# bab ii

# landasan teori

## 2.1. Data *Mining*

Data *mining* merupakan serangkaian proses untuk menggali informasi dengan melakukan analisa data untuk menemukan suatu pola dari kumpulan data tersebut. Data mining mampu menganalisa data yang besar menjadi ekstraksi berupa pola yang mempunyai arti bagi pendukung keputusan (Gunadi and Sensuse, 2012). Data mining juga bisa disebut knowledge discovery adalah proses pengambilan pola pada data yang akan di proses lalu output tersebut berupa informasi yang sangat penting. Proses yang dilakukan untuk mengekstrak pengetahuan dalam data mining adalah pengenalan pola, clustering, asosiasi, prediksi dan klasifikasi (Fitri *et al.*, 2018). Data *mining* memiliki variasi untuk menemukan pola dari ekstraksi sebuah kumpulan sekumpulan data tekstual yang disebut dengan *text mining*. *Text mining* memiliki fokus pada pengolahan data berupa kata atau teks.

## 2.2. *Text Mining*

Dalam jurnal (Februariyanti, 2012), menurut Hearst *text mining* diartikan sebagai penemuan informasi yang baru dan tidak diketahui sebelumnya oleh komputer, dengan secara otomatis mengekstrak informasi dari sumber-sumber yang berbeda. Kunci dari proses ini adalah menggabungkan informasi yang berhasil diekstraksi dari berbagai sumber . Sedangkan menurut Harlian *text mining* memiliki definisi menambang data yang berupa teks dimana sumber data biasanya didapatkan dari dokumen, dan tujuannya adalah mencari kata-kata yang dapat mewakili isi dari dokumen sehingga dapat dilakukan analisa keterhubungan antar dokumen.

*Text mining* merupakan bagian dari *data mining*, yang mana digunakan untuk mendapatkan informasi dari sebuah data atau dokumen berupa sekumpulan teks yang memiliki format yang terstruktur ataupun tidak terstruktur dengan jumlah yang besar. Dalam *text mining* memiliki tugas khusus yaitu klasifikasi dan klasterisasi. Sedangkan dalam penerapannya, *text mining* berfungsi untuk mencari pola dalam teks, meganalisa teks agar bisa menghasilkan keluaran berupa informasi yang bermanfaat pada tujuan tertentu. Dikarenakan data yang diproses pada *text mining* merupakan sebuah teks yang tidak terstruktur, maka diperlukan pemilihan teks sebelum dilakukan proses selanjutnya, pada tahap ini dikenal dengan prapemrosesan (*preprocessing*).

## 2.3. Plagiarisme

Dalam jurnal (Sunyoto and Informatika, 2013). Plagiarisme atau plagiat adalah penjiplakan atau pengambilan karangan, pendapat orang lain dan menjadikannya seolah-olah karangan sendiri. Pendekatan deteksi plagiat terbagi menjadi *intrinsic* dan *external*. Pendekatan *external* terbagi lagi menjadi tiga, yaitu perbandingan teks lengkap, kesamaan kata kunci dan *fingerprinting*. Perbandingan teks lengkap diterapkan untuk membandingkan semua isi dokumen, kesamaan kata kunci bekerja dengan cara mengekstrak dan membandingkan kata kunci antar dokumen, dan *fingerprinting* untuk mendeteksi kemiripan antardokumen dengan prinsip *hashing.*

Berdasarkan penelitian ini mengkategorikan praktek plagiat berdasarkan cara yang digunakan, diantaranya :

* *Copy&Paste plagiarism*, menyalin setiap kata tanpa perubahan.
* *Disguised plagiarism*, tergolong kedalam praktek menutupi bagian yang disalin, teridentifikasi ke dalam empat teknik, yaitu *shake&paste, expansive plagiarism, contractive plagiarism,* dan *mosaic plagiarism.*
* *Technical disguise*, teknik meringkas untuk menyembunyikan konten plagiat dari deteksi otomatis dengan memanfaatkan kelemahan dari metode analisis teks dasar, misal dengan mengganti huruf dengan simbol huruf asing.
* *Undue paraphrasing*, sengaja menuliskan ulang pemikiran asing dengan pemilihan kata dan gaya plagiator dengan menyembunyikan sumber asli.
* *Translated plagiarism*, mengkonversi konten dari satu bahasa ke bahasa lain.
* *Idea plagiarism*, menggunakan ide asing tanpa menyatakan sumber.

*Self plagiarism*, penggunaan sebagian atau keseluruhan tulisan pribadi yang tidak dibenarkan secara ilmiah.

## 2.4. *Preprocessing*

Tahapan *preprocessing* atau praproses merupakan bagian yang sangat penting dalam menyiapkan data, hal ini dikarenakan struktur data yang dihasilkan pada tahap pengumpulan tidak beraturan, sehingga menyebabkan proses menjadi tidak berjalan dengan baik.

Merujuk pada penelitian sebelumnya yang dilakukan oleh Himalatha dalam jurnal (Filcha and Hayaty, 2019) maka pada penelitian ini akan dibahas beberapa tahapan *preprocessing* teks antara lain, *casefolding,* menghapus karakter selain a-z, menghapus teks dengan 1 karakter, mengganti *slangword,* menghapus *stopword,* dan *stemming.*

### 2.4.1. *Casefolding*

Pada proses ini bertujuan untuk mengubah semua karakter huruf menjadi huruf kecil (*lowercase*), hal ini dilakukan untuk menyamakan arti dari suatu kata yang sama, apabila penulisan besar kecilnya huruf tidak sama.

### 2.4.2. Menghapus karakter kecuali a sampai z

Pada proses ini dilakukan penghapusan untuk seluruh karakter berupa simbol dan angka, atau menyisakan hanya karakter angka, termasuk menghapus hashtag (#) dan mention (@), hal ini dilakukan karena simbol dan angka dianggap tidak terlalu penting, tetapi jika ini diperlukan, maka proses ini dihilangkan.

### 2.4.3. Menghapus teks dengan 1 karakter

Pada proses ini dilakukan penghapusan kata dengan jumlah hanya 1 karakter saja, penghapusan karakter ini bertujuan untuk mengurangi kata yang dianggap tidak memiliki arti.

### 2.4.4. Mengganti *slangword*

Teks yang tidak terstruktur membuat sebuah teks terkadang tidak sesuai dengan ejaan bahasa Indonesia yang baku (EYD) pada konteks ini, kata yang tidak baku disebut dengan *slangword,* untuk mendapatkan informasi dari teks agar maksimal, kata-kata tidak baku, baik kata gaul, singkatan atau yang lain sebanyak mungkin ditampung ke dalam kamus *slangword,* untuk kemudian dilakukan *replace* supaya menjadi kata dengan bahasa Indonesia yang baku sesuai EYD.

### 2.4.5. Menghapus *stopword*

*Stopword* merupakan salahsatu kata yang diabaikan dalam pemrosesan, Ringkasnya *stopword* adalah kata hubung atau kata sambung dalam sebuah kalimat, seperti “di”, “pada”, “karena”, “sebuah”, “oleh”, dll. Sebelum melakukan proses penghapusan *stopword*, kumpulkan daftar atau kamus *stopword* yang diberi nama *stoplist*. Kemudian lakukan perbandinngan antara sebuah teks dengan *stoplist*. Jika terdapat kata-kata yang terdapat dalam *stoplist*, maka kata tersebut dihilangkan. Untuk *stoplist* dalam bahasa Indonesia, datanya bersumber dari (Tala, 2003).

### 2.4.6. *Stemming*

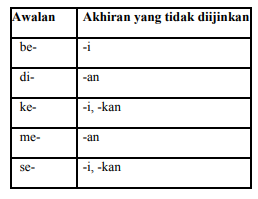
*Stemming* adalah proses pemotongan (pembuangan) imbuhan *(affix),* baik *prefix* maupun *suffix*, dari sebuah *term* untuk mendapatkan kata dasar *(root* atau *stem)* dari kata yang berimbuhan (Wahyudi *et al.*, 2017). Dalam penelitian ini proses stemming menggunakan *library* Sastrawi, algoritme *stemming Nazief* dan Adriani tahun 1996, algoritme ini dikembangkan berdasarkan aturan morfologi Bahasa Indonesia yang mengelompokkan imbuhan menjadi awalan *(prefix)*, sisipan *(infix)*, akhiran *(suffix)* dan gabungan awalan akhiran *(confixes)*. Berikut contoh *stemming* pada Tabel 2.1 yang menjelaskan perubahan pada kata yang memiliki imbuhan, kemudian dilakukan proses *stemming* sehingga menghasilkan sebuah kata dasar.

Algoritme yang dibuat oleh Bobby Nazief dan Mirna Adriani ini memiliki tahap-tahap sebagai berikut:

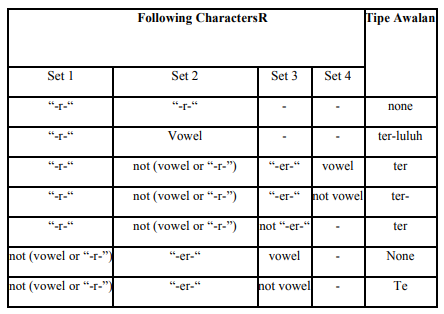
1. Cari kata yang akan distem dalam kamus. Jika ditemukan maka diasumsikan bahwa kata tesebut adalah *root word*. Maka algoritme berhenti.
2. *Inflection Suffixes* (“-lah”, “-kah”, “-ku”, “-mu”, atau “-nya”) dibuang. Jika berupa *particles* (“-lah”, “-kah”, “-tah” atau “-pun”) maka langkah ini diulangi lagi untuk menghapus *Possesive Pronouns* (“-ku”, “-mu”, atau “-nya”), jika ada.
3. Hapus *Derivation Suffixes* (“-i”, “-an” atau “-kan”). Jika kata ditemukan di kamus, maka algoritme berhenti. Jika tidak maka ke langkah c1.
   1. Jika “-an” telah dihapus dan huruf terakhir dari kata tersebut adalah “-k”, maka “-k” juga ikut dihapus. Jika kata tersebut ditemukan dalam kamus maka algoritme berhenti. Jika tidak ditemukan maka lakukan langkah c2.
   2. Akhiran yang dihapus (“-i”, “-an” atau “-kan”) dikembalikan, lanjut ke langkah d.
4. Hapus *Derivation Prefix*. Jika pada langkah c ada *sufiks* yang dihapus maka pergi ke langkah d1, jika tidak pergi ke langkah d2.
   1. Periksa tabel kombinasi awalan-akhiran yang tidak diijinkan. Jika ditemukan maka algoritme berhenti, jika tidak pergi ke langkah d2.
   2. For i = 1 to 3, tentukan tipe awalan kemudian hapus awalan. Jika *root word* belum juga ditemukan lakukan langkah e, jika sudah maka algoritme berhenti. Catatan: jika awalan kedua sama dengan awalan pertama algoritme berhenti.
5. Melakukan *Recoding.*
6. Jika semua langkah telah selesai tetapi tidak juga berhasil maka kata awal diasumsikan sebagai *root word*. Proses selesai.

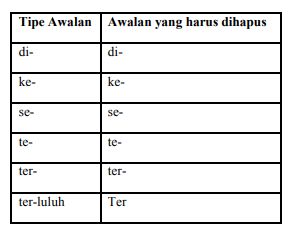
Tipe awalan ditentukan melalui langkah-langkah berikut:

* + 1. Jika awalannya adalah: “di-”, “ke-”, atau “se-” maka tipe awalannya secara berturut-turut adalah “di-”, “ke-”, atau “se-”.
    2. Jika awalannya adalah “te-”, “me-”, “be-”, atau “pe-” maka dibutuhkan sebuah proses tambahan untuk menentukan tipe awalannya.
    3. Jika dua karakter pertama bukan “di-”, “ke-”, “se-”, “te-”, “be-”, “me-”, atau “pe-” maka berhenti.
    4. Jika tipe awalan adalah “none” maka berhenti. Jika tipe awalan adalah bukan *“none”* maka awalan dapat dilihat pada Tabel 2.2 Hapus awalan jika ditemukan.
    5. Pada Tabel 2.3 merupakan daftar awalan berdasarkan tipe awalan dan awalan yang harus dihapus.

Tabel 1 2.1 Kombinasi Awalan Akhiran Yang Tidak Diijinkan

Tabel 2 2.2 Cara Menentukan Tipe Awalan Untuk Kata Yang Diawali “te-”



Tabel 3 2.3 Jenis Awalan Berdasarkan Tipe Awalannya

Untuk mengatasi keterbatasan pada algoritme di atas, maka ditambahkan aturan-aturan dibawah ini:

* + - * 1. Aturan untuk *reduplikasi.*
  1. Jika kedua kata yang dihubungkan oleh kata penghubung adalah kata yang sama maka *root word* adalah bentuk tunggalnya, contoh : “buku-buku” *root word*-nya adalah “buku”.
  2. Kata lain, misalnya “bolak-balik”, “berbalas-balasan, dan ”seolah-olah”. Untuk mendapatkan *root word*-nya, kedua kata diartikan secara terpisah. Jika keduanya memiliki *root word* yang sama maka diubah menjadi bentuk tunggal, contoh: kata “berbalas-balasan”, “berbalas” dan “balasan” memiliki *root word* yang sama yaitu “balas”, maka *root word* “berbalasbalasan” adalah “balas”. Sebaliknya, pada kata “bolak-balik”, “bolak” dan “balik” memiliki *root word* yang berbeda, maka *root word*-nya adalah “bolak-balik”.
     + - 1. Tambahan bentuk awalan dan akhiran serta aturannya.
  3. Untuk tipe awalan “mem-“, kata yang diawali dengan awalan “memp-” memiliki tipe awalan “mem-”.
  4. Tipe awalan “meng-“, kata yang diawali dengan awalan “mengk-” memiliki tipe awalan “meng-”.

### 2.4.7. *Tokenization*

Pada proses ini dilakukan pemisahan sebuah kalimat ke dalam bentuk pencahan kata. Kata-kata yang telah dipisahkan akan dimasukan ke dalam sebuah *index array* untuk digunakan pada tahap berikutnya, dalam penelitian ini proses *tokenizazion* termasuk ke dalam bagian *N-Gram* .

## 2.5. *N-Gram*

Dalam jurnal (Sunardi *et al.*, 2018)menjelaskan bahwa, *n-gram* merupakan metode yang yang dilakukan dengan mengambil rangkaian *substring* dari *string* sejumlah n (rangkaian token sepanjang n). Metode *n-gram* sering digunakan pada teknik analisis statistik dan juga bahasa. *N-gram* paling banyak digunakan dalam teks mining (pengolahan kata) dan pengolahan bahasa. Dalam mendeteksi plagiarisme, *n-gram* digunakan untuk mengambil potongan-potongan karakter huruf atau pemisahan *string* sepanjang n dari sebuah kata atau dokumen secara berkelanjutan (kontinuitas) hingga bergeser sesuai dengan *offsite* yang diberikan atau akhir dari suatu kata atau dokumen.

*N-gram* dalam deteksi plagiarisme sangat mempengaruhi tingkat akurasi atau tingkat *similarity* dari sebuah dokumen yang dibandingkan, maka teknik *n-gram* dipadukan dengan pendekatan statistika untuk memperoleh *similarity* dari antar dokumen seperti *Simple Matching, Cosine Similarity, Jaccard Similarity* dan *Dice Coefficient*. Ada 2 (dua) teknik *n-gram* yaitu membagi *string* menjadi suatu set *substring* dengan panjang n (*overlapping n-gram*) dan mengecek untuk membentuk subtstring yang memiliki struktur yang sama.

## 2.6. *Jaccard Similarity*

Dalam jurnal (Sunardi *et al.*, 2018)menjelaskan bahwa, *Jaccard Similarit*y atau *Jaccard Coefficient* merupakan algoritme yang fungsinya untuk membandingkan dua *sample* yaitu dokumen yang satu dengan yang lainnya berdasarkan kata yang dimilikinya. *Jaccard similarity* biasanya digunakan untuk membandingkan dokumen dan menghitung nilai kemiripan (*similarity*) dari dua buah objek atau dokumen. *Jaccard similarity* dapat dirumuskan sebagai berikut:

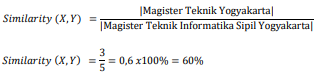


Dimana:

X = Dokumen 1

Y = Dokumen 2

Rumus 1 merupakan dari rumus *Jaccard Similarity* atau *Jaccard Coefficient* yang digunakan untuk mencari persamaan dan perbedaan pada dua sample. sebagai contoh diketahui X “Magister Teknik Informatika Yogyakarta”, dan Y “Magister Teknik Sipil Yogyakarta”. Maka akan menghasilkan nilainya:



Dari dua contoh *sample* di atas setelah dihitung kesamaannya menggunakan *Jaccard similarity*, bahwa kedua *sample* tersebut memiliki kesamaan atau kemiripan sebesar 60%.

Ukuran tingkat kesamaan:

* a. 0% : dua dokumen tidak memiliki kesamaan
* b. <15% : memiliki sedikit kesamaan
* c. 15-50% : termasuk dalam katagori plagiarisme sedang
* d. 50% : mendeteksi adanya plagiarisme
* e. 100% : dokumen tersebut plagiarism

## 2.7. Algoritme *Winnowing*

Dalam jurnal (Sunardi *et al.*, 2018)menjelaskan bahwa, Algoritme *winnowing* merupakan salah satu algoritme yang berfungsi sebagai dokumen *fingerprint* atau algoritme yang digunakan untuk mendeteksi tindakan plagiarisme dengan menggunakan teknik *hashing*. Input dari algoritme *winnowing* berupa dokumen teks, dan akan menghasilkan keluaran berupa kumpulan nilai hash yang terbentuk dari peritungan ASCII pada setiap karakter. Dan nilai-nilai *hash* yang akan digunakan sebagai *fingerprint* untuk mendeteksi adanya suatu tindakan plagiarisme. Gambar 2.1 merupakan konsep algoritme *winnowing*.

Delete kata yang tidak diperlukan

Membentuk *n-gram*

Penghitung *substring* menjadi nilai *hash*

*Window*

*Fingerprint*

Gambar 1 2.1 Konsep Algoritme *Winnowing*

Konsep dari algoritme *winnowing* yang ditujukan pada gambar 2.1 yaitu menghapus karakter yang tidak relevan, membentuk *n-gram* dengan panjang n, menghitung nilai *hash*, membentuk nilai *window*, dan memilih nilai *hash* sebagai dokumen *fingerfrint*. Nilai *hash* didapatkan dari rumus *Rolling hash*. *Rolling hash* adalah metode *hashing* yang digunakan untuk mencari nilai *hash* dari rangkaian *grams* yang telah terbentuk dan memberikan kemampuan untuk menghitung nilai tanpa mengulangi seluruh *string*. Nilai *hash* merupakan nilai numerik yang dibentuk dari kode ASCII[15]. Berikut rumus *rolling hash*



Dimana:

H (C1..Cl) = nilai hash

Cl = nilai ASCII karakter ke -1 pada string

l = panjang string

b = nilai basis hash

Perhitungan awal pada rangkaian *n-gram* paling awal dihitung menggunakan rumus nomor 2, dan rangkaian *n-gram* berikutnya sampai rangkaian *gram* terakhir di hitung menggunakan rumus nomor 3, sehingga proses akan jauh lebih cepat karena tidak menghitung lagi dari awal. Algoritme *winnowing* merupakan eksitensi dari algoritme *Rabin-karp*, proses yang dilakukan hampir sama hanya pada algoritme *winnowing* ditambahkan dengan konsep *window*.

## 2.8. Pengujian

Dalam mengevaluasi performa dari metode yang diusulkan, perlu adanya sebuah pengujian dengan membandingkan hasil dengan data asli yang sudah valid. Dalam penelitian ini, dilakukan pengujian secara *recall* yaitu menentukan tingkat keberhasilan sistem dalam menemukan kembali sebuah informasi sebenarnya.

## 2.9. Studi Literatur

Berdasarkan landasan teori yang telah dijelaskan , terdapat penelitian yang sudah ada sebelumnya, yang di rangkum dalam Tabel 2.4 berikut :

Tabel 4 2.4 Studi Literatur

| **No** | **Penulis** | **Judul** | **Jurnal** | **Deskripsi** |
| --- | --- | --- | --- | --- |
| 1 | Abdullah, Ibnu Aribowo, Eko | Rancang Bangun Aplikasi Pengecekan Kemiripan Judul Skripsi Dengan Metode *Cosine Similarity* (Studi Kasus : Program Studi Teknik Informatika Uad) | JSTIE (Jurnal Sarjana Teknik Informatika) (E-Journal), Tahun 2018, *e-ISSN 2338-5197* | Melakukan pengecekan kemiripan judul skripsi di teknik informarika UAD. Dengan menggunakan metodologi pengembangan sistem yaitu *waterfall model*. Metode *waterfall* dimulai dengan tahap menentukan *system requirement specification,* tahap perancangan sistem, tahap implementasi, dan tahap pengujian sistem. Sistem diuji dengan 2 metode yaitu *Blackbox Test* dan *Usability Test.* Pengujian dengan metode *Blackbox Test* dihasilkan bahwa 100% sistem sudah berjalan dengan baik, layak digunakan dan sesuai kebutuhan. Sedangkan pengujian dengan metode *Usability Test* 0 % responden menyatakan aplikasi tidak diterima, 20 % responden memberikan penilaian marginal dan 80 % responden menyatakan sistem aplikasi bisa diterima. |
| 2 | Khidfi, Muhammad Nur Sari, Jayanti Yusma | Rancang bangun aplikasi pendeteksian kesamaan pada dokumen teks menggunakan algoritme *Enhaced confix stripping* dan algoritme *winnowing* | Jurusan Teknik Informatika, Fakultas Teknik Universitas Halu Oleo, Kendari, Tahun 2018, *ISSN: 2502-8928* | Melakukan pengecekan plagiarisme pada dokumen tugas mahasiswa. Pada penelitian ini menggunakan algoritme *Enhanced Confix Stripping (ECS) Stemmer* untuk proses *stemming* teks dan algoritme *Winnowing* untuk menghitung tingkat kesamaan *(similarity)* antar dokumen. Dengan ditentukannya nilai *gram* dan *window* pada perhitungan algoritme *Winnowing,* memudahkan user dalam menentukan nilai *similarity* yang akurat. Dari hasil pengujian 5 pasang bab 1 tugas akhir mahasiswa yang berkategori sama menghasilkan nilai *similarity* sekitar 45-20% . |
| 3 | Ariantini, Dewa Ayu Rai Lumenta, Arie S. M. Jacobus, Agustinus | Pengukuran Kemiripan Dokumen Teks Bahasa Indonesia Menggunakan Metode *Cosine Similarity* | Jurnal Teknik Informatika, Tahun 2016, *ISSN : 2301-8364* | Melakukan pengecekan plagiat suatu karya tulis. Tahap pengujian sistem dilakukan untuk evaluasi perangkat lunak yang dibagun dengan hasil yang diharapkan, dapat dilakukan dengan cara membandingkan nilai hasil prediksi dan *actual*, nilai prediksi diperoleh dari sistem, kemudian nilai *actual* diproleh dari nilai yang diperiksa secara manual, kemudian nilai yang sudah diketahui prediksi dan actual dengan menggunakan data *dummy* pada sistem deteksi kemiripan dokumen sudah berhasil diterapkan dan sudah bisa digunakan untuk mengukur tingkat kemiripan dokumen. |
| 4 | Fataruba, Firmansyah | Penerapan Metode *Cosine Similarity* Untuk Pengecekan Kemiripan Jawaban Ujian Siswa | Jurnal Mahasiswa Teknik Informatika, Tahun 2018 | Melakukan pengecekan kemiripan jawaban *essay* siswa. Tujuan dari penelitian ini adalah untuk membantu pengajar memberikan nilai yang objektif dengan mengunakan metode *cosine similarity* pada sistem agar dapat melakukan penilaian jawaban *essay* dengan membandingkan kunci jawaban pengajar dengan jawaban peserta didik. Dan juga dapat membandingkan tingkat kemiripan antara siswa satu dengan siswa yang lain. *Cosine similarity* adalah metode untuk menghitung kesamaan dan kemiripan dari dua dokumen, dokumen yang di hitung perbandinganya dalam penelitian ini adalah kunci jawaban pengajar dan jawaban siswa. Berdasarkan hasil penelitian bahwa sistem pengecekan kemiripan jawaban siswa menggunakan metode *cosine similarity* telah berjalan dengan baik untuk ujian *essay* Biologi Hasil uji coba menunjukkan kesesuaian nilai sistem dengan nilai yang diberikan oleh pengajar tingkat akurasinya 80% mengunakan *Confusion Matrix.* |
| 5 | Ulfa, Nur Fadillah Mustikasari, Metty Bastian, Irwan | Pendeteksian tingkat similaritas dokumen berbasis web menggunakan algoritme *winnowing* | Konferensi Nasional Teknologi Informasi dan Komunikasi (KNASTIK), Tahun 2016, *ISSN: 2338-7718* | Melakukan pendeteksian *plagiarisme* mengunakan teknik *winnowing.* Penelitian ini bertujuan untuk membangun sebuah aplikasi berbasis website menggunakan algoritme *Winnowing* untuk mencari kesamaan pada dua dokumen teks yang diuji. Algoritme yang digunakan untuk mencari nilai *hash* dalam *Winnowing* adalah *rolling hash*. Nilai *hash* merupakan nilai numerik yang terbentuk dari perhitungan ASCII (*American Standard Code for Information Interchange*) tiap karakter. Pada hasil eksperimen pengukuran kemiripan ini disimpulkan bahwa semakin kecil tingkat *persentase* kesamaan dokumen teks yang diuji, maka tingkat kemiripan dokumen kecil dan tidak termasuk plagiat, tetapi jika hasil dari pengujian pada dua dokumen semakin besar, maka dokumen tersebut mempunyai tingkat kemiripan yang tinggi dan tindakan tersebut dianggap plagiat. Penelitian ini juga menambahkan perbandingan nilai *k-gram*, basis (bilangan prima), nilai *window*, keterangan *persentase*, dan kategori *plagiarisme.* |
| 6 | Priambodo, Joko | Pendeteksian Plagiarisme Menggunakan Algoritme *Rabin-Karp* dengan Metode *Rolling Hash* | Jurnal Informatika Universitas Pamulang, Tahun 2018, *ISSN 2541-1004* | Melakukan pendeteksian kemiripan suatu dokumen. Pendeteksian plagiarisme menggunakan algoritme *Rabin-Karp* dengan metode *rolling hash* dari hasil pengujian 30 dokumen teks yang sudah dijelaskan pada bab sebelumnya menghasilkan tingkat akurasi yang terbesar yaitu 47.58%. Hasil persentase tersebut termasuk dalam kategori tingkat plagiat 15-50%, berarti menandakan dokumen tersebut termasuk plagiat tingkat sedang. Sedangkan tingkat akurasi yang terkecil yaitu 19.28%, berarti menandakan dokumen tersebut termasuk plagiat tingkat sedang. Berdasarkan analisis proses pendeteksian tingkat plagiarisme menggunakan algoritme raibin-karp dengan metode rolling hash bisa membaca karakter berupa huruf, simbol seperti titik (.), koma (,), dan lain-lain. |
| 7 | Nurdin, Nurdin Rizal, Rizal Rizwan, Rizwan | Pendeteksian Dokumen Plagiarisme dengan Menggunakan Metode *Weight Tree* | Telematika, Tahun 2019, *ISSN : 1979 – 925X e-ISSN : 2442 - 4528* | Melakukan pendeteksian plagiat suatu dokumen dengan menggunakan algoritme *Weight Tree,* algoritme *Weight Tree* yaitu sebuah metode untuk melakukan klasifikasi kemiripan dokumen berdasarkan bobot dari dokumen. Tujuan penelitian ini adalah untuk membangun sebuah sistem pendeteksian kemiripan dari dua dokumen teks yang berbeda untuk jenis dokumen teks berbahasa indonesia dengan format file dokumen yaitu: doc, docx, pdf, rtf. Tahapan yang dilakukan pada penelitian ini terdiri dari pengumpulan data, perancangan sistem, pembuatan aplikasi dan pengujian terhadap aplikasi. Hasil pengujian sistem dapat dikategorikan sebagai sistem pendeteksian atau pengetesan kemiripan dokumen. Nilai rata-rata persentase kemiripan dalam pengujian sistem ini adalah 71,60%. Sistem yang di bangun ini berhasil dengan tingkat keakuratan mencapai 90%. Algoritme *Weight Tree* yang diterapkan pada sistem ini terbukti mampu mengidentifikasi dengan baik kemiripan dokumen plagiarisme. |
| 8 | Rinartha, Komang | Pemodelan Penilaian *Essay* Otomatis Secara *Realtime* Menggunakan Kombinasi *Text Stemming* Dan *Cosine Similarity* | Konferensi Nasional Sistem & Informatika, Tahun 2017 | Melakukan pengecekan ujian siswa yang berbentuk *essay,* soal ujian yang akan dikembangkan adalah soal ujian berbasis web yang bisa memberikan nilai secara otomatis. Pemberian nilai otomatis dilakukan dengan menggunakan *stemming Porter* yang dikombinasikan dengan *Cosine similarity*. Algoritme *Porter* digunakan untuk mencari kata dasar dari kata-kata yang digunakan dalam kunci jawaban. Hasil dari *ekstraksi* kata dasar pada kunci jawaban dan jawaban siswa akan dibandingkan dengan menggunakan *Cosine similarity*. Proses pemodelan dilakukan dengan mengembangkan dari model yang telah ada dengan menyesuaikan konsep realtime yang akan direncanakan. Model yang didapatkan adalah proses *text stemming* dan proses *ekstrak* kata dari jawaban dilakukan pada komputer *client* (pengajar dan siswa) dan tidak membebankan ke sever. |
| 9 | Sunardi, Sunardi Yudhana, Anton Mukaromah, Iif Alfiatul | Implementasi Deteksi Plagiarisme Menggunakan Metode *N-Gram* Dan *Jaccard Similarity* Terhadap Algoritme *Winnowing* | *Transmisi*, Tahun 2018, *ISSN 2407–6422* | Melakukan pengecekan kemiripan suatu dokumen yang terjadi akibat banyaknya orang mengerjakan tugas dengan meng*copy paste*  dari internet, tanpa menyebutkan sumbernya. Hal tersebut bisa diatasi secara dini menggunakan metode *n-gram* untuk mengambil potongan karakter huruf pada dokumen yang akan dicocokan dan *Jaccard similarity* untuk menghitung berapa persen kesamaan pada sebuah dokumen yang diproses menggunakan algoritme *winnowing.* Algoritme *winnowing* merupakan algoritme yang digunakan untuk mencocokan karakter huruf atau angka pada dua dokumen dengan teknik *hashing.* |
| 10 | Filcha, Asvarizal Hayaty, Mardhiya | Implementasi Algoritme *Rabin-Karp* untuk Pendeteksi Plagiarisme pada Dokumen Tugas Mahasiswa | JUITA : Jurnal Informatika, Tahun 2019, *ISSN: 2579-9801* | Melakukan pengecekan plagiarisme pada tugas mahasiswa, ini terjadi karena tugas yang dikumpulkan dalam bentuk digital. Metode yang dapat digunakan adalah menggunakan algoritme *Rabin-Karp.* Algoritme *Rabin- Karp* memiliki keunggulan pencarian *string* dengan pola yang panjang. Algoritme *Rabin-karp* dalam sistem ini memiliki langkah - langkah *text preprocessing* yang terdiri *case folding, tokenizing, punctuation removal, stopword removal* dan *stemming*. Hasil dari *text preprocessing* inilah yang akan di proses menggunakan algoritme *Rabin-karp*. Hasil dari metode ini dihitung menggunakan *dice coefficient*. Perhitungan akurasi dengan melakukan 20 perbandingan antara sistem pendeteksi plagiarisme dan *software* Plagiarisme Checker X menggunakan *confusion matrix* menghasilkan tingkat keakuratan sebesar 90%. |
| 11 | Anggara, Yudha | Deteksi plagiarisme dokumen bahasa indonesia dengan algoritme *jaro-winkler Distance* | Program studi teknik informatika universitas muhammadiyah jember, Tahun 2016 | Melakukan pengecekan tingkat kesamaan antar dokumen. Pada penelitian ini akan dibuat sebuah aplikasi untuk menghitung tingkat kesamaan dokumen dengan algoritme *Jaro-Winkler distance*. Tujuan dari penerapan algoritme ini adalah membandingkan kesamaan antar dokumen teks berbahasa Indonesia, sehingga dapat ditentukan sebuah dokumen tersebut plagiat atau tidak. Pengujian terhadap aplikasi menggunakan data abstrak jurnal skripsi. Dari hasil analisis dokumen uji 1 memiliki kesamaan tertinggi dengan dokumen nomer 55 dengan nilai 86,267%, dokumen 2 memiliki kesamaan tertinggi dengan dokumen 66 dengan nilai 95,922% dan dokumen 3 memiliki kesamaan tertinggi dengan dokumen 23 dengan nilai 98.361 %. |
| 12 | Alamanda, Rio Suhery, Cucu Brianorman, Yulrio | Aplikasi pendeteksi plagiat terhadap karya tulis berbasis web menggunakan natural language processing dan algoritme knuth-morris-pratt | Jurnal *Coding*, Sistem Komputer Untan, Tahun 2016, *ISSN : 2338-493x* | Melakukan pendeteksian kemiripan jurnal dimulai dari algoritme perangkuman, yaitu *TF/IDF*. Kemudian proses dilanjutkan pada proses *Tokenizing, Filtering* dan *Stemming*. Selanjutnya dilanjutkan dengan Algoritme *Knuth-Morris-Pratt* dan proses menghitung persentase kemiripan dengan *Cosine Similarity*. Dari hasil pengujian yang telah dilakukan, rata-rata persentase kemiripan yang dihasilkan dari pendeteksian tanpa menggunakan proses *TF-IDF* sebesar 45,98%, nilai tersebut lebih rendah dibandingkan dengan pendeteksian menggunakan proses *TF-IDF*, *Tokenizing, Filtering* dan *Stemming* yang menghasilkan persentase kemiripan 41,09%. Sedangkan tanpa menggunakan proses *Stemming* yaitu 40,58% serta tanpa menggunakan proses *TF-IDF* dan *Stemming* sebesar 40,54% |
| 13 | Shu, Kai Wang, Suhang Lee, Dongwon Liu, Huan | *Mining disinformation and fake news: Concepts, methods, and recent advancements* | *International Journal of Computer Applications,* Tahun 2020 | Melakukan pendeteksian membutuhkan metode untuk penyelesain masalhnya. Di penelitian ini membahas metode pengecekan kemiripan dokumen menggunakan algoritme berbasis string yang berbeda (*StringBased). String Based* ini di bagi menjadi 2, yaitu *Character Based* dan  *Term Based.*  Dari 2 pembagian ini ada beberapa metode yang bisa di gunakan, sesuai dengan kebutuhan. Untuk *Character Based*  pembagianya ada beberapa metode di antaranya, *Longest Common SubString (LCS), Longest Common SubString (LCS), Jaro, Jaro–Winkler, Needleman-Wunsch, Smith-Waterman, N-gram.* Untuk *Term Based* pembagianya ada, *Block Distance, Cosine similarity, Dice’s coefficient, Euclidean distance, Jaccard similarity, Matching Coefficient, Overlap coefficient.* |
| 14 | Wang, Jiapeng Dong, Yihong | *Measurement of Text Similarity: A Survey* | *Information*, Tahun 2020 | Melakukan pendeteksian kemiripan suatu dokumen pasti menggunakan suatu metode untuk penyelesaian suatu kasus. Penelitian ini menganalisis keuntungan dan kerugian dari suatu metode yang digunakan, mengembangkan sistem deskripsi klasifikasi yang lebih komprehensif dari algoritme pengukuran kesamaan teks, dan merangkum arah pengembangan di masa depan. Dengan tujuan memberikan referensi bagi penelitian dan aplikasi terkait, metode pengukuran kemiripan teks dijelaskan melalui dua aspek yaitu jarak teks dan representasi teks. Jarak teks dapat dibagi menjadi jarak panjang, jarak distribusi, dan jarak semantik; Representasi teks dibagi menjadi teks berbasis string, berbasis korpus, teks semantik tunggal, teks multi-semantik, dan representasi berbasis struktur grafik. Terakhir, perkembangan kemiripan teks juga dirangkum pada bagian pembahasan. |
| 15 | Mujilahwati | *PreProcessing*  *Text Mining*  Pada *Data*  *Twitter* | Seminar  Nasional  Teknologi  Informasi  Dan Komunikasi,  Tahun 2016,  *ISSN 2089-*  *9815* | Membahas teknik penanganan data prampemrosesan data komentar dari Twitter. Hasil ekstraksi ini kemudian diujikan pada pengklasifikasian layanan sebuah perusahaan telekomunikasi serta didapatkan hasil akurasi mencapai 93,11% dengan 450 data uji. |