BAB II

TINJAUAN PUSTAKA

2.1. Penelitian Terdahulu

Pemrosesan bahasa alami atau *Natural Language Processing* adalah cabang ilmu komputer dan linguistik yang mengkaji interaksi antara komputer dengan bahasa manusia (bahasa alami). *NLP* sering dianggap sebagai cabang dari kecerdasan buatan dan bidang kajiannya bersinggungan dengan linguistik komputasional. Kajian *NLP* antara lain mencakup segmentasi tuturan (*speech segmentation*), segmentasi teks (*text segmentation*), penandaan kelas kata (*part-of-speech*), serta pengawataksaan makna (*word sense disambiguitas*). Meskipun kajiannya dapat mencakup teks dan tuturan, pemrosesan tuturan (*speech processing*) telah berkembang menjadi suatu bidang kajian yang terpisah. Sejarah NLP dimulai tahun 1950-an, meskipun telah ada penelitian NLP pada tahun-tahun sebelumnya. Pada tahun 1950, Alan Turing (bapak ilmu komputer) mempublikasikan artikel terkenalnya yang berjudul "*Computing Machinery and Intelligence*" yang didalamnya Alan Turing mengusulkan tes yang sekarang disebut dengan *Turing Test*. *Turing Test* adalah sebuah tes yang mengukur kemampuan mesin (dalam hal ini program komputer) untuk menunjukkan perilaku cerdas. Dalam ilustrasi contoh aslinya, seorang juri manusia akan terlibat dalam percakapan dengan manusia dan mesin yang akan dites. Semua peserta dipisahkan satu sama lain. Jika juri tidak bisa membedakan antara manusia dan mesin, maka mesin tersebut dikatakan lulus tes.

Beberapa penelitian sebelumnya yang menjadi referensi atau pembanding oleh penulis dalam melakukan penelitian ini dijelaskan dalam BAB ini. Ada beberapa penelitian yang telah dilakukan sebelumnya, seperti yang dilakukan oleh Pritendra Kumar Malakar & Pravesh Kumar Dwivedi & Aarti Kashyap (2015) yang berjudul *Sentiment Classification of Hindi Language using Natural Language Processing Techniques*. Penelitian ini menyajikan pendekatan *hybrid* untuk secara otomatis menentukan sentimen atau ungkapan dari teks bahasa Hindi

dengan menggunakan leksikon hindi dan mmembuat kalimat tersebut menjadi polaritas yang berbeda yaitu positif, negatif dan netral. Penelitian ini melakukan klasifikasi sentimen kalimat bahasa Hindi dengan menggunakan dataset 1000 kalimat, 500 kalimat positif dan 500 kalimat negatif. Setelah melakukan klasifikasi, sistem memberikan hasil 70 % benar.

Penelitian lain yang berkaitan dengan *Machine Learning* dilakukan oleh Mohini Chaudhari & Sharvari Govilkar (2015) yang melakukan survey teknik *Machine Learning for Sentiment Classification*. Tujuan melakukan survey tersebut adalah mendapatkan akurasi dari setiap tehnik yang digunakan untuk mengklasifikasi sentimen kalimat. Tehnik yang digunakan yaitu *Naive Bayes, Support Vector Machine (SVM), Multi Layer Perceptron, Maximum Entropy.*

Pierre Ficamos & Yan Liu (2016) melakukan penelitian tentang sentimen kalimat yang mengambil data dari Twitter. Data berjumlah 140 kalimat, dimana setiap kalimat diuji dengan metode *Support Vector Machine (SVM).* Dari hasil pengujian ditemukan akurasi 74,09 %. Jika ukuran dataset pelatihan meningkat secara maksimal, metode ini dapat memberikan akurasi 81,34 % yang konsisten.

Penelitian yang dilakukan Onam Bharti & Mrs. Monika Malhotra (2016) tentang *Sentiment Analysis* yang menggunakan metode *Naive Bayes, KKN,* dan *K-Means.* Penelitian ini menggunakan dataset 500 *mobile reviews* dan di temukan akurasi yang berbeda-beda dari setiap metode. Dengan metode *Naive Bayes* ditemukan akurasi sejumlah 79,66 %, metode *KKN* ditemukan akurasi sejumlah 83,59 %. Penelitian ini juga melakukan modifikasi antara metode *K-Means + Naive Bayes* ditemukan 89 % akurasi, metode *K-Means + NB + KKN* ditemukan 92 % akurasi. Dari sudut pandang MKM, *naive bayes* dan *KKN* paling sesuai untuk klasifikasi teks dan interpretasi sosial.

Preety & Sunny Dahiya (2015) melakukan penelitian yang menggunakan algoritma Naive bayes, *SVM*, dan *Naive bayes + Modified K-Means*. Dari hasil pengujian terhadap 500 mobile dataset ditemukan akurasi *naive bayes* 79,66 % akurasi, metode SVM 83,59 % akurasi, dan NB + *modified K-Means* didapat 89 % akurasi.

Penelitian lain dilakukan oleh (Goldberg, 2016) mengenai penggunaan Neural Network dalam Natural Language Processing. Penelitian ini bertujuan untuk membantu praktisi NLP dalam menggunakan dan memahami model Neural Network diimplementasikan pada Natural Language Processing.

Sebuah Penelitian yang dilakukan oleh (Kiperwasser & Goldberg, 2016) mengenai skema dependency parsing menggunakan bidirectional LSTM. Pada penelitian ini dilakukan dengan tujuan untuk menghasilkan sistem yang sederhana dan efektif untuk melakukan parsing pada suatu kalimat.

Ira Zulfa & Edi Winarko (2017) mengenai sentimen analisis tweet berbahasa Indonesia dengan Deep Belief Network. Penelitian ini bertujuan untuk melakukan klasifikasi terhadap sentimen positif, negatif dan netral terhadap data yang di uji dan untuk mengetahui akurasi model klasifikasi dengan menggunakan metode Deep Belief Network ketika di aplikasikan pada klasifikasi tweet yang berjumlah 2023 tweet untuk menandai kelas sentimen data training berbahasa Indonesia. Dari hasil penelitian ini diperoleh akurasi sebesar 93,31 % ketika dibandingkan dengan metode naive bayes yang memiliki akurasi sebesar 79,10% dan SVM (*Support Vector Machine*) yang memiliki akurasi sebesar 92,18%.

Penelitian lain yang berkaitan dengan sentimen analisis dilakukan oleh Saeful Bahri, Rizal Amegia Putra & Rusda Wajhillah (2017) mengenai Analisa Sentimen Berbasis Natural Language Processing dengan Naive Bayes Clasifier. Penelitian ini menggunakan NLP bertujuan untuk mereduksi dimensi korpus tweet yang besar dan hanya mengambil kata yang dianggap sebagai akar dari kata yang terdapat pada korpus yang diteliti sehingga menjadi korpus yang mengandung sedikit kata yang kurang mengandung makna.

Dari Penelitian yang telah dilakukan maka dibangunlah model *Natural Language Processing text classification* Bahasa Indonesia menggunakan *library spacy*. Melalui model yang dibuat diharapkan dapat menambah sumber daya *NLP* Bahasa Indonesia

2.2. Landasan Teori

2.3.1. Text Classification

Text Classification adalah penelitian dari *text mining* yang menganalisa opini, sentimen, evaluasi, penilaian, sikap, dan emosi. Setiap argumen-argumen yang dituangkan ke dalam tulisan tentunya memiliki sentimen tersendiri dalam kalimat tersebut. (Liu, 2012) menuturkan bahwa suatu entitas seperti produk, layanan, organisasi, individu, isu, kejadian, topik dan yang dinilai oleh masyarakat dalam bentuk tulisan mengandung emosi tertentu. Terdapat tiga buah subproses dari *opinion mining* yaitu *document subjectivity, opinion orientation,* dan *target detection*. Dalam dunia bisnis *opinion mining* banyak digunakan untuk menganalisis secara otomatis opini pelanggan tentang produk dan pelayanannya.

2.3.2. Python

Python merupakan Bahasa pemrograman tingkat tinggi multiguna yang dapat digunakan untuk mengembangkan berbagai macam keperluan perangkat lunak. *Python* memiliki berbagai macam paradigma, utamanya namun tidak terbatas pada pemrograman berorientasi objek. Kode pada *Python* dirancang untuk memiliki tingkat keterbacaan yang tinggi (*readability*). Python juga memiliki *standard library* yang mendukung berbagai macam tugas pemrograman (Lutz, 2013). Ukuran kode *Python* secara umum hanya 1/3 hingga 1/5 kode yang dibuat dengan Bahasa pemrograman *C++* atau *Java*. *Python* secara otomatis mengalokasikan memori untuk objek dan menghapusnya ketika objek sudah tidak digunakan.

2.3.3. Natural Language Processing

Natural Language Processing merupakan cabang dari ilmu komputer yang berkembang dari studi Bahasa dan komputasi linguistik dalam cabang kecerdasan buatan. Tujuan dari *NLP* adalah untuk

mengembangkan dan merancang aplikasi yang mampu menjadi penengah antar interaksi manusia dan mesin dengan perangkat lain melalui penggunaan Bahasa alami (Pustejovsky & Stubbs, 2012). Sebelum perancangan model, diperlukan beberapa tujuan pembuatan model seperti untuk apa model dibuat, corpus atau dataset seperti apa yang diperlukan, dan tujuan apa yang perlu dicapai dalam pembuatan model. Dalam pengumpulan dataset diperlukan beberapa teknik *text pre-processing* agar dataset yang tidak terstruktur memiliki representasi yang terstruktur. Beberapa proses *pre-processing* adalah:

1. *Tokenization*

   Merupakan proses pemotongan string menjadi teks – teks independen yang memiliki makna. Proses tokenisasi pada suatu teks akan memotong teks tersebut menjadi beberapa kalimat atau kata (Sanger & Feldman, 2007).
2. *Part-of-Speech Tagging*

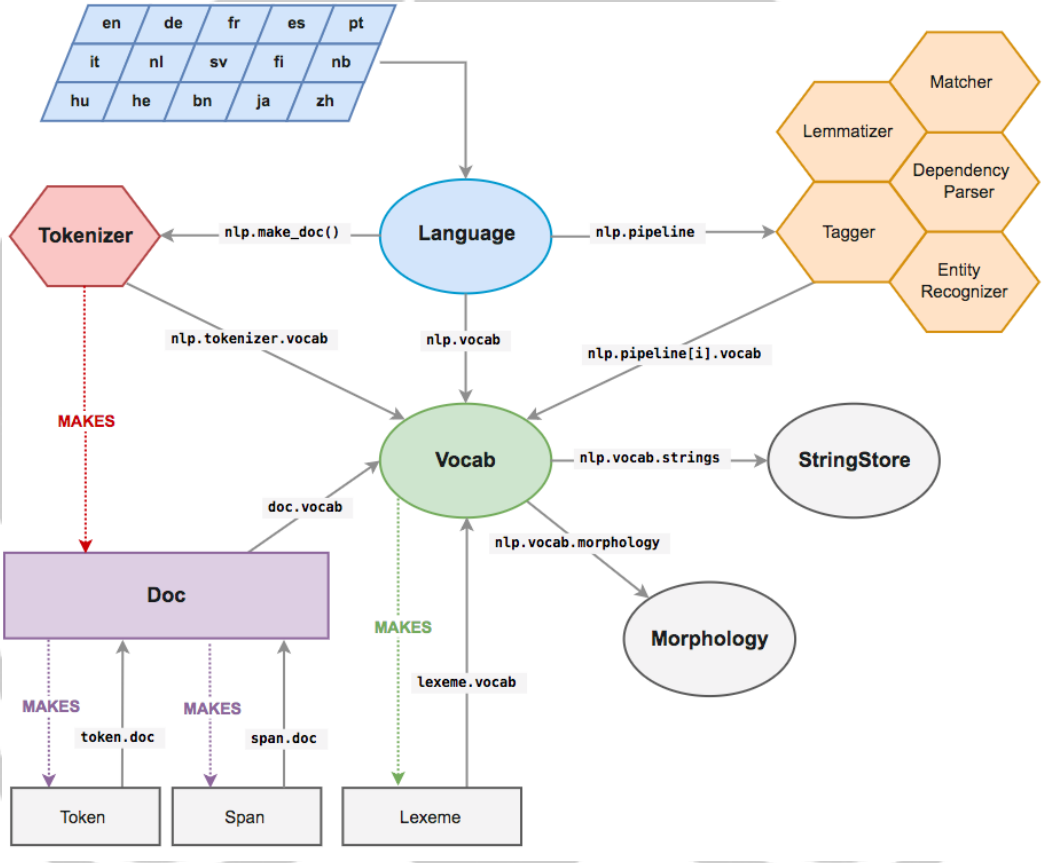
   Merupakan proses pemberian *POS Tag* pada suatu kalimat atau kata. *POS Tagging* dapat dilakukan dengan memberi tiap kata tag seperti kata kerja, kata benda, kata sifat dan aturan grammar lain (Sanger & Feldman, 2007).
3. *Syntactical Parsing*

   Merupakan proses analisis sintaks suatu kalimat berdasarkan teori grammar tertentu. Proses parsing yang umum dapat dibagi dua yaitu *dependency dan constituency* (Sanger & Feldman, 2007).

2.3.4. Arsitektur Spacy

Spacy merupakan *library open source* yang dapat digunakan oleh para peneliti untuk mengembangkan NLP dalam berbagai bahasa diantaranya *matcher, lemmatizer, deppendency parser, text classification,* dan *entity recogizer.* Struktur utama dari *Spacy* ada pada Doc dan Vocab. Objek *Doc* memiliki urutan token dan memuat semua anotasi, sedangkan objek *Vocab* memiliki satu set tabel pencarian yang memuat informasi

umum pada setiap dokumen. Dengan adanya pemusatan data *string* dapat menghindari banyaknya data yang sama yang dapat membuat model menjadi terlalu besar dan memakan banyak memori, pemusatan ini juga bertujuan agar data yang benar hanya memiliki satu sumber yang benar.



Gambar 2. 1 Arsitektur Spacy (https://spacy.io/api/, 1 Juli 2018)

Keterangan Gambar 2.1

1.	**Doc**	**:**	Tempat untuk mengakses anotasi *linguistic*
2.	**Span**	:	Bagian dari objek Doc
3.	Token	:	Token Individu, terdiri dari simbol,

			tanda baca, spasi, dll.
4.	Lemex	:	Tambahan dalam kosakata, memuat kata tanpa konteks yang bukan termasuk token.
5.	Language	:	Sebuah *pipline* pemrosesan teks yang menentukan bahasa yang akan digunakan
6.	Pipe	:	Kelas dasar untuk melakukan pemrosesan *pipeline*
7.	Text classification	:	Berisi anotasi label *Part of Speech* pada objek Doc
8.	DependecyParser	:	Anotasi *dependecy* sintak pada objek Doc
9.	EntityRecognizer	:	Anotasi bernama, seperti orang atau produk pada objek
10.	TextCategorizer	:	Penetapan kategori atau label pada objek Doc
11.	Tokenizer	:	Segmen teks, dan membuat objek Doc dengan batasan segmen yang ditemukan
12.	Lemmatizer	:	Bentuk dasar dari kata
13.	Morphology	:	Menetapkan fitur *linguistic* seperti lemmas, kata benda, dll berasarkan label *Part of Speech.*
14.	Matcher	:	Urutan token berdasarkan polar seperti *ekspresireguler*
15.	PharseMatcher	:	Urutan token berdasarkan frasa

16.	Vocab	:	Tabel pencarian untuk kosa kata yang memungkin untuk mengakses objek *lexeme*
17.	StringStore	:	Penampung *string* peta dari dan ke nilai *hash*
18.	Vectors	:	Kelas *container* untuk data vektor yang dikunci oleh *string*
19.	GoldParse	:	Koleksi untuk anotasi pelatihan
20.	Goldcorpus	:	Korpus beranotasi dengan format *Json*. Tempat mengelola anotasi *Tagging, Dependecy Parsing,* dan *NER*