RE), sebuah bahasa untuk menentukan string pencarian teks. Implementasi ini digunakan dalam setiap bahasa komputer, pengolah kata, dan alat pengolah teks seperti alat Unix grep atau Emacs. Secara formal, *regular expression* adalah notasi aljabar untuk menandai satu set string yang sangat berguna untuk mencari teks, ketika memiliki pola untuk mencari dan korpus teks untuk dicari. Fungsi pencarian *regular expression* akan mencari melalui korpus, mengembalikan semua teks yang sesuai dengan polanya. Korpus bisa jadi satu koleksi dokumen. Contoh, mengambil *regular expression* dan mengembalikan setiap baris dokumen masukan yang sesuai dengan ungkapan (Jurafsky & Martin, 2017).

Jenis *regular expression* yang paling sederhana adalah urutan karakter sederhana. Untuk mencari 'woodchuck', kita mengetik '/woodchuck'. *Regular expression* bersifat *case sensitive*, huruf kecil '/s/' berbeda dari huruf besar '/S/' (/s/ cocok dengan huruf kecil tapi bukan huruf besar S). Artinya pola '/woodchucks/' tidak akan cocok dengan *string* 'Woodchucks'. Masalah ini dapat diselesaikan dengan penggunaan kurung siku []. Karakter di dalam kurung siku menentukan karakter yang tidak sesuai untuk disesuaikan. Tabel 2.3 menunjukan contoh penggunaan *Regular Expression* (Jurafsky & Martin, 2017).

Tabel 1.3 Contoh Reguler Expression

RE	match	Contoh Kalimat
/[wW]oodchuck/	Woodchuck atau woodchuck	"*wood* Woodchuck"
/[0-9]/	Satu angka	"*Chapter* 1: *Down the Rabbit Hole*"
/[A-Z]/	Satu huruh besar	"*we should call it 'Drenched Blossoms'"*
/[1234567890]/	Semua angka	"plenty of 7 to 5"

Sumber: diadaptasi dari Jurafsky (2017)

1.10 Pengukuran Evaluasi

Untuk mengetahui *performance* hasil klasifikasi *Named Entity Recognition* (NER), diperlukan sebuah teknik untuk pengukuran evaluasi. Pengukuran ini dilakukan menggunakan *confusion matrix* dan *accuracy,* dimana *confusion matrix* sangat penting karena menunjukkan

kinerja NER berdasarkan sistem dalam hal *Precision*, *Recall*, dan *F-Measure*. Berikut ini adalah pengukuran *confusion matrix* dan *accuracy* menurut (Roman & Christoph, 2009):

$$\text{Precision, } (\text{P}) = \frac{\text{TP}}{\text{TP+FP}} \quad (2.7)$$

$$\text{Recall, } (\text{R}) = \frac{\text{TP}}{\text{TP+FN}} \quad (2.8)$$

$$\text{F} - \text{Measure} = \frac{2*\text{P}*\text{R}}{\text{P+R}} \quad (2.9)$$

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+FP+TN+FN}} \quad (2.10)$$

Tabel 1.4 Tabel Confusion Matrix

		Nilai Sebenarnya	
		TRUE	FALSE
Nilai Prediksi	TRUE	TP	FP
	FALSE	FN	TN

Sumber: diadaptasi dari Roman (2009)

Dimana:

1. TP, merupakan True Positive.
2. FP, merupakan False Positive.
3. FN, merupakan False Negative.
4. TN, merupakan True Negative.