PART OF SPEECH TAGGING BAHASA INDONESIA MENGGUNAKAN ALGORITMA *DECISION TREE*

Skripsi

untuk memenuhi sebagian persyaratan

mencapai derajat Sarjana S1

Program Studi Teknik Informatika

Disusun oleh:

Ahmad Ardiyanto

16650061

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS ISLAM NEGERI SUNAN KALIJAGA

YOGYAKARTA

2020

HALAMAN PENGESAHAN

KEMENTERIAN AGAMA
UNIVERSITAS ISLAM NEGERI SUNAN KALIJAGA
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Marsda Adisucipto Telp. (0274) 540971 Fax. (0274) 519739 Yogyakarta 55281

PENGESAHAN TUGAS AKHIR

Nomor : B-1691/Un.02/DST/PP.00.9/07/2020

Tugas Akhir dengan judul : PART OF SPEECH TAGGING BAHASA INDONESIA MENGGUNAKAN ALGORITMA DECISION TREE

yang dipersiapkan dan disusun oleh:

Nama : AHMAD ARDIYANTO
Nomor Induk Mahasiswa : 16650061
Telah diujikan pada : Jumat, 17 Juli 2020
Nilai ujian Tugas Akhir : A-

dinyatakan telah diterima oleh Fakultas Sains dan Teknologi UIN Sunan Kalijaga Yogyakarta

TIM UJIAN TUGAS AKHIR

Ketua Sidang/Penguji I

Muhammad Taufiq Nuruzzaman, S.T. M.Eng., Ph.D.
SIGNED

Valid ID: 5f2d0080156dd

Penguji II

Dr. Shofwatul 'Uyun, S.T., M.Kom.
SIGNED

Valid ID: 5f2a31313312c

Penguji III

Maria Ulfah Siregar, S.Kom. MIT., Ph.D.
SIGNED

Valid ID: 5f2bcba158d03

Yogyakarta, 17 Juli 2020
UIN Sunan Kalijaga
Plt. Dekan Fakultas Sains dan Teknologi

Dr. Murtono, M.Si.
SIGNED

Valid ID: 5f31586553dab

SURAT PERSETUJUAN SKRIPSI

SURAT PERSETUJUAN SKRIPSI/TUGAS AKHIR

Hal : Persetujuan Skripsi
Lamp :

Kepada
Yth. Dekan Fakultas Sains dan Teknologi
UIN Sunan Kalijaga Yogyakarta
di Yogyakarta

Assalamu'alaikum wr. wb.

Setelah membaca, meneliti, memberikan petunjuk dan mengoreksi serta mengadakan perbaikan seperlunya, maka kami selaku pembimbing berpendapat bahwa skripsi Saudara:

Nama : Ahmad Ardiyanto
NIM : 16650061
Judul Skripsi : Part Of Speech Tagging Bahasa Indonesia Menggunakan Algoritma Decision Tree

Sudah dapat diajukan kembali kepada Program Studi Teknik Informatika Fakultas Sains dan Teknologi UIN Sunan Kalijaga Yogyakarta sebagai salah satu syarat untuk memperoleh gelar Sarjana Strata Satu dalam Program Studi Teknik Informatika

Dengan ini kami mengharap agar skripsi/tugas akhir Saudara tersebut di atas dapat segera dimunaqsyahkan. Atas perhatiannya kami ucapkan terima kasih.

Wassalamu'alaikum wr. wb.

Yogyakarta, 06 Juli 2020
Pembimbing

M. Taufiq Nuruzzaman, S.T., M.Eng.
NIP. 19791118 200501 1 003

PERNYATAAN KEASLIAN SKRIPSI

Saya yang bertanda tangan dibawah ini:

Nama	: Ahmad Ardiyanto
NIM	: 16650061
Program Studi	: Teknik Informatika
Fakultas	: Sains dan Teknologi

Menyatakan bahwa skripsi saya yang berjudul **"Part Of Speech Tagging Bahasa Indonesia Menggunakan Algoritma Decision Tree"** merupakan hasil penelitian saya sendiri, tidak terdapat pada karya yang pernah di ajukan untuk memperoleh gelar kesarjana di suatu perguruan tinggi, dan bukan plagiasi karya orang lain kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Yogyakarta, 06 Juli 2020

Ahmad Ardiyanto
NIM. 16650061

KATA PENGANTAR

Puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan penelitian yang berjudul ***Part Of Speech Tagging* Bahasa Indonesia Menggunakan Algoritma *Decision Tree*** sebagai salah satu syarat untuk mencapai gelar sarjana program studi Teknik Informatika Universitas Islam Negeri Sunan Kalijaga Yogyakarta. Sholawat serta salam selalu tercurahkan kepada junjungan kita Nabi Muhammad SAW. beserta seluruh keluarga dan sahabat beliau.

Penulis menyadari bahwa apa yang dilakukan dalam penyusunan laporan penelitian ini masih terlalu jauh dari kesempurnaan. Oleh karena itu, penulis mengharapkan kritik dan saran yang berguna dalam penyempurnaan analisis ini di masa yang akan datang. Semoga apa yang telah penulis lakukan dapat bermanfaat bagi pembaca.

Tidak lupa penulis juga mengucapkan terima kasih kepada pihak-pihak yang telah membantu dalam penyelesaian tugas akhir ini, baik secara langsung maupun tidak langsung. Ucapan terima kasih penulis sampaikan kepada:

1. Bapak Prof. Dr. Phil. Al Makin, M.A., selaku Rektor UIN Sunan Kalijaga.
2. Bapak Dr. H. Waryono, M.Ag., selaku Wakil Rektor Bidang Kemahasiswaan dan Kerjasama UIN Sunan Kalijaga.

3. Bapak Dr. Murtono, M.Si., selaku Plt. Dekan Fakultas Sains dan Teknologi UIN Sunan Kalijaga.
4. Bapak Sumarsono, S.T., M.Kom., selaku Ketua Program Studi Teknik Informatika UIN Sunan Kalijaga.
5. Bapak M. Taufiq Nuruzzaman, S.T., M.Eng., Ph.D., selaku dosen pembimbing skripsi yang telah sabar dan meluangkan waktunya untuk memberikan motivasi, koreksi dan kritik saran dalam penyusunan skripsi.
6. Bapak Muhammad Didik Rohmad Wahyudi, S.T., MT., selaku Dosen Pembimbing Akademik.
7. Bapak Dr. Agung Fatwanto, S.Si., M.Kom., Bapak Agus Mulyanto, S.Si., M.Kom., Ph.D., Bapak Aulia Faqih Rifa'i, M.Kom., Ibu Maria Ulfah Siregar, S.Kom. MIT., Ph.D., Bapak Nurochman, S.Kom., M.Kom., Bapak Rahmat Hidayat, S.Kom., M.Cs., Ibu Dr. Shofwatul 'Uyun, S.T., M.Kom., selaku dosen pengampu mata kuliah program studi Teknik Informatika UIN Sunan Kalijaga yang telah banyak membantu sehingga penulis dapat menyusun tugas akhir.
8. Seluruh staf dan karyawan Fakultas Sains dan Teknologi UIN Sunan Kalijaga.
9. Kedua orang tua saya, ibu Kotimah dan bapak Heryanto yang selalu memberikan doa, perhatian, kasih sayang dan segala dukungan yang tak terhingga.

10. Adik saya, Tiara Damayanti yang selalu memberikan doa, kasih sayang dan semangat kepada penulis selama mengerjakan skripsi ini.
11. Seluruh Keluarga Besar Simbah Zainudin dan Simbah Giman yang senantiasa mendoakan dan memberikan dukungan kepada penulis.
12. Muhammad Dzulfikar Fauzi S.Kom., M.Cs., dan Usfita Kiftiyani M.Sc., selaku mentor dalam pengerjaan skripsi.
13. Sahabat-sahabat saya Ari, Ayyub, Amrul, Adam, Azmi, Aat, Hendra, Nur, Dio, Husni, Sekar, Lina, Lia, Ulfa, Yayang, Nida yang telah senantiasa menemani penulis serta memberikan semangat, dorongan, dan inspirasi dalam proses pengerjaan skripsi ini.
14. Teman-teman yang pernah menjadi satu kelompok selama kuliah dan seluruh teman-teman Teknik Informatika 2016 yang tidak dapat penulis sebutkan satu per satu.
15. Teman-teman Kontrakan Lur, HMKK dan Squad SWAG yang telah memberikan warna hari-hari penulis selama kuliah.
16. Teman-teman Fasilitator dan Instruktur ITTC UIN Sunan Kalijaga.
17. Teman-teman KKN Kelompok 36 Angkatan 99, Bapak Kepala Dusun dan seluruh warga Dusun Pancar.
18. Serta semua pihak yang tidak dapat penulis sebutkan satu persatu dan telah memberikan banyak doa dan dukungan sehingga penelitian ini dapat terselesaikan.

Semoga dukungan dan bantuan yang telah diberikan kepada penulis, menjadi amal baik dan mendapatkan pahala dari Allah SWT. Akhir kata penulis ucapkan terima kasih.

Yogyakarta, 30 Juni 2020

Penulis

HALAMAN PERSEMBAHAN

Skripsi ini penulis persembahkan

untuk kedua orang tua saya

Ibu Kotimah dan Bapak Heryanto.

Terima kasih untuk segalanya.

HALAMAN MOTTO

وَاَنْ لَّيْسَ لِلْاِنْسَانِ اِلَّا مَا سَعٰى

"dan bahwa manusia hanya memperoleh apa yang telah diusahakannya"

(Q.S An-Najm 53:39)

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR RUMUS

***Part Of Speech Tagging* Bahasa Indonesia**

Menggunakan Algoritma *Decision Tree*

Ahmad Ardiyanto

16650061

INTISARI

Penelitian ini bertujuan untuk menerapkan dan menganalisis *Part of Speech Tagging* (*POS Tagging*) bahasa Indonesia menggunakan algoritma *Decision Tree.* Proses *POS Tagging* dapat menggunakan kamus bahasa sebagai rujukannya atau membantu dalam pemberian kelas kata (*tag*), namun ada beberapa kendala seperti penentuan kelas kata untuk kata ambigu dan kata yang tidak ada di kamus, maka diperlukan pendekatan *POS Tagging* lebih lanjut untuk menangani masalah tersebut.

Penelitian ini menggunakan data Kamus Besar Bahasa Indonesia (KBBI) dan data berita online CNN Indonesia. Data berita tersebut kemudian dilakukan proses *preprocessing*, ekstraksi fitur, dan *vectorization* sehingga menghasilkan *dataset* sebanyak 40.071 yang kemudian digunakan sebagai data latih untuk membangun model *Decision Tree*.

Pengujian *POS Tagging* pada penelitian ini mendapatkan akurasi sebesar 95,3%, serta persentase keberhasilan 90,0% dalam memberikan *tag* untuk kata yang tidak ada di KBBI dan keberhasilan sebesar 91,9% untuk memberikan *tag* pada kata ambigu. Hasil implementsi aplikasi *POS Tagging* terhadap kalimat berbahasa indonesia sudah baik meskipun belum maksimal.

Kata kunci : *part of speech tagging, decision tree, bahasa indonesia*

Part Of Speech Tagging on Indonesian Language

Using Decision Tree Algorithm

Ahmad Ardiyanto

16650061

ABSTRACT

This study aims to apply and analyze Indonesian Part of Speech Tagging (POS Tagging) using the Decision Tree algorithm. The POS Tagging process can use language dictionaries as a reference or assist in the provision of word classes (tags), but there are some constraints such as the determination of word classes for ambiguous words and words that are not in the dictionary, so a further POS Tagging approach is needed to overcome this problem.

This study uses Indonesian Language Dictionary (KBBI) data and Indonesian CNN online news data. The news data is then carried out the process of preprocessing, feature extraction, and vectorization so as to produce a dataset of 40,071 which are then used as training data to build the Decision Tree model.

POS Tagging testing in this study obtained an accuracy of 95.3%, and a percentage of success of 90.0% in tagging words that were not on KBBI and success of 91.9% for tagging ambiguous words. The results of the implementation of the POS Tagging application for Indonesian sentences are already good although not yet at maximum.

Keywords: part of speech tagging, decision tree, indonesian language

BAB I

PENDAHULUAN

1.1. Latar Belakang

Negara Indonesia menggunakan bahasa Indonesia sebagai bahasa resminya, selain itu bahasa Indonesia merupakan identitas bangsa serta lambang kebanggaan nasional, yang secara luas dan umum digunakan sebagai alat komunikasi oleh 222 juta orang (Ramadhanti et al., 2019). Banyaknya jumlah pengguna maka sangat diperlukan pembelajaran atau penelitian untuk memahami lebih jauh tentang penggunaan bahasa Indonesia, dalam memahami dan mempelajari Bahasa Indonesia dapat mengunakan rujukan keilmuan seperti Kamus Besar Bahasa Indonesia (KBBI) dan Tata Bahasa Baku Bahasa Indonesia (TBBBI). Di era modern ini, terdapat salah satu bidang keilmuan di *Computer Sience* yang mempelajari tentang kebahasaan atau bahasa alami dengan menggunakan kecerdasan buatan dan bantuan komputer yaitu *Natural Language Processing* (NLP), sehingga dalam pemrosesan pembelajaran atau pengetahuan bahasa alami dapat dilakukan dalam waktu yang lebih cepat. NLP terdiri dari berbagai proses salah satunya *Part of Speech Tagging (POS Tagging)*.

POS Tagging adalah upaya untuk memberikan label pada kata di dalam teks sesuai dengan kelas katanya dan juga kemungkinan fitur morfologinya, sehingga *POS Tagging* dapat memberikan informasi mengenai definisi dan konteks kata (Kamayani, 2019). Hasil penelitian *POS Tagging* pada dokumen dapat digunakan sebagai dasar penelitian dalam NLP lainnya, seperti: *Language Generator*,

Information Retrieval, Text Summarization, Question and Answering, dan *Machine Translation.*

Proses *POS Tagging* dapat menggunakan *lexicon* atau kamus bahasa yang dalam hal ini KBBI untuk digunakan sebagai rujukannya atau membantu dalam pemberian kelas kata (*tag*), namun tidak dapat sepenuhnya menggunakan kamus karena ada beberapa kendala seperti penentuan kelas kata untuk kata ambigu dan kata yang tidak ada di kamus, maka diperlukan pendekatan *POS Tagging* lebih lanjut untuk menyelesaikan masalah ini. *POS Tagging* juga dapat dilakukan dengan berbasis aturan (*rule based*) dan probabilitas (*probability-based*) dari sebuah model yang dibangun. *Rule based tagging* dilakukan dengan cara *top-down*, yaitu melakukan konsultasi dengan ahli linguistik untuk mendefinisikan aturan-aturan yang biasa digunakan manusia. *Probability based tagging* dilakukan dengan cara *bottom-up*, yaitu menggunakan korpus sebagai *training* data untuk menentukan secara probabilistik *tag* yang terbaik untuk sebuah kata dalam sebuah konteks (Sabloak et al., 2016).

POS Tagging berbasis aturan (*rule based*) lebih akurat dan lebih baik dalam menggambarkan fenomena linguistik (Marquez & Rodriguez, 1998) karena *rule based* ditulis dari sudut pandang linguistik dan secara eksplisit menggambarkan fenomena linguistik, namun *POS Tagging* dengan menggunakan *rule based* tentunya tidak mudah, terutama untuk bahasa yang lebih kompleks dalam aturan penulisan seperti Bahasa Indonesia, maka dibutuhkan usaha dan waktu yang lebih banyak dari pakar linguistik untuk membuat atau menetapkan aturan-aturan yang

akan diterapkan pada *rule based*. *POS Tagging* berbasis probabilitas menggunakan algoritma yang mempelajari dan membangun model dari *dataset* pelatihan dan menerapkan hasil pembelajarannya pada data yang diuji atau *instance* baru dengan sedikit keterlibatan manusia (untuk pelatihan dan menguji persiapan algoritma), sehingga meminimalkan biaya yang diperlukan untuk mengembangkan aturan klasifikasi tidak seperti penandaan *POS Tagging* dengan *rule based*. *Hidden Markov Model, Naive Bayes, Support Vector Machine* dan *Decision Tree* adalah beberapa algoritma penandaan berbasis probabilitas yang ada untuk penandaan *POS Tagging*.

Pada penelitian (Orphanos & Tsalidis, 1999) yang berjudul "*Combining Handcrafted And Corpus-Acquired Lexical Knowledge Into A Morphosyntactic Tagger*" menunjukkan aplikasi yang sukses dari implementasi algoritma *Decision Tree* yang diinduksi secara otomatis untuk masalah disambiguasi *POS Tagging* dan menebak kata yang tidak diketahui dari Bahasa Yunani Modern hingga mencapai kinerja sekitar 93,5%. Selain itu *Decision Tree* merupakan algoritma yang hasil modelnya dapat dipahami manusia dan dapat diperbaiki lebih lanjut tidak seperti penandaan statistik lainnya. Sehingga dapat diverifikasi apakah hasil model *Decision Tree* menangkap fenomena linguistik yang mendasarinya atau tidak (Orphanos & Christodoulakis, 1999). Saat ini juga telah dikembangkan *software package* atau *library* dari *Decision Tree* yang lebih mudah dalam penerapannya, diantaranya adalah *library machine learning scikit-learn* di Python (Pedregosa et al., 2011). Berdasarkan penjelasan diatas, penelitian ini bermaksud menerapkan algoritma *Decision Tree* dalam *Part-of-speech Tagging* Bahasa Indonesia.

1.2. Rumusan Masalah

Berdasarkan latar belakang di atas, maka rumusan masalah dalam penelitian ini adalah Bagaimana menerapkan algoritma *Decision Tree* dalam *Part-of-Speech Tagging* bahasa Indonesia, khususnya dalam mengatasi penentuan kelas kata pada kata ambigu dan kata yang tidak ada di kamus bahasa ?

1.3. Tujuan Penelitian

Penelitian ini bertujuan untuk menerapkan dan menganalisis *Part-of-speech Tagging* bahasa Indonesia menggunakan algoritma *Decision Tree.*

1.4. Batasan Penelitian

Berdasarkan rumusan masalah pada penelitian ini, maka dibutuhkan batasan masalah untuk menghindari pembahasan yang meluas, berikut beberapa batasan masalah pada penelitian ini, yaitu:

1.4.1. Algoritma *Decision Tree* menggunakan *library* Scikit-klearn.

1.4.2. *Lexicon* atau Kamus yang digunakan adalah Kamus Besar Bahasa Indonesia (KBBI) edisi IV tahun 2008 yang diterbitkan oleh Pusat Bahasa Departemen Pendidikan Nasional.

1.4.3. Kelas kata (*tagset*) yang digunakan mengacu pada *tagset* KBBI versi IV tahun 2008.

1.4.4. Korpus yang digunakan sebagai data latih adalah berita CNN Indonesia tanggal 1-20 Februari 2020 sebanyak 140 berita.

1.4.5. Kamus dan korpus dari sumber sebagaimana disebutkan dalam poin sebelumnya, diasumsikan sudah lengkap untuk digunakan.

1.4.6. Pemberian kelas kata dilakukan untuk setiap satu kata bukan gabungan kata.

1.4.7. Menggunakan bahasa pemrograman Python, Jupyter Notebook dan *Database Management System* MySQL.

1.5. Manfaat Penelitian

Berdasarkan latar belakang dan tujuan di atas, maka penelitian ini dapat memberikan manfaat sebagai berikut:

1.5.1. Penelitian ini menambah wawasan dan pengembangan di bidang NLP, khususnya di bagian *POS Tagging*.

1.5.2. Menambah *corpus* bahasa Indonesia yang dapat digunakan untuk penelitian selanjutnya.

1.5.3. Penelitian ini mengembangakan program *POS Tagging* bahasa Indonesia yang dapat mengatasi masalah kata ambigu dan kata yang tidak ada di kamus bahasa.

1.6. Keaslian Penelitian

Penelitian mengenai *POS Tagging* sudah pernah dilakukan baik bahasa asing atau bahasa Indonesia. Namun, penelitian yang yang diajukan sebagai Tugas Akhir S1 program studi Teknik informatika Fakultas Sains dan Teknologi UIN Sunan Kalijaga mengenai "*Part Of Speech Tagging* Bahasa Indonesia Menggunakan Algoritma *Decision Tree*" belum pernah dilakukan.

1.7. Sistematika Penulisan

Sebagai gambaran dan kerangka yang jelas mengenai pokok bahasan setiap bab dalam penelitian ini, maka diperlukan sistematika penulisan. Penyusunan laporan tugas akhir ini memiliki sistematika penulisan yang diawali dari BAB I dan diakhiri BAB V. Berikut adalah penjelasan pada tiap-tiap bab dalam laporan penelitian ini:

BAB I PENDAHULUAN

Bab pendahuluan berisikan penjelasan mengenai latar belakang dilakukannya penelitian, rumusan masalah penelitian, batasan masalah, tujuan penelitian, manfaat penelitian, keaslian penelitian, dan sistematika penulisan penelitian.

BAB II TINJAUAN PUSTAKA DAN LANDASAN TEORI

Bab tinjauan pustaka dan landasan teori berisikan mengenai penelitian terdahulu dan teori-teori dasar yang terkait dengan penelitian ini.

BAB III METODE PENELITIAN

Bab metode penelitian berisi tentang penjelasan mengenai metode ataupun algoritma yang digunakan serta tahapan-tahapan yang dilakukan untuk mencapai tujuan dan kesimpulan tugas akhir.

BAB IV HASIL DAN PEMBAHASAN

Bab hasil dan pembahasan membahas analisis data dan hasil dari penelitian yang telah dilakukan.

BAB V PENUTUP

Bab penutup berisi tentang kesimpulan dari hasil penelitian yang telah dilakukan. Selanjutnya, kekurangan yang ada pada penelitian dituliskan pada saran untuk pengembangan penelitian dimasa yang akan datang.

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan dengan menggunakan data KBBI dan data berita online CNN Indonesia yang menghasilkan 40.071 *dataset* yang digunakan untuk data latih atau membangun model *Decision Tree* pada aplikasi *POS Tagging*. maka dapat disimpulkan bahwa *POS Tagging* dengan menggunakan algoritma *Decision Tree* pada bahasa Indonesia mendapatkan akurasi sebesar 95,3%, serta mendapatkan persentase keberhasilan 90,0% dalam memberikan tag untuk kata yang tidak ada di kamus dan keberhasilan sebesar 91,9% untuk memberikan tag pada kata ambigu. Hasil implementsi aplikasi *POS Tagging* terhadap kalimat berbahasa indonesia sudah baik meskipun belum maksimal, hal ini dipengaruhi metode dengan pendekatan probabilistik tingkat kualitas hasilnya dipengaruhi oleh jumlah dataset latih yang digunakan ketika membangun model *Decision Tree* atau fitur yang digunakan.

5.2 Saran

Berdasarkan hasil penelitian ini, ada beberapa saran untuk dijadikan perbaikan untuk penelitian selanjutnya, yaitu:

1. Pada penelitian selanjutnya dapat melakukan perbandingan ataupun menggunakan metode klasifikasi lain sehingga dapat mengetahui metode klasifikasi terbaik untuk *POS Tagging.*

2. Menambah dan menggunakan *corpus* dari berbagai jenis media lain seperti novel, jurnal ilmiah, artikel dan lain sebagainya.
3. Menambah atau mencari fitur terbaik dalam *POS Tagging* Bahasa Indonesia.
4. Menambahkan proses *Named Entity Recognition* (NER) pada *POS Tagging*

DAFTAR PUSTAKA

Adriani, M., Asian, J., Williams, H. E., Tahaghoghi, S. M. M., & Williams, H. E. (2007). Stemming Indonesian. *Conferences in Research and Practice in Information Technology Series*, *38*(4), 307–314. https://doi.org/10.1145/1316457.1316459

Daniel Jurafsky & James H. Martin. (2006). Speech and Language Processing: AnAn introduction to natural language processing, computational linguistics, and speech recognition. Daniel. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.*, 1004.

Departemen Pendidikan Nasional, Pusat Bahasa. (2008). Kamus Besar Bahasa Indonesia Edisi IV. Jakarta : Pusat Bahasa

Ferdinandus, F. X., Wijaya, T. K., & Sugianto, N. A. (2009). *Part of Speech Tagging on Indonesian Language using Descision Tree*. 12–17.

Fu, S., Lin, N., Zhu, G., & Jiang, S. (2018). Towards Indonesian Part-of-Speech Tagging : Corpus and Models. *Proceedings of LREC 2018 Workshop on Belt and Road LRE*, *1*, 2–7. http://universaldependencies.org/

Han, J., Micheline, K., & Jian, P. (2012). *Data mining: Data mining concepts and techniques* (3rd ed.). Elsevier Inc. https://doi.org/10.1109/ICMIRA.2013.45

Kamayani, M. (2019). Perkembangan Part-of-Speech Tagger Bahasa Indonesia. *Jurnal Linguistik Komputasional (JLK)*, *2*(2), 34.

https://doi.org/10.26418/jlk.v2i2.20

Kao, A., & Poteet, S. R. (2007). Natural language processing and text mining. In *Natural Language Processing and Text Mining*. https://doi.org/10.1007/978-1-84628-754-1

Kementerian Pendidikan dan Budaya, Badan Pengembangan dan Pembinaan Bahasa. (2017). Tata Bahasa Baku Bahasa Indonesia Edisi IV. Jakarta : Badan Pengembangan dan Pembinaan Bahasa

Kuhlman, D. (2013). A Python Book. *A Python Book*, 1–227.

Larose, D. T. (2014). Discovering Knowledge in Data: An Introduction to Data Mining: Second Edition. In *Discovering Knowledge in Data: An Introduction to Data Mining: Second Edition* (Vol. 9780470908). https://doi.org/10.1002/9781118874059

Manliguez, C. (2016). *Generalized Confusion Matrix for Multiple Classes*. *November*, 2–4. https://doi.org/10.13140/RG.2.2.31150.51523

Marquez, L., & Rodriguez, H. (1998). Part-of-speech tagging using decision trees. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *1398*, 25–36. https://doi.org/10.1007/bfb0026668

Orphanos, G., Kalles, D., Papagelis, T., & Christodoulakis, D. (1999). Decision Trees and NLP : A Case Study in POS Tagging. *Proceedings of Annual Conference on Artificial Intelligence (ACAI)*, 1–7.

Orphanos, G. S., & Christodoulakis, D. N. (1999). *POS disambiguation and unknown word guessing with decision trees*. 134. https://doi.org/10.3115/977035.977054

Orphanos, G., & Tsalidis, C. (1999). Combining handcrafted and corpus-acquired Lexical Knowledge into a Morphosyntactic Tagger. *Proceedings of the 2nd Research Colloquium for Computational Linguistics in United Kingdom*, 1–8.

Pedregosa, F., Weiss, R., & Brucher, M. (2011). *Scikit-learn : Machine Learning in Python*. *12*, 2825–2830.

Pisceldo, F., Adriani, M., & Manurung, R. (2009). *Probabilistic Part Of Speech Tagging for Bahasa Indonesia*.

Prihatini, P. M. (2016). *Implementasi ekstraksi fitur pada pengolahan dokumen berbahasa indonesia*. *6*(3), 174–178.

Ramadhanti, F., Wibisono, Y., & Sukamto, R. A. (2019). Analisis Morfologi untuk Menangani Out-of-Vocabulary Words pada Part-of-Speech Tagger Bahasa Indonesia Menggunakan Hidden Markov Model. *Jurnal Linguistik Komputasional (JLK)*, *2*(1), 6. https://doi.org/10.26418/jlk.v2i1.13

Sabloak, N., Hardono, B. A., & Alamsyah, D. (2016). *Part-of-Speech (POS) Tagging Bahasa Indonesia Menggunakan Algoritma Viterbi*. *x*, 1–11.

Sarmah, J., & Sarma, S. K. (2016). Decision Tree based Supervised Word Sense Disambiguation for Assamese. *International Journal of Computer*

Applications, *141*(1), 42–48. https://doi.org/10.5120/ijca2016909488

Yazid, A. S., & Fatwanto, A. (2018). Penentuan Kelas Kata Pada Part of Speech Tagging Kata Ambigu Bahasa Indonesia. *JISKA (Jurnal Informatika Sunan Kalijaga)*, *2*(3), 157. https://doi.org/10.14421/jiska.2018.23-05

LAMPIRAN

Lampiran 1 *Source Code Function (allFunction)*

```
1.  import pandas as pd
2.  import pymysql
3.  import csv
4.  import re
5.  import nltk
6.  from bs4 import BeautifulSoup
7.  from newspaper import Article
8.  import datetime
9.  import pandas as pd
10. import requests
11. from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
12.
13.
14. # mendapatkan link berita berdasarkan rentang waktu tertentu
15. def getLinksByDate(page,startDate,finalDate,month,year):
16.     #alamat website berita CNN
17.     website = 'https://www.cnnindonesia.com/indeks/'
18.
19.     #menyimpan link
20.     links = []
21.
22.     for i in range(startDate, finalDate+1):
23.         for j in range(page):
24.             dateNews = datetime.date(year, month, i)
25.             dateNews = dateNews.strftime('%Y/%m/%d')
26.
27.             websitePage = requests.get(website+str(j+1)+'?date='+str(dat
    eNews))
28.             soup = BeautifulSoup(websitePage.text,'lxml')
29.
30.             indeksBerita = soup.find('div', {'class': 'l_content'})
31.             articles = indeksBerita.find_all('article')
32.
33.             for article in articles:
34.                 links.append(article.find('a')["href"])
35.
36.     return links
37.
38. # mendapatkan isi berita dari link a=yang sudah diapatkan dari getLinksB
    yDate
39. def getContent(link):
40.     websitePage = requests.get(link)
41.     soup = BeautifulSoup(websitePage.text,'lxml')
42.
43.     article = ''
44.
45.     try:
46.         article = soup.find("div", {"id": "detikdetailtext"})
47.     except:
48.         pass
49.
50.     try:
51.         article.find('script').decompose()
52.     except:
53.         pass
```

```
54.
55.     try:
56.         article.find("a", {"id": "idvideocnn"}).decompose()
57.     except:
58.         pass
59.
60.     try:
61.         article.find('b').decompose()
62.     except:
63.         pass
64.
65.     try:
66.         tables = article.find_all('table')
67.         for table in tables:
68.             table.decompose()
69.     except:
70.         pass
71.
72.     try:
73.         allBr = article.find_all('br')
74.         for br in allBr:
75.             br.replace_with(' ')
76.     except:
77.         pass
78.
79.     try:
80.         allCenter = article.find_all('center')
81.         for center in allCenter:
82.             center.replace_with(' ')
83.     except:
84.         pass
85.
86.     return article.text.replace('\n',' ')
87.
88. # cleaning data berita
89. def cleaning(news):
90.     news = re.sub(r'\xa0',' ', news)
91.     news = re.sub(r'^.*\n', ' ', news).strip()
92.     news = re.sub(r'Bagikan :', ' ', news).strip()
93.
94.     return news
95.
96. # mencari data pada database lexicon (KBBI)
97. def lexiconLookUp(word):
98.     #database connection
99.     db = pymysql.connect("localhost","root","","kbbi4")
100.
101.         # prepare a cursor object using cursor() method
102.         cursor = db.cursor()
103.
104.         # for word in words:
105.         sql = "SELECT kelas_kata FROM kata_dasar WHERE kata ='"+db.es
    cape_string(str(word))+"' UNION SELECT kelas_kata FROM kata_turunan WHER
    E kata ='"+db.escape_string(str(word))+"'"
106.
107.         # Execute the SQL command
108.         cursor.execute(sql)
109.
110.         if cursor.execute(sql) > 1:
111.             rows = cursor.fetchall()
112.             result = []
```

```
113.                 for row in rows:
114.                     if row[0] not in result and row[0] != "" :
115.                         result.append(row[0])
116.
117.                 if len(result) < 2 :
118.                     result = result[0]
119.                 else :
120.                     result.sort()
121.
122.             elif cursor.execute(sql) == 1:
123.                 row = cursor.fetchone()
124.                 result = row[0]
125.
126.                 if result == "" :
127.                     result = "none"
128.             else:
129.                  result = "none"
130.
131.             return str(result)
132.
133.         # memberikan label kelas kata
134.         def labeling_lexicon(word) :
135.             if re.search("\d", word):
136.                 if re.search("[A-Za-z]", word):
137.                     tag = "n"
138.                 else :
139.                     tag = "num"
140.             elif(re.match("^\W+$", word)):
141.                 tag = "z"
142.             else:
143.                 tag = lexiconLookUp(word)
144.
145.             return tag
146.
147.         # mencari imbuhan awal dan akhir pada kata
148.         def morphology(word) :
149.
150.             factory = StemmerFactory()
151.             stemmer = factory.create_stemmer()
152.
153.             list_sufiks = ["lah","kah","tah","pun","ku","mu","nya","kan",
    "an","i"]
154.             list_prefiks = ["be","di","ke","me","pe","se","te"]
155.
156.             kata = word.lower()
157.             stamming = stemmer.stem(kata)
158.             prefiks = ""
159.             sufiks = ""
160.
161.             if stemmer.stem(kata) != kata :
162.                 for i in range(4,1,-1):
163.                     if kata[:i] in list_prefiks :
164.                         if stamming[:i] not in kata[:i] :
165.                             prefiks = kata[:i]
166.                             break
167.
168.                         elif stamming[:i] in kata[:i] and kata[:i]*2 in k
    ata :
169.                             prefiks = kata[:i]
170.                             break
171.
```

```
172.                 for i in range(5,0,-1):
173.                     if kata[-i:] in list_sufiks :
174.                         if stamming[-i:] not in kata[-i:] :
175.                             sufiks = kata[-i:]
176.                             break
177.
178.                         elif stamming[-i:] in kata[-i:] and (kata[-
    i:]*2 in kata and kata[-i:] != "i") :
179.                             sufiks = kata[-i:]
180.                             break
181.
182.             return prefiks,sufiks
183.
184.         # function ekstraksi fitur data
185.         def create_feature(idx,word,tag,data) :
186.             punc = [".","!","?"]
187.             tag_lexicon = labeling_lexicon(word)
188.
189.             feature = {
190.                 "word" : word,
191.                 "tag"  : tag,
192.
193.                 "ambiguity" : tag_lexicon if re.search("\[",tag_lexicon)
    else "",
194.
195.                 "is_capitalized" : word[0].upper() == word[0],
196.                 "is_all_capitalized" : word.upper() == word,
197.                 "is_all_lower": word.lower() == word,
198.
199.                 "is_digit" : word.isdigit(),
200.
201.                 "prefix" : morphology(word)[0],
202.                 "suffix" : morphology(word)[1],
203.
204.                 "prev_tag" : "first" if idx == 0 or data["word"][idx-
    1] == "." or (idx > 1 and data["word"][idx-
    2] in punc and re.match("\W", data["word"][idx-
    1])) else data["tag"][idx-1],
205.                 "next_tag" : "last" if idx == len(data)-
    1 or data["word"][idx+1] in punc else data["tag"][idx+1],
206.             }
207.
208.             return feature
209.
210.         # perulangan mencari fitur pada semua kata
211.         def form_data(data) :
212.             features = []
213.             labels = []
214.             for idx,val in data.iterrows():
215.
216.                 if val["tag"] != "z" and val["tag"] != "." and val["tag"]
     != "fw":
217.                     word = val["word"]
218.                     tag = val["tag"]
219.
220.                     labels.append(tag)
221.                     features.append(create_feature(idx,word,tag,data))
222.
223.             return features, labels
224.
225.         # mengeluarkan fitur pada dictionary yang tidak digunakan
```

```
226.        def pop_dict (key,data_dict):
227.            result_pop = []
228.            for i in data_dict :
229.                i.pop(key)
230.
231.                result_pop.append(i)
232.
233.            return result_pop
```

Lampiran 2 *Source Code Scraping* Web Berita CNN Indonesia

```
1.  from newspaper import Article
2.  import pandas as pd
3.  import csv
4.  import allFunction
5.
6.  # scraping berita berdasarkan rentang waktu tertentu
7.  links = POSTagging.getLinksByDate(1,1,20,1,2020)
8.
9.  with open('../dataset/data-berita-1-20-jan-
    20.csv','w',newline='',encoding='utf-8') as csvFile:
10.     csvWriter = csv.writer(csvFile)
11.     csvWriter.writerow(['publish_date','link','title','text'])
12.
13.     for link in links:
14.         # menggunakan Article dari library newspaper
15.         news = Article(link)
16.         news.download()
17.         news.parse()
18.         news.nlp()
19.
20.         print("Scraping.... "+news.title)
21.
22.         try:
23.             contentNews = POSTagging.getContent(link)
24.         except:
25.             contentNews = news.text
26.             contentNews.replace('\n',' ')
27.
28.         csvWriter.writerow([news.publish_date,link,news.title,contentNew
    s])
29.
30. # menampilkan hasil scraping berita CNN Indonesia
31. berita = pd.read_csv('../dataset/data-berita-1-20-jan-20.csv')
32. berita
```

Lampiran 3 *Source Code Labeling Data Set*

```
1.  import pandas as pd
2.  import csv
3.  import re
4.  import nltk
5.  import allFunction
6.
7.  # menampilkan hasil scraping berita CNN Indonesia
8.  allNews = pd.read_csv("../dataset/data-berita-1-20-jan-20.csv")
9.  allNews
10.
```

```
# memcah data menjadi perkalimat
sentences = []
for articles in allNews["text"]:
    articles = POSTagging.cleaning(articles)
    result_st = nltk.sent_tokenize(articles)

    for result in result_st:
        sentences.append(result)

# menampilkan data kalimat berita
sentences = pd.DataFrame(sentences, columns =['Kalimat'])
sentences.head()

# memecah data kalimat berita menjadi perkata
words = []
for sentence in sentences["Kalimat"]:
    result_wt = nltk.word_tokenize(sentence)

    for result in result_wt:
        words.append(result)

# labeling data kata berdasarkan lexicon (kbbi)
words = pd.DataFrame(words, columns =['Token'])

with open('../dataset/new-data-labeling-corpus-1-20-jan-
20.csv','w',newline='',encoding='utf-8') as csvFile:
    csvWriter = csv.writer(csvFile)
    csvWriter.writerow(['token','tag'])

    for word in words["Token"]:

        tag = POSTagging.labeling_lexicon(word)

        print(word,tag)
        csvWriter.writerow([word,tag])
```

Lampiran 4 *Source Code* Ekstraksi Fitur

```
import pandas as pd
import numpy as np
import pickle
import allFunction

# load file data trining
path_file = "../dataset/data-labeling-corpus-1-20-jan-20-edited.csv"

data = pd.read_csv(path_file)
data = data[["word","tag"]]
data

# create feature
features, labels = POSTagging.form_data(data)

# simpan features
a_file = open("../dataset/all-features.pkl","wb")
pickle.dump(features,a_file)
a_file.close()

# simpan labels
```

```
with open("../dataset/all-labels.data","wb") as filehandle :
    pickle.dump(labels,filehandle)
```

Lampiran 5 *Source Code* Evaluasi Model *Decision Tree*

```
import pandas as pd
import numpy as np

from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score
from sklearn.feature_extraction import DictVectorizer

from sklearn import tree

import pickle
import re

import allFunction

# buka file features
a_file = open("../dataset/all-features.pkl","rb")
features = pickle.load(a_file)
a_file.close()

# buka file labels
with open("../dataset/all-labels.data","rb") as filehandle :
    labels = pickle.load(filehandle)

# keluarkan fitur tag dan is_all_lower pada dict features
fetaures = POSTagging.pop_dict("tag",features)
fetaures = POSTagging.pop_dict("is_all_lower",features)

# menampilkan jumlah dataset yang digunakan
len(features)

# split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(features, labels, te
st_size=0.3, random_state=5) # 70% training and 30% test

# jumlah data latih hasil split
print("Jumlah Data Latih : "+str(len(X_train)))

# keluarkan fitur word pada data training
X_train = POSTagging.pop_dict("word",X_train)

# vectorizing ekstraksi fitur pada data training
vectorizer = DictVectorizer()
X_train = vectorizer.fit_transform(X_train)

# learning decision tree
clf_tree = tree.DecisionTreeClassifier()
dt = clf_tree.fit(X_train,y_train)

# dataframe feature dan labels pada data test
df_features = pd.DataFrame.from_dict(X_test)
df_features["labels"] = y_test
df_features
```

```
54.
55. # labelng kata berdasarkan database lexicon (KBBI)
56. result_labeling = []
57. for idx, val in df_features.iterrows() :
58.     word = val["word"]
59.     tag = POSTagging.labeling_lexicon(word)
60.     print([word,tag])
61.     result_labeling.append([word,tag])
62.
63. # menampilkan hasil data test setelah labeling lexicon
64. result_labeling = pd.DataFrame(result_labeling,columns = ["kata", 'kelas
     kata kamus'])
65. result_labeling
66.
67. # pemberian kelas kata pada kata ambigu dan kata yang tidak ada di lexic
    on dengan menggunakan decision tree
68. results = []
69. y_pred = []
70.
71. for idx,val in result_labeling.iterrows() :
72.     if val["kelas kata kamus"] == 'none' or re.search("\[",val["kelas ka
    ta kamus"]) :
73.         kelas_kata_pred = str(dt.predict(vectorizer.transform(X_test[idx
    ]))[0])
74.         print([val["kata"],str(val["kelas kata kamus"]),kelas_kata_pred,
    y_test[idx]])
75.     else :
76.         kelas_kata_pred = val["kelas kata kamus"]
77.
78.     y_pred.append(kelas_kata_pred)
79.
80.     results.append([val["kata"],str(val["kelas kata kamus"]),kelas_kata_
    pred])
81.
82. results = pd.DataFrame(results,columns = ["kata", "kelas kata kamus", "k
    elas kata prediksi"])
83.
84. # menampilkan hasil pos-tagging
85. results
86.
87. # menampilkan hasil klasifikasi report
88. print(classification_report(y_pred,y_test))
89.
90. # menampilkan hasil confusion matrix
91. confusion_matrix(y_test,y_pred)
92.
93. # menampilkan accuracy score
94. accuracy_score(y_pred,y_test)
95.
96. # menghitung jumlah kata yang tidak diketahui
97. a = 0
98. b = 0
99. for idx,val in results.iterrows() :
100.
101.         if val["kelas kata kamus"] == "none" :
102.             a = a+1
103.
104.             if val["kelas kata prediksi"] == y_test[idx] :
105.                 b = b+1
106.
107.     print("Jumlah Kata Yang Tidak Diketahui "+str(a))
```

```
        print("Jumlah Prediksi Benar Kata Yang Tidak Diketahui "+str(b))

        # menghitung jumlah kata ambigu
        a = 0
        b = 0
        x = []
        for idx,val in results.iterrows() :

            if re.search("\[",val["kelas kata kamus"]) :
                a = a + 1
                x.append(val["kelas kata kamus"])

                if val["kelas kata prediksi"] == y_test[idx] :
                    b = b+1

        print("Jumlah Kata Ambigu "+str(a))
        print("Jumlah Prediksi Benar Kata Ambigu "+str(b))
```

Lampiran 6 ***Source Code Learning Decision Tree***

```
import pandas as pd
import numpy as np

from sklearn.feature_extraction import DictVectorizer
from sklearn import tree

import pickle
import re

import allFunction

# buka file features
a_file = open("../dataset/all-features.pkl","rb")
features = pickle.load(a_file)
a_file.close()

# buka file labels
with open("../dataset/all-labels.data","rb") as filehandle :
    labels = pickle.load(filehandle)

# keluarkan fitur pada dict features data training
features = POSTagging.pop_dict("word",features)
features = POSTagging.pop_dict("tag",features)
features = POSTagging.pop_dict("is_all_lower",features)

# vectorizing ekstraksi fitur
vectorizer = DictVectorizer()
vectorized_features = vectorizer.fit_transform(features)
vectorized_features

# menampilkan semua features hasil vectorization
vectorizer.get_feature_names()

# learning decision tree
clf_tree = tree.DecisionTreeClassifier(criterion='entropy')
dt = clf_tree.fit(vectorized_features,labels)

```

```
# simpan object vectorizer dan metode-metode kedalam bentuk binary file
vectorizer
vectorizer_file = open("../dataset/vectorizer.b",'wb')
pickle.dump(vectorizer, vectorizer_file)

# decision tree
dt_file = open("../dataset/dt.b","wb")
pickle.dump(dt, dt_file)
```

Lampiran 7 *Source Code Aplikasi Part of Speech Tagging*

```
import streamlit as st
import pandas as pd
import numpy as np
import pickle
import nltk
import re
from sklearn.feature_extraction import DictVectorizer
from sklearn import tree
import allFunction

vectorizer = DictVectorizer()
dt = tree.DecisionTreeClassifier()

# load pengetahuan atau file pickle
vectorizer = pickle.load(open('../dataset/new-vectorizer.b','rb'))
dt = pickle.load(open('../dataset/new-dt.b','rb'))

# form input kalimat
text = st.text_input("Masukkan Kalimat")

# part of speech tagging
words = []
for word in nltk.word_tokenize(text) :
    tag = POSTagging.labeling_lexicon(word)
    words.append([word, tag])
result_labeling = pd.DataFrame(words,columns = ["word", "tag"])

results = []
for idx,val in result_labeling.iterrows() :
    if val["tag"] == 'none' or re.search("\[",val["tag"]) :
        features = POSTagging.create_feature(idx,val["word"],val["tag"],result_labeling)
        result_tagging = str(dt.predict(vectorizer.transform(features))[0])

    else :
        result_tagging = val["tag"]

    results.append([val["word"],str(val["tag"]),result_tagging])

results = pd.DataFrame(results,columns = ["kata", "kelas kata kamus","kelas kata prediksi"])

# menampilkan hasil pos-tagging
if len(results) != 0 :
    st.write(results.T)
```

CURRICULUM VITAE

A. Biodata Diri

Nama Lengkap	:	Ahmad Ardiyanto
Jenis Kelamin	:	Laki-laki
Tempat, Tanggal Lahir	:	Magelang, 19 Oktober 1996
Alamat Asal	:	Dusun Macanan RT 02/RW 01, Desa Tanjung, Kec. Muntilan, Kab. Magelang, Jawa Tengah
Alamat Tinggal	:	Dusun Macanan RT 02/RW 01, Desa Tanjung, Kec. Muntilan, Kab. Magelang, Jawa Tengah
E-mail	:	ahmadardiyanto23@gmail.com

B. Latar Belakang Pendidikan Formal

Jenjang	Nama Sekolah	Tahun
TK	TK ABA TANJUNG	2001-2002
SD	SD NEGERI TANJUNG	2002-2008
SMP	SMP NEGERI 3 MUNTILAN	2008-2011
SMA	SMK MA'ARIF KOTA MUNGKID	2011-2014
S1	UIN SUNAN KALIJAGA	2016-2020