Aplikasi Himpunan dalam Mendeteksi Duplikat pada Twitter

Athif Nirwasito - 13521053
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13521053@std.itb.ac.id

***Abstract*—*Tweet scraping* adalah sebuah metode ekstraksi data opini yang diperoleh dari media sosial Twitter. Namun, sebelum data dapat digunakan, data tersebut harus dibersihkan terlebih dahulu. Salah satu proses pembersihan data adalah penghapusan duplikat. Namun, proses pendeteksi duplikat pada *tweets* tidaklah mudah karena *tweets* bisa sama sebagian. Akibatnya, dibutuhkan langkah tambahan untuk mendeteksi semua duplikat dari data hasil *scraping*. Pada makalah, akan diuji dua metode pengukuran kesamaan string yang memanfaatkan himpunan untuk mendeteksi duplikat-duplikat tersebut, yaitu dengan index Jaccard serta koefisien tumpang tindih.**

***Keywords*—Himpunan, Tweet Cleaning, Overlap Coefficient, Jaccard Index**

I. PENDAHULUAN

Tweet scraping merupakan salah satu metode untuk menganalisa sentimen masyarakat mengenai suatu topik dengan cara mengumpulkan *tweet-tweet* yang berkaitan dengan topik tersebut. *Tweet* dapat dikumpulkan dengan menggunakan modul snscrape di python. Semua *tweet* yang dikumpulkan tersebut kemudian disimpan dalam sebuah dataset.

Sebelum semua dataset *tweet* tersebut dianalisa, dataset tersebut harus dibersihkan terlebih dahulu. Proses pembersihan tweet umumnya mencakup penghapusan karakter non-alfanumerik, dan lain-lain. Setelah proses tersebut dilakukan, isi dari semua *tweet* akan seragam sehingga kumpulan kata di dalam isi *tweet* dapat di tokenasi. Proses tersebut juga dilakukan sehingga tweet *spam* yang menvariasikan isinya melalui perubahan tanda baca atau huruf kapital dapat terdeteksi. Dalam library Pandas pada Python, umumnya duplikat dalam dataset dapat dihilangkan menggunakan fungsi "drop_duplicate()". Namun, fungsi tersebut hanya menghilangkan duplikat yang sama persis. Terdapat kasus *tweet* yang menyisipkan kata tambahan diluar isi pokok *tweet* sehingga terdapat content yang duplikat dari konten *tweet* lain, tetapi tidak sama persis.

Gambar1. Kasus Isi Sama dengan Sisipan kata yang Berbeda

Ada pula *tweet* yang memanfaatkan semua tag yang populer pada masa itu.

Gambar 2. Kasus Spam Tag

Kasus-kasus tersebut tidak dapat dihapus dengan fungsi "drop_duplicate". Oleh karena itu, dibutuhkan penghapusan duplikat berdasarkan sebuah persentase kesamaan. Beberapa cara umum untuk mengukur kesamaan dua buah string adalah dengan menggunakan algoritma berbasis jarak seperti jarak Levenshtein atau jarak Hamming, algoritma berbasis himpunan token seperti index Jaccard atau Sorensen-Dice, serta algoritma-algoritma lainnya. Algoritma perhitungan kesamaan string berbasis jarak bisa mendeteksi duplikat pada kasus pertama, namun tidak bisa mendeteksi kasus spam karena susunan tagnya diubah. Oleh karena itu, dibutuhkan algoritma pendeteksi duplikat atau spam yang bisa mendeteksi sebuah spam walaupun susunan katanya berbeda.

II. TEORI DASAR

A. Himpunan

Himpunan (*set*) adalah kumpulan objek yang berbeda. Objek tersebut disebut sebagai elemen. Semua elemen pada sebuah himpunan harus berbeda dengan satu sama lain. Himpunan yang memperbolehkan adanya dua atau lebih elemen yang sama disebut himpunan-ganda (*multi-set*). Terdapat tiga cara untuk menyatakan sebuah himpunan:

1) Enumerasi

Enumerasi merupakan penyajian himpunan dengan merincikan semua elemen himpunan. Contoh:

A ={“pajak”,”20”, “kucing”}

B ={4,6,8,10}

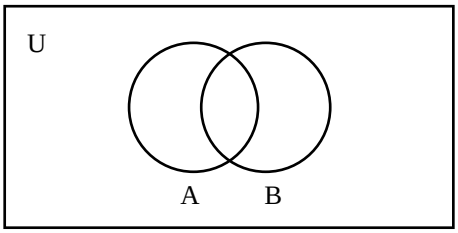
2) Notasi Pembentuk Himpunan

Himpunan dapat dinyatakan dengan notasi {x|syarat keanggotaan x}. Contohnya adalah A = {x|x bilangan bulat lebih besar dari 0} atau A = { x | x > 0 }.

3) Simbol-simbol Baku

P = himpunan bilangan bulat positif
N = Himpunan bilangan alamai
Z = Himpunan bilangan bulat positif
Q = Himpunan bilangan rasional
R = Himpunan bilangan riil
C = Himpunan bilangan kompleks
U = Semesta
∅ = Himpunan kosong

Selain notasi, sebuah himpunan juga dapat divisualisasi menggunakan diagram Venn. Setiap himpunan direpresentasikan sebagai sebuah lingkaran dengan elemen-elemenya di dalam lingkaran tersebut. Lingkaran-lingkaran tersebut berada dalam sebuah persegi panjang yang merepresentasikan himpunan semesta.



Gambar 3. Contoh Diagram Venn

Pada sebuah himpunan, x merupakan anggota dari A jika x adalah salah satu elemen dari A. Keanggotaan tersebut dapat dinyatakan sebagai x ∈ A yang berarti x adalah anggota dari A.

Pada dua buah himpunan A dan B, himpunan A dikatakan himpunan bagian dari himpunan B jika dan hanya jika semua anggota dari A merupakan anggota dari B juga. Secara formal, himpunan bagian dapat dinyatakan sebagai berikut:

$$A \subseteq B \leftrightarrow \forall x(x \in A \rightarrow x \in B)$$

Kardinalitas dari sebuah himpunan menyatakan jumlah elemen dari sebuah himpunan. Kardinal dari sebuah himpunan A dapat dinyatakan sebagai n(A) atau |A|. Contoh:

$$A = \{satu, dua, tiga\} \rightarrow |A| = 3$$
$$B = \{5,2,6,1,4\} \rightarrow |A| = 5$$

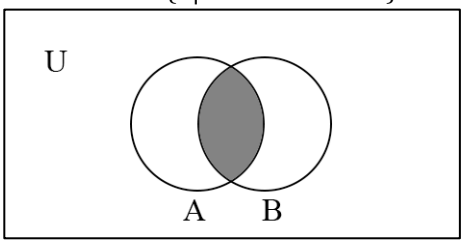
B. Operasi Himpunan

Terdapat beberapa operasi yang dapat dilakukan pada sebuah himpunan. Pada makalah ini, operasi himpunan yang akan digunakan adalah irisan dan gabungan.

1) Irisan

Operasi irisan dapat dinyatakan sebagai $A \cap B$ yang menghasilkan sebuah himpunan dengan elemen-elemenya merupakan anggota dari A serta anggota dari B. Operasi irisan dapat dinyatakan sebagai berikut:

$$A \cap B = \{x|x \in A \; dan \; x \in B\}$$

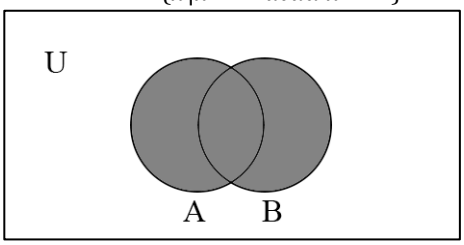


Gambar 4. Ilustrasi irisan

2) Gabungan

Operasi gabungan dapat dinyatakan sebagai $A \cup B$ yang menghasilkan sebuah himpunan dengan elemen-elemenya merupakan anggota dari A atau anggota dari B. Operasi gabungan dapat dinyatakan sebagai berikut:

$$A \cup B = \{x|x \in A \; atau \; x \in B\}$$



Gambar 5. Ilustrasi gabungan

Dari operasi irisan dan gabungan, berlaku prinsip inklusi-eksklusi, yaitu

$$|A \cup B| = |A| + |B| - |A \cap B|$$

C. String Simmilarity Metric

String similarity metric adalah metode-metode untuk mengkuantifikasi kesamaan dari buah string. Pada makalah ini, akan digunakan dua buah pengukuran kesemaan string yang mengimplementasikan teori himpunan, yaitu koefisien tumpang tindih dan index Jaccard.

1) Koefisien Tumpang Tindih

Koefisien tumpang-tindih (*overlap coefficient*) atau koefisien Szymkiewicz-Simpson merupakan salah satu metode untuk mengukur kesamaan dari dua buah string. Koefisien tumpang-tindih dihitung dengan cara membagi kardinalitas irisan dari dua buah himpunan dengan kardinalitas minimum antara dua buah himpunan tersebut.

$$Overlap\_Coefficient = \frac{|A \cap B|}{\min(|A|, |B|)}$$

2) Index Jaccard

Index Jaccard merupakan salah satu cara untuk mengukur kesamaan dua buah string. Index tersebut dihitung dengan membagi irisan dua buah set dengan gabungan dua buah set tersebut.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Dengan prinsip inklusi-eksklusi, persamaan tersebut dapat diubah mejadi sebagai berikut:

$$J(A,B) = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

C. Twitter Scraping

[7] Web scraping adalah proses pengumpulan data dari sebuah situs. Dengan itu, dapat didefinisikan bahwa Twitter Scraping adalah proses pengumpulan tweets yang diunggah oleh pengguna-pengguna media sosial Twitter. [6] Informasi dari hasil twitter scrapping dapat digunakan untuk mengukur sentiment masyarakat akan sesuatu. Beberapa informasi yang dapat diperoleh dari twitter scrapping mengetahui sentiment masyarakat atas suatu kebijakan pemerintah, reputasi dari suatu produk, atau menganalisa pelanggan melalui media sosial. Twitter dapat di *scrape* menggunakan modul *open-source* yang ada di python. Beberapa diantaranya adalah Snscrape, Twint, dan Tweepy.

D. Tweet Cleaning

Sebelum data tweet dianalisa, data tersebut harus dibersihkan terlebih dahulu. [8] Proses tweet cleaning dilakukan dengan tujuan untuk menyeragamkan struktur teks dari *tweet* yang diperoleh. Proses pembersihan data *tweet* umumnya dilakukan dengan langkah-langkah tersebut berikut:

1) Penghapusan tautan

Tautan yang ada pada twitter memiliki awalan “https”, “www”, atau “t.co”. Untuk menghapus tautan dapat digunakan regular expression.

2) Penghapusan mentions dan hashtag

Penghapusan mentions dan hashtag dapat dilakukan dengan menggunakan regex yang mencari semua kata yang diawali oleh “@” atau “#”. Proses ini harus dijalankan sebelum penghapusan karakter non-alfabet.

3) Penghapusan Karakter Non-Alfabet

Karakter-karakter yang dihapus mencakup symbol, angka, ASCII, dan tanda baca. Karakter-karakter tersebut dapat dihilangkan menggunakan regular expression

4) Tokenisasi

Tokenasi adalah proses perubahan string menjadi token kata atau frasa.

5) Case Folding

Merubah semua huruf kapital menjadi huruf kecil. Dapat dilakukan dengan fungsi bawaan Python .”lower()”.

6) Stemming

Stemming adalah proses yang merubah semua kata menjadi kata dasarnya. Stemming untuk Bahasa Indonesia dapat dilakukan dengan modul sastrawi yang dapat digunakan di Python.

7) Penghapusan stopword

Penghapusan kata-kata yang tidak makna diluar untuk penyusunan kata. Contohnya konjungsi seperti “dan” atau “serta”.

III. PEMBAHASAN

Keefektifan koefisien tumpang tindih dan index Jaccard dalam mendeteksi duplikat akan diuji pada dataset hasil scraping twitter dengan topik inggris. Twitter di *scrape* menggunakan python dengan modul snscrape dan disimpan dalam sebuah *dataframe* menggunakan *library* pandas. Jumlah tweet yang dikumpulkan sebanyak 3000 tweet dengan tweet yang dikumpulkan di post pada rentang tanggal 20 Agustus 2022 sampai tanggal 12 Desember 2022, dengan tanggal perolehan 12 Desember 2022.

Sebelum konten dari *tweet* diubah menjadi himpunan kata, isi dari konten harus dibersihkan terlebih dahulu. Pertama-tama, semua huruf kapital yang ada di dalam konten *tweet* harus diubah menjadi huruf kecil. Kemudian, hapus semua link, hashtag, dan mentions dengan bantuan *regular expression*. Setelah itu semua tanda baca pada konten *tweet* dihapus juga. Kemudian, hapus *whitespace* dari isi konten. Dengan ini, isi dari konten *tweet* hanya akan terdiri dari kata atau angka. Setelah pembersihan awal, langkah selanjutnya adalah *stemming*. *Stemming* dilakukan untuk merubah semua kata menjadi kata dasar. Untuk *stemming*, library yang digunakan adalah sastrawi. Terakhir, hapus semua baris yang isi dari *tweetnya* kosong setelah proses pembersihan tersebut.

```
def unified_cleaning(string):
    try:
        string = string.lower() # Lower casing
    except:
        pass
    try:
        string = re.sub("@[A-Za-z0-9_]+","", string) # Remove HashTag and Mentions
    except:
        pass
    try:
        string = re.sub("#[A-Za-z0-9_]+","", string)
    except:
        pass
    try:
        string = re.sub(r"http\S+", "", string) # Remove url Links
    except:
        pass
    try:
        string = re.sub(r"www\S+", "", string)
    except:
        pass
    try:
        string = re.sub(r"t.co\S+", "", string)
    except:
        pass
    punctuation = "!=\-_][]{};:'?/,.$%^&*()+"# Remove punctuation, need fix
    try:
        string = re.sub(punctuation, "",string)
    except:
        pass
    try:
        string = string.strip()
    except:
        pass
    try:
        string = string.strip() # Remove whitespace
    except:
        pass

    return string

df['content'] = df['content'].apply(unified_cleaning)

✓ 0.1s
```

Gambar 6. Proses Pembersihan Data

Setelah konten dari *tweet* dibersihkan, dibuat kolom baru dengan nama “set_content” pada dataframe yang berisi himpunan kata dari isi *tweet*. Kemudian untuk setiap baris pada *dataframe*, ukur kesamaan isi konten pada baris tersebut dengan baris lainnya menggunakan index Jaccard dan koefisien

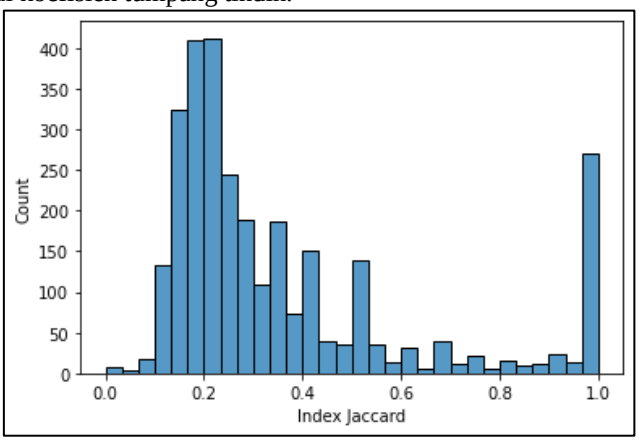
tumpang tindih. Hitung nilai maksimum dari index Jaccard dan koefisien tumpang tindih yang diperoleh dari baris tersebut dan simpan nilainya pada kolom baru beserta isi content yang mendekati string tersebut.

```
def transformStringtoSet(string):
    temp = string
    temp = temp.split()
    temp = set(temp)
    return temp
def getOverlapJaccard(set1, set2):
    intersect = len(set1&set2)
    a = len(set1)
    b = len(set2)
    return (intersect/min(a,b)),(intersect/(a+b-intersect))
```

```
for counti, i in enumerate(df_transformed["set_" + collumn]):
    final = 0
    final2 = 0
    temp1 = ""
    temp2 = ""
    for countj, j in enumerate(df_transformed["set_" + collumn]):
        if counti != countj:
            coef, jaccard = getOverlapJaccard(i, j)
            if final< coef:
                final = coef
                temp1 = df_transformed.iloc[countj]["content"]
            if final2 < jaccard:
                final2 = jaccard
                temp2 = df_transformed.iloc[countj]["content"]
    temp_array.append(final)
    jaccard_array.append(final2)
    coef_index.append(temp1)
    jaccard_index.append(temp2)
df_transformed["overlap_index"] = temp_array
df_transformed["jaccard_index"] = jaccard_array
df_transformed["coef_content"] = coef_index
df_transformed["jaccard_content"] = jaccard_index
```

Gambar 7. Implementasi Index Jaccard dan Koefisien Tumpang Tindih

Berikut adalah distribusi dari index Jaccard serta distribusi nilai koefisien tumpang tindih.

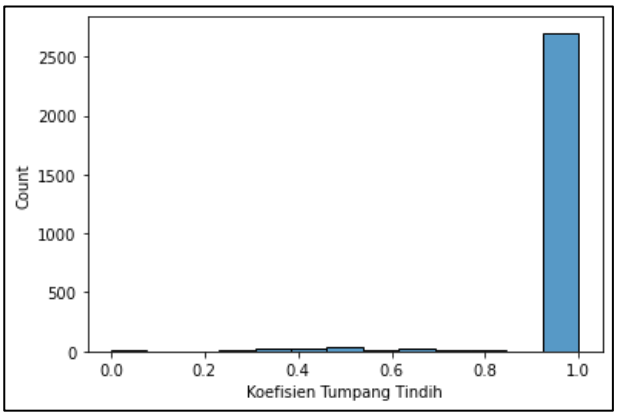


Gambar 8. Distribusi Index Jaccard Maksimum pada Setiap Tweet.

Berdasarkan histogram diatas, index Jaccard memiliki distribusi yang condong ke kiri. Dapat dilihat juga bahwa terdapat cukup banyak *tweet* yang memiliki index Jaccard bernilai 1. Hal tersebut terjadi karena dari data yang diperoleh, akun Twitter dari lembaga-lembaga yang menyebarkan berita sering kali mengepos berita yang sama baik dengan akun yang sama atau akun yang berbeda.

```
tauberitamedia    22
_BangFu           12
Linguosco         11
kompasiana        10
miakucink          9
                  ..
```

Gambar 9. Frekuensi User yang Terdeteksi Mengeposkan String yang Sama



Gambar 10. Distribusi Koefisien Tumpang Tindih pada Setiap Tweet

Di sisi lain, lebih dari 75% dari tweet memiliki koefisien tumpang tindih yang bernilai 1. Hal ini terjadi karena koefien tumpang tindih membagi kardinalitas irisan dengan kardinalitas terkecil antara dua set. Akibatnya, akan muncul masalah jika salah satu himpunan mempunyai kardinalitas 1-3. Hal ini diperkuat dengan tabel frekuensi berikut:

```
df_transformed.loc[df_transformed["overlap_index"]>0.8]['coef_content'].value_counts()
✓ 0.7s
inggris
2545
ini di inggris
64
a
27
inggris jg
23
inggris vs prancis
21
...
```

Gambar 11. Frekuensi String yang Sama berdasarkan Koefisien Tumpang Tindih

Dapat dilihat bahwa terdapat 2545 string yang mendekati string “inggris”. Pada kardinalitas tersebut, besar peluangnya untuk himpunan yang berkardinalitas kecil merupakan himpunan bagian dari himpunan lainnya yang mengakibatkan koefisien tumpang tindih bernilai 1. Karena masalah tersebut, koeffisien tumpang tindih kurang efektif untuk diterapkan dalam deteksi duplikat yang ada pada *twitter*.

Selanjutnya, dataset akan dipisah berdasarkan index Jaccard. Semua baris yang memiliki index Jaccard lebih dari atau sama dengan 0,8 akan disimpan sementara pada *dataframe* lain. Kemudian, hapus semua *tweet* yang memiliki index Jaccard

lebih dari atau sama dengan 0,8 dari *dataframe* awal. Dengan 0,8 sebagai sebagai batas, 344 baris memiliki index lebih dari 0,8 dan 2640 baris memiliki index Jaccard yang kurang dari 0,8.

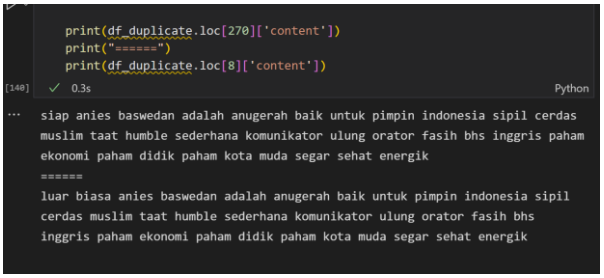
Dari dataset yang memiliki index Jaccard kurang dari 0,8, masih dapat ditemukan sejumlah *spam* yang menyalahgunakan tag.

Gambar 12. Wordcloud Isi Tweet dari Pencarian Topik Inggris

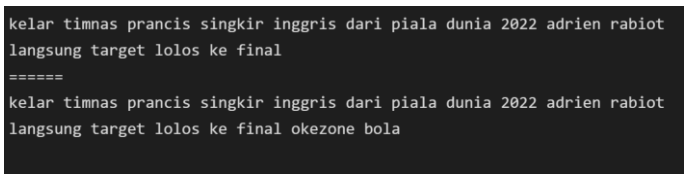
Dari wordcloud tersebut, dapat ditemukan bahwa masih terlihat beberapa tag populer yang tidak berkaitan dengan "inggris". Hal ini menandakan bahwa masih terdapat *spam* yang memanfaatkan tag-tag populer pada dataset. Beberapa dari tag tersebut adalah "kaesang" dan "daddies".

Jaccard index tidak dapat mendeteksi semua *spam* karena indeks Jaccard merupakan metrik yang digunakan untuk mengukur kesamaan string. Akibatnya jika terdapat kasus tag spam yang panjang isi aslinya 50% dari isi *tweet* dengan 50% lainnya digunakan untuk spam, maka jika tidak ada tweet lain yang memiliki isi pokok konten yang sama, indeks jaccard yang didapatkan akan bernilai kurang lebih 0,5. Selain itu, seiring berjalannya waktu, pastinya tag yang populer juga akan berubah sehingga tag yang ada di dalam konten spam akan menjadi variatif sehingga spam yang teriris tidak terdeteksi. Jumlah tag yang digunakan juga mempengaruhi pengukuran indeks Jaccard karena tag spam yang digunakan merupakan subset dari tag spam pada *tweet* lainnya, maka semakin banyak tag spam yang digunakan pada *tweet* lainnya, semakin kecil indeks Jaccardnya.

Meskipun demikian, indeks Jaccard masih dapat digunakan dalam mendeteksi duplikat semu. Indeks Jaccard dapat mendeteksi spam yang isi pokok *tweet*nya sama, tetapi terdapat sisipan kata atau perubahan beberapa kata dari *tweet* tersebut.

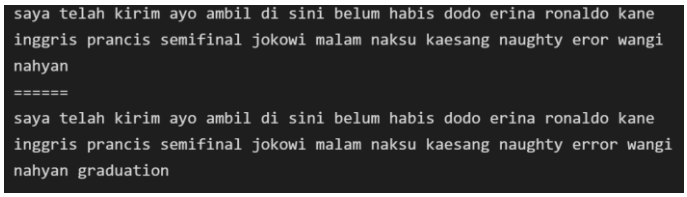


Gambar 13. Kasus Duplikat dengan Perubahan Kata



Gambar 14. Kasus Duplikat dengan Penyisipan Kata

Beberapa dari *spam* yang memiliki isi yang sama dapat terdeteksi dengan menghitung indeks Jaccard.



Gambar 15. Kasus Spam Tag

Oleh karena itu, indeks Jaccard masih dapat digunakan untuk mendeteksi *tweet-tweet* yang merupakan duplikat semu. Hanya saja metode tersebut tidak dapat digunakan untuk mendeteksi *tweet* yang menyalahgunakan tag. Karena spam-spam yang ada di *twitter* cukup unik untuk tidak dapat dideteksi dengan mengukur kesamaan string, maka dibutuhkan langkah tambahan yang memeriksa isi *tweet* atau cara lain untuk mendeteksi semua *spam* tersebut.

IV. KESIMPULAN

Deteksi duplikat semu dapat dilakukan dengan menerapkan teori himpunan dengan cara mengukur kesamaan string dari dua duplikat. Pengukuran kesamaan dua buah *tweet* dengan menggunakan koefisien tumpang tindih tidaklah efektif karena koefisien tumpah tindih sangat sensitif dengan apakah sebuah string merupkaan subset dari string lainnya. Di sisi lain, pengukuran kesamaan dua buah *tweet* dapat dilakukan dengan mengukur indeks Jaccard. Walupun indeks Jaccard tidak dapat mendeteksi *tweets spam*, indeks Jaccard masih dapat menentukan kasus duplikat semu dengan akurat.

V. APENDIKS

Implementasi dan dataset yang digunakan dapat dilihat pada pranala berikut:

https://github.com/Onyxcodeotto/Aplikasi-Teori-Himpunan-dalam-Mendeteksi-Duplikat-pada-Twitter.

Untuk dataset hasil *scraping* dan *cleaning* dapat dilihat di raw.xlsx, untuk dataset yang tidak terindikasi duplikat dapat dilihat pada final.xlsx, dan untuk data yang terindikasi duplikat dapat dilihat pada duplicate.xlsx.

VI. UCAPAN TERIMA KASIH

Puji dan syukur penulis panjatkan kepada Tuhan yang Maha Esa atas segala rahmat dan hidayahnya sehingga makalah yang berjudul "Aplikasi Teori Himpunan dalam Mendeteksi Duplikat pada Twitter" dapat diselesaikan dengan baik dan tepat waktu. Penulis mengucapkan terima kasih untuk seluruh. Penulis juga mengucapkan terima kasih kepada teman-teman yang baik secara langsung atau tidak langsung membantu penulis dalam

menyusun makalah ini. Tidak lupa, penulis mengucapkan terima kasih kepada orang tua dan keluarga atas doa dan dukungannya sehingga penulis dapat menghadapi segala hambatan yang ada dalam penyusunan makalah ini.

DAFTAR PUSTAKA

[1] https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2022-2023/Himpunan(2022)-1.pdf, diakses pada 10 Desember 2022

[2] https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2022-2023/Himpunan(2022)-2.pdf, diakses pada 12 Desember 2022

[3] https://itnext.io/string-similarity-the-basic-know-your-algorithms-guide-3de3d7346227, diakses pada 11 Desember 2022

[4] https://algotech.netlify.app/blog/kemiripan-teks/, diakses pada 11 Desember 2022

[5] https://web.archive.org/web/20070304092115/http://www.dcs.shef.ac.uk/~sam/stringmetrics.html#qgram, diakses pada 11 Desember 2022

[6] https://understandingdata.com/how-to-scrape-twitter-data/#:~:text=Scraping%20Twitter%20can%20yield%20many,read%20and%20write%20Twitter%20data., diakses pada tanggal 12 desember 2022.

[7] https://frdi.medium.com/scraping-data-twitter-tanpa-api-twitter-934443591122, diakses tangal 12 Desember 2022.

[8] https://iopscience.iop.org/article/10.1088/1742-6596/801/1/012072/pdf, diakses pada tanggal 12 Desember 2022.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 12 Desember 2022

Athif Nirwasito-13521053