# Argumentation Mining Menggunakan Ekstraksi Lexical Feature dan Contextual Feature dengan Metode Naïve Bayes

**Dievarino Dwisusanto[1], Ibnu Asror[2],Yanuar Firdaus Arie Wibowo[3]**
[1;2;3]Fakultas Informatika, Universitas Telkom, Bandung
[1]dievarinods@student.telkomuniversity.ac.id, [2]iasror@telkomuniversity.ac.id,
[3]yanuar@telkomuniversity.ac.id

**Abstrak**

**Zaman ini perkembangan informasi semakin pesat termasuk perkembangan pada data yaitu data teks. Teks merupakan data berbahasa alami yang tidak terstruktur. Seiring berjalannya waktu data teks semakin banyak, dengan itu maka dilakukan pemanfaatan pada data teks , salah satu nya pemanfaatan untuk menentukan komponen argumen pada teks. Argumen sendiri sering kali ditemukan diberbagai data teks, contohnya pada berita, essai, debat online. Dengan mendeteksi secara otomatis komponen argumen pada teks, dapat diketahui bahwa sebuah teks tersebut mengandung unsur argumen atau tidak, yang itu berguna untuk pencarian dan pegolahan informasi. Solusi dari permasalahan teks tersebut adalah dengan membangun sebuah model sistem yaitu naive bayes classifer yang akan mengklasifikasikan komponen argumen pada teks, yaitu komponen *claim* atau komponen *premise*. *Claim* dan *premise* tersebut dibutuhkan untuk pembentukan sebuah argumen. Dengan dibangunnya model *classifier*, evaluasi yang dilakukan pada hasil klasifikasi menggunakan *preprocessing*, lalu ekstraksi fitur *lexical* dan *contextual*. Hasil paling optimal pada penelitian ini adalah penggunaan fitur *lexical* dan *contextual* dan tanpa menggunakan laplace smoothing yang mendapatkan tingkat akurasi 66.84% dan *f1 score* 79.45%.**

**Kata kunci : argumen, *claim*, *premise*, naive bayes, ekstraksi fitur, *lexical*, *contextual***

**Abstract**

**In this era, the development of information is growing very fast including the development of data especially text data. Text is unstructured natural language data, as time goes by the text data will be more and more, so it can be used on utilization itself, one of the utilization of data text is how to determine the components of the argument in the text. The argument itself is often found in various text data, for example on news, essays, and online debates. By automatically detecting the components of the argument in the text, then it can be known that the text contains an argument or not, which is useful for information retrieval and searching information. The solution for this problem is to build a system model called naive bayes classifier that will classify the components of the argument in the text, the components are *claim* or *premise*. The *claim* and *premise* are the component to build an argument itself. By building the classifier then the classification result will be evaluated by doing *preprocessing*, then extracting the *lexical* and *contextual* feature. The most optimal results in this study are the use of *lexical* and *contextual* features and without using laplace smoothing which gets an *accuracy* rate of 66.84% and *f1 score* 79.45%.**

**Keywords: argument, *claim*, *premise*, naive bayes, feature extraction, *lexical*, *contextual***

## 1. Pendahuluan

### Latar Belakang

*Text mining* adalah proses ekstrasi pola berupa informasi dan pengetahuan yang berguna dari sejumlah besar sumber data teks, seperti dokumen word, PDF, kutipan teks, dll. Jenis masukan untuk penambangan teks ini disebut data tak terstruktur dan merupakan pembeda utama dengan penambangan data yang menggunakan data terstruktur atau basis data sebagai masukan.[1] Upaya pencarian dan penambangan data yang ada di dalam sebuah dokumen, bertujuan untuk mencari kata-kata yang dapat mewakili isi dari sebuah dokumen sehingga dapat dilakukan analisa keterhubungan pada setiap dokumen. *Text minning* memiliki banyak teknik salah satunya *Argumentation Mining.*

Argumentasi adalah sebuah alasan yang biasa digunakan untuk memperkuat atau membantah suatu pendapat, gagasan, atau pendirian.[2] Argumentasi biasanya dilakukan dengan lisan ataupun tulisan. *Argumentation mining* merupakan sebuah teknik yang berfokus pada ekstraksi dan analisis sebuah kalimat argumen dalam suatu teks bahasa. [2]. Argumentasi merupakan aktivitas yang bertujuan untuk meningkatkan atau mengurangi penerimaan sebuah sudut pandang bagi para pendengar atau pembaca, yang bertujuan untuk membenarkan sebuah sudut pandang yang lebih masuk akal [3].Penggunaan *Argumentation Mining* ini dapat memudahkan penulis atau pembuat pernyataan dalam memerikasa teks yang lebih rasional dan meningkatkan

kualitas sebuah argumentasi[4]. *Argumentation Mining* sendiri berfokus pada identifikasi kalimat argumen yang berupa komponen dan relasi antar kalimat argumen[5].Pada komponen sebuah kalimat argumen , terdiri dari *claim* dan *premise*. *Claim* adalah komponen utama pada sebuah kalimat argumen yang dapat diterima atau ditolak pendengar atau pembaca. Sedangkan *premise* bertujuan untuk memvalidasi komponen *claim* agar pendengar atau pembaca dapat memastikan penerimaan pada *claim* tersebut tersebut atau malah sebaliknya.[6]. Identifikasi dan klasifikasi yang akan dilakukan menggunakan ektraksi fitur, ektraksi fitur adalah salah satu cara untuk melakukan identifikasi dan klasifikasi. Penggunaanya ekstraksi fitur mengambil ciri-ciri yang dapat menggambarkan karakteristik pada sebuah teks[7]

Permasalahanya, banyak orang yang menyebutkan teks argumen pada sebuah teks seperti teks artikel, teks essai, teks debat, teks novel dan lain lain. Isi dari teks tersebut memiliki kalimat pernyataan yang disebut *claim*. Tetapi tidak semua kalimat *claim* memiliki tujuan yang sama, seperti menunjukan sebuah kalimat argumentasi pada teks tersebut. Maka teks tersebut belum bisa dikatakan sebagai teks argument. Sehingga Pada penelitian ini, akan menerapkan klasifikasi argumen dengan *argumentation mining* pada data teks.. Untuk mencari tingkat akurasi suatu fitur pada kalimat argument. Dalam melakukan klasifikasi argument akan menggunakan *Argumentation mining*, dan akan diekstraksi menggunakan *lexical features* dan *contextual features*, karena *lexical* features dan *contextual* features masing-masing menggunakan cara ekstraksi yang berbeda. Setelah itu hasil klasifikasi dari *lexical features* dan *contextual features* akan diproses menggunakan metode *naïve bayes*.alasan penggunaan *lexical features* dan *contextual features* dikarenakan 2 fitur tersebut memiliki kesamaan pada saat melakukan pencirian pada kalimat. 2 fitur tersebut terfokus pada label kata-kata pada kalimat, yang dilabelkan pada saat *post tagging.*

**Batasan Masalah**

Pada penelitian cara mengklasifikasikan komponen argumen dari esai persuasif berbahasa Inggris yang berasal dari *website TU-Darmstadt* dengan hanya menggunakan ekstrasi fitur *lexical* dan *contextual*, model *naive bayes* sebagai klasifikasi. Dimana data teks sudah dilabelkan ke dalam bentuk *claim* dan *premise*. Sedangkan untuk nilai atribut fitur pada klasifikasi *premise* dan *claim* belum ada parameter yang pasti, sehingga nilai parameter berdasarkan dataset yang telah disediakan. Dan pada saat melakukan preprocessing menggunakan NLTK pada python.

**Tujuan**

Tujuan dari penelitian ini adalah menghitung hasil performansi dari ekstrasi fitur *lexical* dan *contextual* dalam mengklasifikasikan komponen argumen dengan menggunakan metodel naive bayes. Kemudian evaluasi menggunakan 10-fold Validation untuk membagi data train dan data test lalu model evaluasi dihitung menggunakan nilai *accuracy*, precision, *recall*, dan F1-Score.

## 2. Studi Terkait

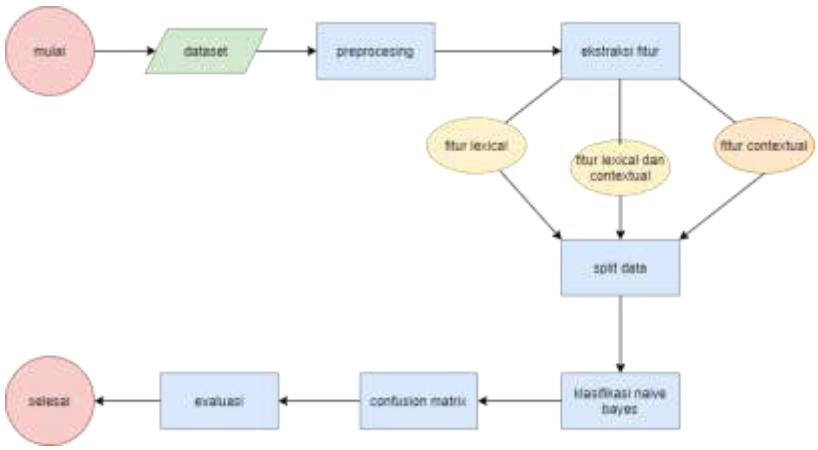
Pada tahun 1993 Knott dan Dale membuat sebuah *list keyword* dalam dan dikelompokan dalam set fitur[8]. Dimana *list keyword* tersebut digunakan untuk acuan pada penggunaan fitur pada penelitian-penelitian selanjutnya. Pada tahun 2007 Moens, Palau, Boiy, dan Reed mereka menerapkan klasifikasi pada dokumen hukum dengan menggunakan ekstrasi fitur. Fitur yang digunakan adalah *unigrams*, *bigrams*, *trigrams*, *adverbs*, *verbs*, *modal auxiliary*, *word couples*, *text statistic*, *punctuation*, dan *parse features*. Klasifikasi menggunakan *naive bayes* dan *maximum entropy model*. Dari hasil yang dihasilkan, penggunaan fitur *word couples*, *verbs*, dan *text statistic* dengan model *naive bayes* memiliki performa yang paling baik[9].

Pada tahun 2014 Stab dan Gurevych mereka melakukan penelitian untuk mengklasifikasikan komponen argumen yaitu *claim* dan *premise* dengan menggunakan pengelompokkan fitur dari Moens, Palau, Boiy, dan Reed yang dilakukan pada tahun 2007 diubah menjadi fitur *Lexical, Contextual, Structural, Syntacticl* dan *Indicator* yang memiliki karakteristik berbeda-beda. Klasifikasi yang dilakukan menggunakan 90 esai persuasif dan menggunakan model *support vector machine* yang dievaluasi *10-fold cross validation*. *[4]*.

Pada tahun 2015 Suhartono melakukan klasifikasi komponen menggunakan *support vector machine* (SVM) dengan menggunakan fitur *lexical*, *indicator*, *structural* yang diadopsi dari penelitian Stab dan Gurevych (2014) untuk mengetahui pengaruh penggunaan *keyword* dari penelitian Knott dan Dale (1993). Evaluasi yang dilakukan dengan menggunakan *10-fold cross validation* , dengan penggunaan fitur *indicator* menghasilkan nilai akurasi yang paling baik[2].

### 2.1 Sistem yang Dibangun

Sistem yang akan dibuat pada penelitian ini adalah sistem klasifikasi otomatis pada persuasif essai yang dilakukan dalam berbagai tahap. Tahap pertama adalah melakukan *preprocessing* dataset, yang nanti akan di ektraksi fitur dan dianalisis hasil klasifikasi nya dan bagaimana pengaruhnya. Kemudian untuk klasifikasi nya menggunakan na¨ıve bayes classifier untuk menentukan probabilitas kelas *claim* dan kelas *premise*. Lalu untuk validasi evaluasi datanya digunakan k-fold cross validation untuk mendapatkan hasil yang dinamis dan optimal. Berikut ini adalah gambaran umum sistem yang akan dibangun pada Gambar 2 :




**Gambar 1**. Gambaran Umum Sistem

### 2.2 Dataset

Dataset adalah objek yang mewakili data, dan data nya ini berupa text yang tidak terstruktur, yang akan diubah menjadi teks terstruktur untuk memudahkan klasifikasi pada sistem. Dataset yang digunakan adalah 1520 record data hasil pelabelan manual dari hasil anotasi 90 persuasive essay yang dibuat Stab et al. dengan catatan major *claim* di anggap *claim*. Record data tersebut dilabelkan secara manual sesuai kalimat argumennya dengan label 0 (*premise*) dan label 1 (*claim*), dan atribut yang digunakan adalah atribut argument.hasil pelabelan manual tersebut antara lain sebanyak 1.010 kalimat merupakan label 0, dan 510 kalimat merupakan label 1.

Untuk melakukan validasi penelitiannya dilakukan k-fold cross validation untuk pembagian data dengan in-putan k = 6 dan k = 10 untuk dibandingkan keoptimalannya. Selanjutnya dataset akan di *preprocessing* untuk memudahkan sistem dalam mengklasifikasi dan mengetahui bagaimana pengaruh nya kepada hasil klasifikasi. Berikut ini adalah gambaran dari dataset yang digunakan pada Tabel 1 :

**Tabel 1.** Gambaran Dataset

| Label | Argument |
|---|---|
| 1 | competition can effectively promote the development of economy |
| 0 | The winner is the athlete but the success belongs to the whole team |
| 1 | they are able to sustain their cultural identities |
| 0 | the one will learn living without depending on anyone else |

### 2.3. *Preprocessing*

Pada proses ini, dataset yang telah dibagi menjadi data train dan data test akan di proses agar mendapatkan data yang lebih baik untuk proses berikutnya. Data *Preprocessing* adalah suatu proses untuk membuat data ber-kualitas rendah menjadi data yang berkualitas tinggi agar mudah untuk diolah. *Preprocessing* yang dilakukan pada penelitian ini yaitu pengurangan dimensi dataset, *case folding*, *tokenization*, Pos tagger. Pengurangan dimensi dataset merupakan pemilihan dimensi pada dataset, dalam hal ini argumen.

#### 2.3.1. *Case folding*

*Case folding* adalah salah satu bentuk *text preprocessing* yang paling sederhana dan efektif meskipun sering diabaikan. Tujuan dari *case folding* untuk mengubah semua huruf dalam dokumen menjadi huruf kecil. Hanya huruf 'a' sampai 'z' yang diterima. Karakter selain huruf dihilangkan dan dianggap *delimiter*. Pada tahap ini tidak menggunakan *external library* apapun, kita bisa memanfaatkan modul yang tersedia di python. Ada beberapa cara yang dapat digunakan dalam tahap *case folding*, anda dapat menggunakan beberapa atau menggunakan semuanya, tergantung pada tugas yang diberikan. Seperti merubah text menjadi lowercase, menghapus angka, menghapus tanda baca,dan menghapus karakter kosong.

#### 2.3.2 .*Tokenization*

Proses untuk memecah korpus teks menjadi elemen individu yang berfungsi sebagai masukan untuk berbagai algoritma pemrosesan bahasa. Pemisahan teks menjadi potongan-potongan yang disebut sebagai token untuk kemudian di analisa. Kata, angka, simbol, tanda baca dan entitas penting lainnya dapat dianggap sebagai token. Didalam token diartikan sebagai "kata" meskipun tokenize juga dapat dilakukan pada paragraf maupun kalimat.

#### 2.3.3 . *Pos tagging*

POS Tagger adalah *the part of speech* (POS) yang menjelaskan bagaimana sebuah kata digunakan pada sebuah kalimat. Maksudnya melabeli kata dengan POS yang sesuai, salah satu contoh POS nya antara lain : kata benda, kata sifat, kata kerja.
Berikut ini adalah gambaran hasil *preprocessing* (*case folding* dan *tokenization*) pada Tabel 2 :

**Tabel 2.** Contoh hasil *Preprocessing*

| Argument | Hasil *Preprocessing* |
|---|---|
| Competition can effectively promote the development of economy | 'competition','can', 'effectively', 'promote', 'the','development', 'of','economy' |
| We should attach more importance to cooperation | 'we', 'should', 'attach','more', 'importance', 'to','cooperation' |

### 2.4 *Lexical*

Kata-kata leksikal adalah kata-kata yang memiliki makna independen (seperti Noun (N), kata kerja (V), kata sifat (A), adverb (Adv), atau preposisi (P). Leksikal adalah deskriptif, melaporkan penggunaan aktual dalam penutur bahasa, dan perubahan dengan mengubah penggunaan istilah, daripada preskriptif, yang akan tetap dengan versi yang dianggap sebagai "benar", terlepas dari penyimpangan dalam makna yang diterima . Mereka cenderung inklusif, mencoba untuk menangkap semua istilah yang digunakan untuk merujuk, dan dengan demikian sering terlalu kabur untuk banyak tujuan.

***Modal Verb***

*Modal verb* adalah kata yang mengekspresikan kebutuhan atau kemungkinan. Yang biasa digunakan untuk membantu kata kerja lain yang menunjukan suasana hati dan *tenses* [8]. Apabila terdapat kata yang mengandung *modal verb* pada komponen argumen, maka nilai fitur yang didapatkan adalah **1**, sedangkan apabila tidak ada maka nilai fitur adalah **0**. Kata-kata yang termasuk dalam modal verb adalah :

**Tabel 3**. Tabel *Modal Verb* [8]

| | | |
|---|---|---|
| Could | Must | May |
| Would | Shall | Might |
| Will | Can | Should |

### 2.5 *Contextual*

konteks memainkan peran utama untuk mengidentifikasi komponen argumen. Misalnya, sebuah premis hanya dapat diklasifikasikan seperti itu, jika ada klaim yang sesuai. Oleh karena itu, kami mendefinisikan fitur-fitur berikut masing-masing diambil dari kalimat sebelumnya dan mengikuti kalimat penutup dari komponen argumen: jumlah tanda baca, jumlah sub-klausa yang menunjukkan keberadaan kata kerja.

***Subclauses***

*Subclauses* adalah kelompok kata yang merupakan bagian suatu kalimat yangmengandung subjek dan predikat. Penggunaan *subclauses sendiri biasanya digunakan* untuk menghubungkan dengan kalimat sebelumnya [8]. Apabila ada kata yang mengandung *subclauses* pada komponen argumen, maka nilai fitur yang didapatkan adalah **1**, sedangkan jika tidak ada maka nilai fitur adalah **0**. Kata-kata yang termasuk *subclauses* adalah :

**Tabel 4**. Tabel *Subclauses* [8]

| | | | | |
|---|---|---|---|---|
| whose | whom | whoever | who | whichever |
| wherever | whether | while | why | which |
| until | when | whenever | where | whereas |
| since | so that | than | that | unless |
| whosever | whomever | although | as | rather than |
| provided | once | even if | even | though |
| after | because | before | in order | if |

### 2.6 *K-fold Cross Validation*

Merupakan proses validasi silang dengan membagi data yang sudah dilabelkan menjadi dua bagian yaitu untuk training dan untuk testing.mempartisi data secara acak menjadi beberapa himpunan.[11]. Biasanya dipilih 10 K, 1 digunakan untuk testing, dan sisanya (K – 1) digunakan untuk training. Pendekatan ini akan diulang dengan memilih masing-masing setiap segmen K yang berbeda pada data sebagai data testing, kemudian rata-rata dari akurasi data testing itu dicatat. Penggunaan *K-fold cross validation* sangat penting karena dapat meningkatkan nilai-nilai pada *recall*, *precission*, *accuracy* dan *f1-score*.

### 2.7 *Laplace Smoothing*

Laplace smoothing merupakan teknik yang bertujuan untuk menambahkan nilai satu atau bilangan kecil lainnya pada suatu fitur untuk menghindari zero probability [12]. Jadi cara kerjanya adalah dengan menambahkan bilangan 1 ke setiap probabilitas kemunculan kata yang bernilai 0, agar terhindar dari proses gagal.

### 2.8 Klasifikasi *Na¨ıve Bayes*

na¨ıve bayes classifier termasuk supervised learning yang didasarkan pada teorema bayes untuk conditional probability yang berfokus pada klasifikasi teks[13]. Dalam kasus umum pada dokumen yang memiliki ukuran besar, algoritma *naive bayes* ini akan lebih efektif.
Berikut adalah perhitungan probabilitas nya berdasarkan teorema bayes, bisa dilihat pada rumus :

$$P(c|x) = \underset{c \in C}{\operatorname{argmax}} \; P(x1, x2, ..., xn \mid C)\, P(c) \quad (1)$$

$$P(cjx) = \underset{c \in C}{\operatorname{argmax}} \; P(c_j) \prod_{x \in X} P(x \mid c) \quad (2)$$

P(xjc) adalah conditional probability dari kata xi yang muncul pada kalimat yang berada dikelas C, atau bisa disebut sebagai likehood probability x pada kelas C. lalu P(c) adalah prior probability dari kalimat yng muncul pada kelas C. Maka perhitungan tersebut akan menentukan penempatan kelas dengan cara membanding kan P(c—x) atau posterior probability terbesar untuk hasil prediksi akan masuk pada kelas mana.

perhitungan prior probability bisa dilihat pada rumus :

$$\dot{P}(c) = \frac{N_c}{N} \quad (3)$$

$N_c$ adalah jumlah kalimat yang ada pada kelas c, sedangkan N adalah jumlah kalimat dari seluruh kelas.
Perhitungan conditional probability bisa dilihat pada rumus :

$$P(wi \mid c\,j) = \frac{count(wi, c\,j)}{2! \sum w\, V\; count(w, c\, j)} \quad (4)$$

count($w_i$ , c $_j$) adalah jumlah kemunculan kata w di kalimat yang ada dikelas c, dan $\sum_{w\ V} count(w, c_j)$ adalah jumlah kemunculan seluruh kata pada kelas c.

### 2.9. *Confusion Matrix*

Confusion matrix berisikan informasi mengenai actual dan predicted oleh model classifier.
Berikut tabel confusion matrix pada Tabel 7 :

**Tabel 5**. Tabel Confusion Matrix

| | | Predicted | |
|---|---|---|---|
| | | Negatif | Positif |
| Actual | Negatif | TN | FP |
| | Positif | FN | TP |

Dengan menggunakan confusion matrix pengukurannya lebih akurat dari akurasi biasa (jumlah data benar dibagi total data dikali 100%), karena dalam confusion matrix akan dihitung dan dijabarkan precision dan *recall* nya. Precision itu untuk mengukur berapa banyak data yang diprediksi positif adalah benar, *recall* itu untuk mengukur berapa banyak data positif yang benar diprediksi.

Setelah menghitung jumlah TN, FN, FP, dan TP nya maka untuk mengevaluasi performa nya :
Rumus Precision:

$$\text{Precision} = \frac{TP}{TP+FP} \quad (5)$$

Rumus *Recall*:

$$Recall = \frac{TP}{TP+FN} \quad (6)$$

Rumus F1-Score:

$$F1\ Score = \frac{Precision x Recall}{Precision + Recall} \quad (7)$$

## 3. Perancangan Sistem

### *3.1 Preprocessing*

Pada saat melakukan tahapan *preprocessing* dataset yang dimiliki akan dibagi menjadi data train dan data test. Preprocesing yang dilakukan pada penelitian ini adalah *case folding tokenization,* dan *Pos tagging* :

#### 3.1.1 *Case folding*

Pada tahap *case folding* dilakukan konversi huruf *capital menjadi huruf kecil semua*
Input : "*We should attach more importance to cooperation during primary education*"
Output : "*we should attach more importance to cooperation during primary education*"

Pada contoh diatas, huruf kapital yang berada di kata "We" dirubah menjadi "we".

#### 3.1.2 *Tokenization*

Selanjutnya pada tahap *tokenization* dilakukan pemecahan korpus teks menjadi elemen masing-masing.
Input : "*we should attach more importance to cooperation during primary education*"
Output : "*we*", "*should*", "*attach*", "*more*", "*importance*", "*to*", "*cooperation*", "*during*", "*primary*", "*education*"

#### 3.1.3 *Pos tagging*

Memberikan klasifikasi pada data teks, yang mana kata tersebut akan ditandai sebagai kata benda, kata kerja, kata sifat, dan sebagainya.
Input : "*we*", "*attach*", "*import*", "*cooper*", "*primary*", "*educ*"
Output :"*we*_PRP", "*attach*_VB", "*import*_NN", "*cooper*_NN", "*primary*_JJ", "*educ*_FW"

### 3.2 Ekstraksi Fitur

Selanjutnya akan dilakukan tahap penhitungan fitur ekstraksi. Dimana perbedaanya pada fitur *lexical* pada saat *Pos tagging* dia mencari *verb, adverb,* dan *modal*, sedangkan pada itur *contextual* yang dilihat adalah jumlah punctuation dan jumlah subklausa. Contoh dari ektraksi fitur pada penelitian ini adalah:

#### 3.2.1 Fitur *Lexical*

Pada fitur *lexical* pada setiap kalimat kita lihat apakah memiliki verb, adverb dan modal. Jika pada kalimat memiliki itu maka nilai setiap komponen adalah 1. Jika tidak maka nilainya adalah 0. Dan pada saat penggunaan kategori jika jumlah dari masing-masing verb,adverb dan modal diantara 0-3 itu maka masing akan bernilai 0, 4-7 masing-masing akan bernilai 1 dan >7 masing-masing akan bernilai 2.
Input: *"**we should attach more importance to cooperation during primary education**."*
.

**Tabel 6.** Contoh Perhitungan Ektraksi Fitur *Lexical* tanpa kategori

| verb | adverb | modal |
|---|---|---|
| 1 | 0 | 1 |

Dari contoh pada tabel 6 diatas pada kalimat tersebut memiliki verb dan modal verb namun tidak mengandung adverb, maka nilai komponen pada verb dan modal adalah 1 sedangkan adverb 0.

**Tabel 7.** Contoh Perhitungan Ektraksi Fitur *Lexical* dengan kategori

| verb | adverb | modal |
|---|---|---|
| 0 | 0 | 0 |

Dari contoh pada tabel 7 diatas pada kalimat tersebut jumlah dari verb, adverb, dan modal verb masing-masing tidak lebih dari 3, maka nilai dari setiap komponennya adalah 0.

### 3.2.2 Fitur *Contextual*

Pada fitur *contextual* berbeda dengan *lexical* kita menghitung berapa banyak punctuation pada setiap kalimat dan subklausanya.. Jika pada kalimat memiliki itu maka nilai setiap komponen adalah 1. Jika tidak maka nilainya adalah 0. Dan pada saat penggunaan kategori jika jumlah dari masing-masing jumlah punctuation dan subklausa diantara 0-3 itu maka masing akan bernilai 0, 4-7 masing-masing akan bernilai 1 dan >7 masing-masing akan bernilai 2. Contohnya sebagai berikut :

Input: *"**we should attach more importance to cooperation during primary education**"*

**Tabel 8.** Contoh Perhitungan Ekstraksi Fitur *Contextual*tanpa kategori

| subcaluse | num of punctuation |
|---|---|
| 0 | 0 |

Dari contoh pada tabel 8 diatas pada kalimat tersebut tidak memiliki punctuation dan subclause maka nilai komponen pada punctuation dan subclause masing-masing adalah 0.

**Tabel 9.** Contoh Perhitungan Ekstraksi Fitur *Contextual* dengan kategori

| subcaluse | num of punctuation |
|---|---|
| 0 | 0 |

Dari contoh pada tabel 9 diatas pada kalimat tersebut tidak memiliki punctuation dan subclause maka nilai komponen pada punctuation dan subclause masing-masing adalah 0.

### 3.3 *Split Data*

dataset dibagi menjadi dua bagian yaitu data *train* dan data *test.* Data train dilatih terlebih dahulu sebagai acuan untuk pengujian data *test*. Pada penelitian ini, split data menggunakan *10-Fold Cross Validation.*

**Tabel 10.** Split Data 10 Fold

| Sampel | Dataset | |
|---|---|---|
| | *Premise* | *Claim* |
| A | 1-104 | 1-53 |
| B | 105-208 | 54-106 |
| C | 209-312 | 107-159 |
| D | 313-416 | 160-212 |
| E | 417-520 | 213-265 |
| F | 521-624 | 266-318 |
| G | 625-728 | 319-371 |
| H | 729-832 | 372-424 |
| I | 833-936 | 425-477 |
| J | 937-1033 | 478-510 |

Pada tabel 10 menampilkan cara pembagian data 10 fold. Dari tabel diatas diambil data *premise* dan *claim* akan dikelompokkan menjadi satu sampel.

### 3.4 Cara Kerja Klasifikasi *Naïve Bayes*

**Tabel 11.** Contoh Perhitungan *Naïve bayes* pada fitur *Lexical*

| modal | verb | adverb | label | ket |
|---|---|---|---|---|
| 1 | 1 | 1 | claim | train |
| 1 | 0 | 1 | claim | train |
| 0 | 0 | 1 | claim | train |
| 1 | 1 | 0 | premise | train |
| 1 | 0 | 0 | premise | train |
| 1 | 1 | 1 | | test |

1. Hitung Prior probability
   Jumlah data train = 5
   Prior *premise* = 2/5
   Prior *claim* = 3/5

2. Hitung Conditional Probability
   Jumlah data kelas *premise* = 2
   Jumlah data kelas *claim* = 3

| | | | |
|---|---|---|---|
| P(modal = 0 \| *premise*) | = 0/2 | P(modal = 1 \| *premise*) | = 2/2 |
| P(verb = 0 \| *premise*) | = 1/2 | P(verb = 1 \| *premise*) | = 1/2 |
| P(adverb = 0 \| *premise*) | = 2/2 | P(adverb = 1 \| *premise*) | = 0/2 |
| | | | |
| P(modal = 0 \| *claim*) | = 1/3 | P(modal = 1 \| *claim*) | = 2/3 |
| P(verb = 0 \| *claim*) | = 2/3 | P(verb = 1 \| p *claim*) | = 1/3 |
| P(adverb = 0 \| *claim*) | = 3/3 | P(adverb = 1 \| *claim*) | = 0/3 |

3. Menghitung probabilitas dari masing masing data test *premise* dan *claim*
4. Dari hasil probabilitas yang didapat menghitung posterior probabilitynya
5. Dan akan didapatkan hasil test

**Tabel 12.** Contoh Perhitungan *Naïve bayes* pada fitur *Contextual*

| punctuation | subkalusa | label | ket |
|---|---|---|---|
| 2 | 1 | claim | train |
| 1 | 0 | claim | train |
| 1 | 1 | claim | train |
| 1 | 2 | premise | train |
| 0 | 2 | premise | train |
| 1 | 2 | | test |

1. Hitung Prior probability
   Jumlah data train = 5
   Prior *premise* = 2/5
   Prior *claim* = 3/5

2. Hitung Conditional Probability
   Jumlah data kelas *premise* = 2
   Jumlah data kelas *claim* = 3

| | | | |
|---|---|---|---|
| P(punctuation = 0 \| *premise*) | = ½ | P(punctutation = 1 \| *claim*) | = ½ |
| P(subkalusa = 0 \| *premise*) | = 0/2 | P(subklausa = 1 \| *claim*) | = 0/2 |
| P(punctuation = 2 \| *premise*) | = 0/2 | | |
| P(subklausa = 2 \| *premise*) | = 2/2 | | |
| P(punctuation = 0 \| *claim*) | = 0/3 | P(punctutation = 1 \| *claim*) | = 2/3 |
| P(subkalusa = 0 \| *claim*) | = 1/3 | P(subklausa = 1 \| *claim* | = 2/3 |
| P(punctuation = 2 \| *claim*) | = 1/3 | | |
| P(subklausa = 2 \| *claim*) | = 0/3 | | |

3. Menghitung probabilitas dari masing masing data test *premise* dan *claim*
4. Dari hasil probabilitas yang didapat menghitung posterior probabilitynya
5. Dan akan didapatkan hasil test

## 4. Pengujian dan Analisis

Pada penelitian ini tujuan akan tercapai saat dilakukannya klasifikasi komponen pada kalimat 10cenario de-ngan label *premise* dan *claim*, untuk mendapatkan hasil terbaik maka dibuat 10cenario pengujian untuk dijadikan perbandingan evaluasi model,sebagai berikut :

1. Scenario dengan menghitung performa model menggunakan ektraksi fitur *lexical*, bertujuan untuk mengetauhi hasil tingkat akurasi dan *f1 score* pada fitur *lexical*
2. Scenario dengan menghitung perorma model menggunakan ektraksi fitur *contextual*, bertujuan untuk mengetauhi hasil tingkat akurasi dan *f1 score* pada fitur *contextual*
3. Scenario dengan menghitung perorma model menggunakan ektraksi fitur *lexical* dan *contextual*, yang bertuuan untuk mengetauhi hasil tingkat akurasi dan *f1 score lexical* dan *contextual* secara bersamaan

Pengujian ini dilakukan dengan beberapa tahap, tahap pertama yaitu *case folding*. Kemudian tahap selanjutnya adalah mela-kukan *preprocessing* pada dataset, setelah mendapatkan hasil *preprocessing* maka akan digunakan ekstraksi fitur *lexical* dan *contextual*. Kemudian dari hasil fitur tersebut dilakukan k-fold cross validation dan dilakukan klasifikasi oleh model yang telah dibuat yaitu na¨ıve bayes.

Berikut ini adalah hasil pengujian scenario 1 dalam bentuk chart pada Tabel :

**Tabel 13**. Hasil ekstraksi fitur *Lexical*

| No | Pengujian | *Accuracy* | F1-Score |
|---|---|---|---|
| 1 | tanpa kategori dan tanpa *laplace smoothing* | 66.31% | 79.55% |
| 2 | dengan kategori dan tanpa *laplace smoothing* | 66.44% | 79.80% |
| 3 | tanpa kategori dan dengan *laplace smoothing* | 66.31% | 79.55% |
| 4 | dengan kategori dan dengan *laplace smoothing* | 66.44% | 79.80% |

Berdasarkan Tabel 13 diatas maka penggunaan ektraksi fitur *lexical* dengan K = 10 lebih akurat pada penggunaan kategori dan laplace smoothing dibandingkan yang lain. Namun penggunaan laplace smoothing tidak terlalu efektif pada fitur ini karena tidak ada perbedaannya. Nilai yang didapat sebesar rata-rata *accuracy* 66.44% dan F1-score 79.80% .Hal tersebut dikarenakan pengkategorian lebih detail dalam perhitungan pada *preprocessing* dibandingkan dengan tidak menggunakan pengkategorian. Penggunaan *accuracy, precission, recall,* dan *f1-score* agar nilai dari *confusion matrix* dapat dinilai dan dibandingkan.

Berikut ini adalah hasil pengujian scenario 2 dalam bentuk chart pada Tabel :

**Tabel 14**. Hasil ekstraksi fitur *Contextual*

| No | Pengujian | *Accuracy* | F1-Score |
|---|---|---|---|
| 1 | tanpa kategori dan tanpa *laplace smoothing* | 66.24% | 79.11% |
| 2 | dengan kategori dan tanpa *laplace smoothing* | 66.38% | 79.12% |
| 3 | tanpa kategori dan dengan *laplace smoothing* | 66.44% | 79.80% |
| 4 | dengan kategori dan dengan *laplace smoothing* | 66.44% | 79.80% |

Berdasarkan Tabel 14 diatas maka penggunaan ektraksi fitur *contextual* dengan K = 10 pada penggunaan kategori dan tanpa kategori memiliki hasil yang sama. Jadi penggunaan pengkategorian dan penggunaan *laplace smoothing* tidak terlalu efektif pada fitur ini karena tidak ada perbedaannya. Nilai yang didapat sebesar rata-rata *accuracy* 66.44% dan F1-score 79.80% . Penggunaan *accuracy, precission, recall,* dan *f1-score* agar nilai dari *confusion matrix* dapat dinilai dan dibandingkan.

Berikut ini adalah hasil pengujian scenario 3 dalam bentuk chart pada Tabel :

**Tabel 15**. Hasil ekstraksi fitur *Lexical* dan fitur *Contextual*

| No | Pengujian | *Accuracy* | F1-Score |
|---|---|---|---|
| 1 | *contextual* tanpa kategori, *lexical* tanpa kategori, dan tanpa *laplace smoothing* | 66.84% | 79.45% |
| 2 | *contextual* tanpa kategori, *lexical* tanpa kategori, dan dengan *laplace smoothing* | 66.38% | 79.60% |
| 3 | *contextual* tanpa kategori, *lexical* dengan kategori, dan tanpa *laplace smoothing* | 66.44% | 79.80% |

| | | | |
|---|---|---|---|
| 4 | *contextual* tanpa kategori, *lexical* dengan kategori, dan dengan laplace smoothing | 66.44% | 79.80% |
| 5 | *contextual* dengan kategori, *lexical* tanpa kategori, dan tanpa laplace smoothing | 66.64% | 79.71% |
| 6 | *contextual* dengan kategori, *lexical* tanpa kategori, dan dengan laplace smoothing | 66.64% | 79.71% |
| 7 | *contextual* dengan kategori, *lexical* dengan kategori, dan tanpa laplace smoothing | 66.44% | 79.80% |
| 8 | *contextual* dengan kategori, *lexical* dengan kategori, dandengan laplace smoothing | 66.44% | 79.80% |

Berdasarkan Tabel 15 diatas maka penggunaan ektraksi fitur *lexical* dan *contextual* dengan K = 10 lebih untuk akurasi nilai tertinggi pada pengujian no.1 dengan nilai *accuracy* 66.84% sedangkan untuk f1-score nilai tertinggi dipegang pada pengujian no.3,4,7, dan 8. Jadi dalam konteks ini penggunaan *contextual* tanpa kategori dan *lexical* tanpa kategori dapat menghasilkan nilai *accuracy* yang besar dibandingkan dengan yang lain. Sedangkan untuk f1-score nilai tinggi didapat saat pengujian menggunakan *lexical* dengan kategori. Karena selalu menghasilkan nilai f1-score 79.80%. Penggunaan *accuracy, precission, recall,* dan *f1-score* agar nilai dari *confusion matrix* dapat dinilai dan dibandingkan.

## 5. Kesimpulan

Berdasarkan hasil pengujian dan analisis yang telah dilakukan, dapat diambil beberapa kesimpulan sebagai berikut :

1. Skenario pengujian akan menghasil kan performa yang berbeda-beda karena dalam semua skenario pengujian yang dilakukan dapat mempengaruhi performa model yang dibangun, baik itu meningkatkan maupun menurunkan performa model.
2. Penggunaan pengkategorian dalam fitur *lexical* sangat berpengaruh untuk peningkatan nilai akurasi pada kasus ini
3. Penggunaan *laplace smoothing* tidak terlalu berpengaruh dalam peningkatan nilai akurasi pada kasus ini
4. Penggunaan pada fitur *contextual* sangat dipengaruhi adanya *preprocessing* karena tanda baca tidak dihilangkan pada saat *preprocessing* (*case folding*)

**Daftar Pustaka**


[1] Berry, M.W.; Kogan, J. (2010). *Text Mining: Application and Theory*. Chichester: John Wiley & Sons, Ltd.

[2] D. Suhartono, "Klasifikasi Komponen Argumen Secara Otomatis pada Dokumen Teks berbentuk Esai Argumentatif," *Technical Report Program Studi Doktor Ilmu Komputer* , 2015.

[3] C. Stab, Argumentative Writing Support by means of Natural Language Processing, Darmstadt: TU Darmstadt, 2017

[4] C. Stab and I. Gurevych, "Identifying Argumentative Discourse Structures in Persuasive Essays," *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 46-56, 2014.

[5] C. Stab, Argumentative Writing Support by means of Natural Language Processing, Darmstadt: TU Darmstadt, 2017.

[6] C. Stab and I. Gurevych, "Annotating Argument Components and Relations in Persuasive Essays," *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pp. 1501-1510, 2014.

[7] D. Satria and Mushthofa, "Perbandingan Metode Ekstraksi Ciri Histogram dan PCA untuk Mendeteksi Stoma pada Citra Penampang Daun Freycinetia," *Ilmu Komputer Agri-Informatika*, vol. 2, pp. 20-28, 2013.

[8] A. Knott and R. Dale, *Using Linguistic Phenomena to Motivate a Set of Rhetorical*, pp. 1-30, 5 Juli 1993.

[9] M.-F. Moens, R. M. Palau, E. Boiy and C. Reed, "Automatic Detection of Arguments in Legal Texts," *The 11th International Conference on Artificial Intelligence and Law*, pp. 225-230, 2007.

[10] I. Gurevych, "Argument Annotated Essay," 16 May 2017. [Online]. Available: https://www.informatik.tu-darmstadt.de/ukp/research_6/data/argumentation_mining_1/argument_annotated_essays/index.en.jsp . [Accessed 11 April 2018].

[11] Suyanto, Data Mining: Untuk Klasifikasi dan Klasterisasi Data, Bandung: Informatika, 2017.

[12] P.Goyal. What is laplace smoothing and why do we need it in a naive bayes classifier? https://www.quora.com/ What-is-Laplace-smoothing-and-why-do-we-need-it-in-a-Naive-Bayes-classifier, 2017. Online; Accessed 23 August 2018.

[13] S. Raschka. Naive bayes and text classification i : Introduction and theory. https://arxiv.org/pdf/ 1410.5329.pdf, 2014. Online; Accessed 1 August 2018.