

# Evaluasi Kmeans Clustering pada Preprocessing Sistem Temu Kembali Informasi

Yudha Pradana Putra<sup>1</sup>, Yoppy Yunhasnawa<sup>2</sup>, Faisal Rahutomo<sup>3</sup>

<sup>1,2,3</sup> Program Studi Teknik Informatika, Jurusan Teknologi Informasi, Politeknik Negeri Malang

<sup>1</sup>yudhapradana102@gmail.com, <sup>2</sup>yunhasnawa@gmail.com, <sup>3</sup>faisal@polinema.ac.id

**Abstrak**—Pada masa sekarang, berita masih menjadi salah satu konsumsi masyarakat pada dunia maya. Namun dengan seiring waktu jumlah berita yang diterbitkan semakin banyak. Hal ini dapat teratasi dengan dipergunakannya sistem temu kembali informasi yang dapat mencari berita dengan cepat. Sistem temu kembali informasi yang ada masih dikaji efisiensinya, jika berhubungan dengan jumlah berita yang sangat besar. Pada penelitian ini melakukan pengujian efisiensi dengan menambahkan proses *clustering* pada sistem temu kembali informasi. Selain itu juga dilakukan perbandingan hasil pengujian dengan metode *clustering* yang pernah digunakan pada penelitian sebelumnya. Pada *preprocessing* ini mengimplementasikan metode *kmeans clustering* dan pembobotan kata yang digunakan adalah *tfidf* dan *doc2vec*. Kemudian pencocokan *query* dengan dokumen disederhanakan menjadi pencocokan vektor *query* dengan vektor *centroid cluster*. Hasil pengujian efisiensi menunjukkan sistem temu kembali informasi yang menggunakan metode *kmeans clustering* dapat mencari berita lebih cepat. Sedangkan pengujian *precision*, *recall* dan *f-score* menunjukkan jika proses pencarian sistem temu kembali informasi menggunakan pembobotan kata *tfidf*, paling baik dilakukan pada *query* 4g lte pada *threshold* 0.05 pada jumlah klaster 750 dengan nilai *f-score* 0,818.

**Kata kunci**—sistem temu kembali informasi, *kmeans clustering*

## I. PENDAHULUAN

Metode *clustering* dapat diterapkan pada sistem temu kembali informasi untuk mengelompokkan informasi yang akan ditemukan sesuai dengan kebutuhan pengguna. Hal ini dikarenakan terkadang muncul permasalahan pada efisiensi sewaktu sistem memproses data yang sangat besar. Kurang efisiensi tersebut dikarenakan waktu tunggu sistem yang menjadi lebih lama karena diperlukan waktu untuk menghitung tingkat kemiripan *query* dengan dokumen. Sistem temu kembali adalah sebuah sistem yang digunakan untuk menemukan kembali (*retrieve*) informasi-informasi yang relevan terhadap kebutuhan pengguna dari suatu kumpulan informasi secara otomatis.

Berdasarkan pemaparan diatas, penelitian yang akan dilakukan adalah membangun sistem temu kembali yang secara otomatis dapat menemukan berita dengan relevan dan cepat berdasarkan *query* yang diinputkan dengan terlebih dahulu menerapkan metode *clustering* untuk mengelompokkan dokumen. Caranya dengan mengelompokkan dokumen-dokumen ke dalam cluster berdasarkan kedekatan antar

dokumen. Tujuan dari pengelompokan dokumen terlebih dahulu untuk mengurangi jumlah pencocokan *query* dengan dokumen [1]. Metode *clustering* yang digunakan adalah metode *k-means clustering* untuk proses *clustering* berita pada tahap *preprocessing*. Setelah itu dilakukan pencocokan *query* dengan dokumen yang disederhanakan menjadi pencocokan *query* dengan *centroid* klaster.

## II. RELEVANSI

Pada Penelitian kasus sebelumnya telah dilakukan penelitian oleh Tri Endah Sulistyoningrum pada tahun 2019 mengenai Implementasi *Single Pass Clustering* pada *Preprocessing* Temu Kembali Teks. Dalam penelitian tersebut, pada tahap *preprocessing* mengimplementasikan *single pass clustering* untuk mengelompokkan informasi yang ada terlebih dahulu serta melakukan pencocokan *query* dengan dokumen disederhanakan kepada pencocokan *query* dengan *centroid cluster*, mampu mencari berita dengan lebih cepat. Sedangkan pengujian efektifitas menggunakan nilai pengujian *precision*, *recall* dan *f-score*. Dari pengujian tersebut, didapatkan hasil jika proses pencarian paling tepat dilakukan pada *cluster* dengan nilai *threshold* 0,1. Hasil tersebut didapatkan ketika pengujian dilakukan menggunakan keyword ‘iphone’, ‘transaksi online’, ‘4g lte’, ‘aplikasi whatsapp’, dan ‘virtual reality’. Dari kelima pengujian tersebut hasil terbaik didapatkan pada pengujian dengan keyword ‘4g lte’. Yang mana pada pengujian tersebut nilai pengujian *f-score* sebesar 0,732. Nilai tersebut adalah nilai pengujian *f-score* yang paling baik dari pengujian lainnya. Sedangkan untuk nilai pengujian *precision* sebesar 0,756 dan pengujian *recall* sebesar 0,708 [1].

## III. LANDASAN TEORI

### A. Text Preprocessing

Sebelum melakukan analisa, *preprocessing* harus dilakukan terlebih dahulu untuk melakukan normalisasi kata-kata yang tidak diperlukan. Tahap ini bertujuan untuk mempersiapkan teks menjadi data yang akan mengalami pengolahan pada tahapan berikutnya. Pada proses *text preprocessing* meliputi, *case folding*, *tokenizing*, *filtering* dan *stemming*.

### B. Term Frequency – Inverse Document Frequency

*Term Frequency – Inverse Document Frequency* (TF-IDF) adalah metode untuk menghitung bobot pada setiap kata

yang paling umum digunakan pada *information retrieval*. Metode ini terkenal efisien, mudah dan akurat. Metode ini akan menghitung nilai *Term Frequency* (TF) dan *Inverse Document Frequency* (IDF) pada setiap token (kata) di setiap dokumen dalam korpus. Secara sederhana, metode TF-IDF digunakan untuk mengetahui berapa sering suatu kata muncul di dalam dokumen.

### C. Doc2Vec

Doc2Vec merupakan pengembangan dari implementasi metode *word embedding Word2Vec* yang bertujuan untuk mempresentasikan dokumen ke dalam bentuk *vector*. Dalam perkembangannya, Le dan Mikolov mengusulkan *Paragraph Vector*, teknik *unsupervised* yang mempelajari representasi *vector* kontinyu dan terdistribusi untuk potongan teks. Teks nya dimaksud disini panjangnya dapat bervariasi, mulai dari kalimat hingga dokumen. *Paragraph Vector* adalah untuk menekankan fakta bahwa metode ini dapat diterapkan pada potongan teks yang panjangnya bervariasi. Teknik ini dinilai cocok untuk merepresentasikan dokumen yang pada umumnya memiliki ukuran yang besar.

Doc2Vec dapat melakukan ekstraksi fitur dengan menggunakan semua informasi atau kata yang ada pada dokumen, karena setiap kata yang ada pada dokumen digunakan untuk proses learning. Doc2Vec menghasilkan *vector* dokumen dan *vector* kata yang ada pada data *training* [3].

### D. Kmeans Clustering

*K-Means* merupakan suatu algoritma pengklasteran yang cukup sederhana yang mempartisi *database* ke dalam beberapa *cluster k*. Algoritma cukup mudah untuk diimplementasikan dan dijalankan, relatif cepat, mudah disesuaikan dan banyak digunakan. Prinsip utama dari teknik ini adalah menyusun *K* buah partisi/pusat massa (*centroid*)/rata-rata (*mean*) dari sekumpulan data. Algoritma *K-Means* dimulai dengan pembentukan partisi klaster diawal kemudian secara iteratif partisi *cluster* ini diperbaiki hingga tidak terjadi perubahan yang signifikan pada partisi *cluster*.

Pengelompokan yang dapat digunakan seperti pengelompokan *non* hierarki yang membagi data kedalam bentuk dua atau lebih kelompok. *K-means* merupakan metode analisis kelompok yang mengarah pada pembagian *N* objek pengamatan kedalam *K* kelompok (*cluster*) dan setiap objek pengamatan dimiliki oleh suatu kelompok dengan rata-rata (*mean*) terdekat. Dasar algoritma *K-means* sebagai berikut :

- 1) Menentukan *k* sebagai jumlah *cluster* yang ingin dibentuk
- 2) Membangkitkan nilai *random* untuk pusat *cluster* awal (*centroid*)
- 3) Menghitung jarak setiap data input terhadap masing-masing *centroid* menggunakan rumus jarak *Euclidean* (*Euclidean Distance*) hingga ditemukan jarak yang paling dekat dari setiap data dengan *centroid*. Berikut persamaannya :

$$d(x_i, \mu_j) = \sqrt{\sum (x_i - \mu_j)^2}$$

Dimana :

$x_i \rightarrow$  data kriteria,

$\mu_j \rightarrow$  *centroid* pada *cluster* ke-*j*

- 4) Mengelompokkan setiap data berdasarkan kedekatannya setiap data berdasarkan kedekatannya dengan *centroid* (jarak terkecil)
- 5) Memperbaharui nilai *centroid*. Nilai *centroid* baru diperoleh dari rata-rata *cluster* yang bersangkutan dengan menggunakan rumus :

$$\mu_j(t+1) = \frac{1}{N_{sj}} \sum_{j \in S_j} x_j$$

Dimana :

$\mu_j(t+1) \rightarrow$  *centroid* baru pada iterasi ke (*t*+1)

$N_{sj} \rightarrow$  banyak data pada *cluster* *Sj*

### E. Kmeans++

Algoritma *k-means* sering digunakan dalam teknik *clustering* untuk meminimalkan jarak kuadrat yang telah dirata-ratakan antar titik dalam *cluster* yang sama. Tetapi algoritma *k-means* memiliki kelemahan yaitu tidak bisa memberikan akurasi yang baik dan kurang tepat. Menurut Jurnal *K-Means++ The Advantages of Carefull Seeding*, jika *k-means* ditambahkan dengan metode *randomized seeding technique* dapat meningkatkan nilai akurasi pada algoritma *k-means* [4].

Akurasi algoritma *k-means* bergantung pada nilai titik *centroid* awalnya. Jika menggunakan nilai *centroid* yang berbeda, maka akan menghasilkan yang berbeda juga bahkan membutuhkan banyak iterasi untuk menentukan *centroid* selanjutnya. Dengan menambahkan rumus *randomized seeding technique* maka akan lebih menentukan nilai *centroid* awal. Berikut rumus dari *randomized seeding technique* :

$$\frac{D(x^i)^2}{\sum_{x \in X} D(x^i)^2}$$

Keterangan :

$D(x^i)^2 =$  Jarak *Euclidean Distance*

$\sum_{x \in X} D(x^i)^2 =$  Jumlah Jarak *Euclidean Distance*

### F. Cosine Similarity

*Cosine Similarity* adalah ukuran kesamaan antara dua buah vektor dalam sebuah ruang dimensi yang didapat dari nilai *cosinus* sudut dari perkalian dua buah vektor yang dibandingkan karena *cosinus* dari 0° adalah 1 dan kurang dari 1 untuk nilai sudut yang lain, maka nilai *similarity* dari dua buah vektor dikatakan mirip ketika nilai dari *cosine similarity* adalah 1 [5]. Berikut adalah rumus *cosine similarity* :

$$\text{Similarity} = \cos(\phi) = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=1}^n A_i \times B_i}{\sqrt{\sum_{i=1}^n (A_i)^2} \times \sqrt{\sum_{i=1}^n (B_i)^2}}$$

Keterangan :

*A* = vektor

*B* = vektor

$A_i =$  bobot term *i* dalam blok  $A_i$

$B_i =$  bobot term *i* dalam blok  $B_i$

*i* = jumlah term dalam kalimat

*n* = jumlah vektor

### G. Pengukuran Performa

Pengujian dilakukan ketika sistem dijalankan untuk mengukur tingkat keberhasilan sistem dengan tujuan dibuatnya penelitian ini. Pengujian pada penelitian ini dilakukan menggunakan *precision*, *recall*, *f-score* dan waktu.

- 1) *Precision*

Precision digunakan untuk mengukur tingkat ketepatan antara informasi yang diminta oleh pengguna dengan jawaban yang diberikan oleh sistem.

$$\frac{N2 \cap N1}{N2}$$

## 2) Recall

Recall digunakan untuk mengukur tingkat keberhasilan sistem dalam menemukan kembali sebuah informasi.

$$\frac{N2 \cap N1}{N1}$$

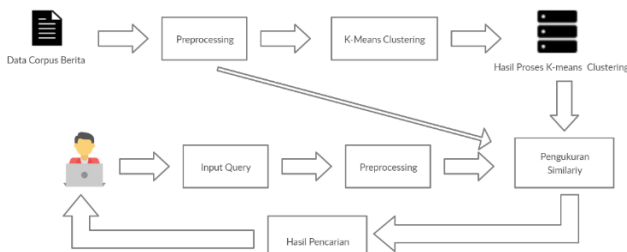
## 3) F-Score

F-score merupakan rata-rata harmonik dari precision dan recall.

$$\frac{2 \times (Precision \times Recall)}{(Precision + Recall)}$$

## IV. ARSITEKTUR SISTEM

Pada tahap ini, peneliti menentukan arsitektur sistem yang akan dibuat. Pada penelitian ini diterapkan tahap preprocessing pada corpus yang berisi berita untuk diklasterkan terlebih dahulu. Sebelum dilakukan proses clustering, berita diproses terlebih dahulu pada text processing. Hasil dari proses tersebut kemudian digunakan untuk mencari nilai TF-IDF atau Doc2vec. Hasil dari proses klaster tersebut digunakan untuk proses sistem temu kembali informasi berdasarkan query yang diinputkan pengguna yang telah dilakukan tahap preprocessing sebelumnya dan dilakukan proses pengukuran similarity menggunakan metode cosine similarity antara query dari pengguna dengan centroid masing-masing cluster.



Gambar 1.1 Arsitektur Sistem

## V. PENGUMPULAN DATA

### A. Data

Data yang diperoleh berasal dari penelitian sebelumnya yang dilakukan oleh Aad Miqdad Muadz Muzad dan Faisal Rahutomo, dalam tesis yang berjudul Korpus Berita Daring Bahasa Indonesia Dengan Depth First Focused Crawling. Pada penelitian ini, data yang digunakan adalah berita yang berasal dari corpus. Corpus yang digunakan adalah Indonesian news corpus. Corpus ini dibuat pada penelitian sebelumnya. Corpus yang digunakan memiliki format xml dan json. Corpus berita tersebut bisa didapatkan melalui situs internet Mendeley data. Data berita online ini diambil dari beberapa situs portal berita online seperti kompas.com, republika.co.id, viva.co.id dan tribunnews.com [2].

Data yang digunakan pada penelitian ini adalah data dari dengan kategori teknologi. Waktu berita diambil dalam rentang waktu bulan Juli hingga bulan Desember 2015.

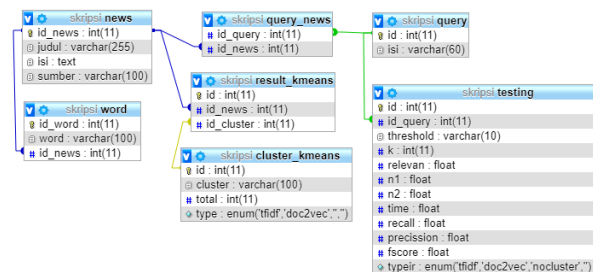
Selain itu, data yang akan diterapkan pada sistem ini bersifat dinamis, dimana pada penelitian ini menggunakan 800 dataset berita. Namun tidak menutup kemungkinan jumlah dataset tersebut bisa bertambah, berkurang atau dimodifikasi menggunakan fitur CRUD (Create, Read, Update, Delete) yang akan diterapkan pada sistem temu kembali informasi ini.

### B. Pengolahan Data

Pada penelitian ini, diterapkan sebuah preprocessing untuk melakukan pengelompokkan berita. Hasil dari proses clustering ini digunakan untuk proses temu kembali informasi berdasarkan query yang dimasukkan pengguna. Sebelum berita dikelompokkan, dilakukan tahap preprocessing terlebih dahulu menggunakan proses case folding, tokenization, filtering dan stemming. Hasil tersebut kemudian akan dilakukan perhitungan untuk mendapatkan nilai vector dokumen menggunakan metode TF-IDF dan Doc2Vec. Tujuan dari tahap ini adalah untuk mendapatkan nilai vektor dokumen untuk digunakan pada tahap proses clustering menggunakan k-means clustering.

## VI. IMPLEMENTASI

### A. Implementasi Basis Data



Gambar 1.2 Database

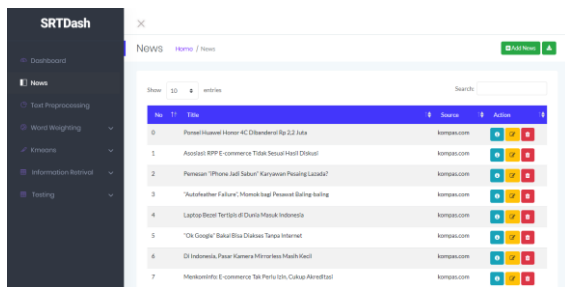
Implementasi database dari sistem temu kembali informasi yang menggunakan teknik clustering pada tahap preprocessing-nya tertera pada gambar. Pada database ini memiliki 7 tabel yang saling berelasi dan memiliki fungsi sendiri setiap tabel. Tabel news memiliki 4 field yaitu id\_news, judul, isi, dan sumber. Tabel ini berfungsi untuk menyimpan data berita yang menjadi data utama pada sistem temu kembali informasi dan dikelompokkan menggunakan teknik clustering. Tabel word memiliki 3 field yaitu id\_word, word, dan id\_news. Dimana id\_news pada tabel word memiliki relasi dengan id\_news yang ada pada tabel news. Tabel ini berfungsi menyimpan kata yang berasal dari berita yang telah dilakukan proses text preprocessing. Pada tabel cluster\_kmeans memiliki 4 field yaitu id, cluster, total, dan type. Tabel ini berfungsi menyimpan data cluster yang telah terbentuk dari proses clustering. Untuk tabel result\_kmeans adalah tabel yang berfungsi untuk menyimpan hasil pengelompokkan berita pada setiap klasternya. Dimana memiliki pada field-nya, memiliki 2 relasi tabel. Yang pertama id\_news pada result\_kmeans memiliki relasi dengan id\_news pada tabel news. Sedangkan yang kedua adalah id\_cluster memiliki relasi dengan id pada tabel cluster\_kmeans. Tabel query adalah tabel yang berfungsi menyimpan data query yang akan digunakan ujicoba pada sistem temu kembali informasi. Sedangkan tabel query\_news

adalah tabel yang menyimpan data *query* yang relevan dengan berita atas hasil dari pencocokan secara manual yang digunakan pada penelitian sebelumnya. Dan tabel *testing* berfungsi untuk menyimpan hasil pengujian sistem.

## B. Implementasi Sistem

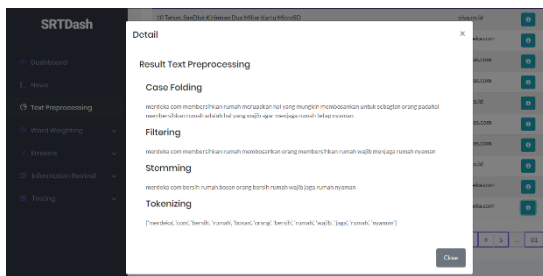
Sistem yang dibuat memiliki tampilan dan fungsi seperti gambar dibawah ini:

- 1) Tampilan berita : pada tampilan ini menunjukkan halaman yang menampilkan semua data berita yang tersimpan di database. Data yang ditampilkan adalah judul berita, sumber berita dan isi berita. Pada halaman ini admin dapat menambahkan, menghapus dan mengubah berita.



Gambar 1.3 Tampilan Berita

- 2) Tampilan hasil *preprocessing* berita : pada tampilan ini menunjukkan tampilan halaman hasil proses *text preprocessing* (*case folding*, *filtering*, *stemming*, *tokenizing*) setiap berita. Untuk melihat detailnya, dapat memilih tombol biru pada setiap berita.



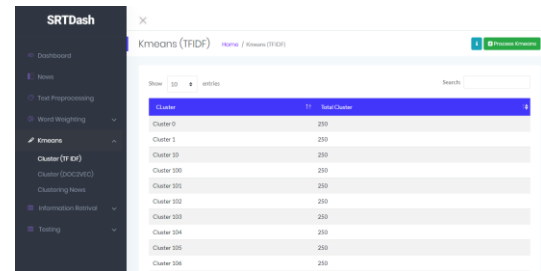
Gambar 1.4 Tampilan *Preprocessing* Berita

- 3) Tampilan pembobotan kata : pada tampilan ini menunjukkan tampilan halaman hasil proses pembobotan kata menggunakan metode Tf-Idf maupun Doc2vec. Untuk melihat detailnya, dapat memilih tombol biru pada setiap berita.



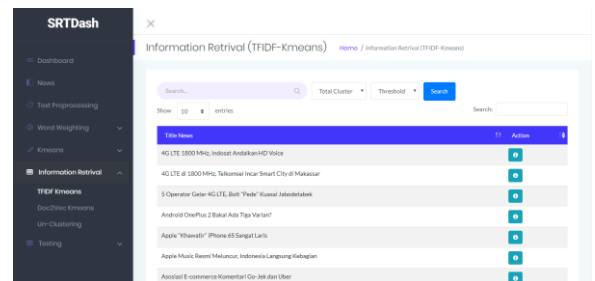
Gambar 1.5 Tampilan Pembobotan Kata

- 4) Tampilan hasil cluster : pada tampilan ini menunjukkan tampilan halaman hasil *cluster* yang terbentuk dari proses *clustering* yang menggunakan metode kmeans dan tfidf. Untuk menambahkan *clustering*, memilih tombol *Process Kmeans* berwarna hijau.



Gambar 1.6 Tampilan Hasil Cluster

- 5) Tampilan sistem temu kembali informasi : pada tampilan ini menunjukkan tampilan halaman sistem temu kembali informasi yang menggunakan metode *kmeans clustering* dan pembobotan kata tfidf, metode *kmeans clustering* dan pembobotan kata doc2vec dan tanpa *clustering*. Pada halaman ini *query* dimasukkan untuk dilakukan proses pencarian dengan *threshold* dan jumlah *cluster* terpilih dan hasilnya akan menampilkan berita yang berada pada *cluster* terpilih.



Gambar 1.7 Tampilan *Information Retrieval*

## VII. HASIL DAN PEMBAHASAN

Pada penelitian ini hasil pengujian akan dibandingkan dengan penelitian sebelumnya yang menggunakan metode *single pass clustering* (Faisal Rahutomo, Dwi Puspitasari, Tri Endah Sulistyoningrum, 2019). Hasilnya meliputi :

TABEL 1.1 HASIL *FScore* DENGAN *CLUSTERING*

| Query             | SPC-TFIDF | Kmeans-TFIDF | Kmeans-Doc2vec |
|-------------------|-----------|--------------|----------------|
| Android           | 0.304     | 0.572        | 0.258          |
| Iphone            | 0.715     | 0.807        | 0.155          |
| Game              | 0.264     | 0.678        | 0.088          |
| Facebook          | 0.4       | 0.74         | 0.146          |
| Transaksi Online  | 0.19      | 0.7          | 0.039          |
| 4g lte            | 0.732     | 0.818        | 0.176          |
| Aplikasi Whatsapp | 0.111     | 0.6          | 0.017          |
| Virtual Reality   | 0.222     | 0.636        | 0.032          |

Pada tabel 1.1 menunjukkan hasil perbandingan antara penelitian sebelumnya yang dilakukan oleh Tri Endah

Sulistiyoningrum pada tahun 2019, pada sistem temu kembali yang mengimplementasikan metode *single pass clustering* dan *tfidf* dengan sistem temu kembali yang mengimplementasikan metode *kmeans clustering* dan *tfidf* serta sistem temu kembali yang mengimplementasikan metode *kmeans clustering* dan *doc2vec*. Dari hasil tersebut diperoleh bahwa sistem temu kembali yang menerapkan metode *kmeans clustering* dan pembobotan kata *tfidf* memiliki nilai *fscore* yang lebih baik dari setiap *query* yang telah diujikan di berbagai variasi nilai *threshold*. Hasil yang berkebalikan dengan sistem temu kembali yang mengimplementasikan metode *kmeans clustering* dan *doc2vec* yang memiliki nilai *fscore* yang paling kecil.

TABEL 1.2 HASIL *FSCORE* TANPA *CLUSTERING*

| Query                        | Penelitian<br>Sebelumnya | Penelitian yang<br>dilakukan |
|------------------------------|--------------------------|------------------------------|
| <b>android</b>               | 0.369                    | 0.692308                     |
| <b>iphone</b>                | 0.735                    | 0.901639                     |
| <b>game</b>                  | 0.489                    | 0.678571                     |
| <b>facebook</b>              | 0.535                    | 0.764706                     |
| <b>transaksi<br/>online</b>  | 0.363                    | 0.625                        |
| <b>4g lte</b>                | 0.807                    | 0.83237                      |
| <b>aplikasi<br/>whatsapp</b> | 0.153                    | 0.6                          |
| <b>virtual<br/>reality</b>   | 0.222                    | 0.636364                     |

Pada tabel 1.2 menunjukkan hasil perbandingan antara penelitian sebelumnya yang dilakukan oleh Tri Endah Sulistiyoningrum pada tahun 2019, pada sistem temu kembali yang tidak mengimplementasikan metode *clustering* dengan sistem temu kembali yang sama namun menggunakan pengujian pada beberapa *threshold*. Dari hasil tersebut diperoleh bahwa penelitian sebelumnya memiliki nilai *fscore* yang lebih rendah daripada sistem temu kembali yang tidak mengimplementasikan metode *clustering* yang melakukan pengujian dengan beberapa *threshold*. Mayoritas hasil terbaik didapatkan pada pengujian *threshold* 0.05.

#### VIII. KESIMPULAN DAN SARAN

Pengujian dilakukan sebanyak empat kali terhadap semua sistem temu kembali informasi yang dibuat, mulai dari sistem temu kembali informasi yang mengimplementasikan *Kmeans Clustering* dan *Tfidf*, sistem temu kembali informasi yang mengimplementasikan *Kmeans Clustering* dan *doc2vec* dan sistem temu kembali informasi yang tidak mengimplementasikan metode *clustering*. Dalam hasil pengujian waktu, sistem temu kembali informasi yang menggunakan metode *clustering*, dan dengan jumlah *cluster* yang kecil, menghasilkan waktu proses yang lebih cepat. Sedangkan Hasil terbaik pada sistem temu kembali informasi yang mengimplementasikan metode *kmeans clustering* dan *tfidf* adalah pengujian pada *query* '4g lte' dengan nilai *threshold* 0.05 pada jumlah klaster 750 dengan nilai *recall* sebesar 0.886076, *precision* sebesar 0.76087 dan *fscore* sebesar 0.818713.

Selanjutnya hasil terbaik pada sistem temu kembali informasi yang mengimplementasikan metode *kmeans*

*clustering* dan *doc2vec* adalah pengujian pada *query* 'android' dengan nilai *threshold* 0.05 pada jumlah klaster 750 dengan nilai *recall* sebesar 1, *precision* sebesar 0.148564 dan *fscore* sebesar 0.258696.

Hasil perbandingan dengan penelitian sebelumnya, sistem temu kembali yang mengimplementasikan metode *kmeans clustering* dan *tfidf* menghasilkan nilai *fscore* yang lebih baik dari sistem temu kembali yang mengimplementasikan metode *single pass clustering*. Berbanding terbalik dengan sistem temu kembali yang mengimplementasikan metode *kmeans clustering* dan *doc2vec* yang memiliki nilai paling kecil. Hal ini dikarenakan nilai *similarity* setiap *cluster* memiliki nilai yang saling berdekatan dan data training yang masih kurang.

Untuk penelitian selanjutnya bisa dilakukan penambahan *dataset* agar dapat memaksimalkan data *training* untuk metode *doc2vec*. Selain itu juga bisa menambahkan fitur pada metode *doc2vec* agar menghasilkan nilai *similarity cluster* tidak berdekatan.

#### IX. DAFTAR PUSTAKA

- [1] Rahutomo, Faisal., Puspitasari, D., & Sulistiyoningrum, E, T. (2019). Implementasi Single Pass Clustering pada Preprocessing Temu Kembali Koleksi Teks. *Jurnal Edukasi dan Penelitian Informatika*, Vol. 6, No.1.
- [2] Muzad, Aad Miqdad Muadz., & Rahutomo, F. (2016). Korpus Berita Daring Bahasa Indonesia Dengan Depth First Focused Crawling. *Pros. Sentrinov (Seminar Nas. Terap. Ris. Inov.*, Vol. 2, No. 1, pp. 11–20.
- [3] Widyaningtyas, W, C., Adiwijaya & Faraby, S, A. (2018). Klasifikasi Sentiment Analysis pada Review Film Berbahasa Inggris dengan Menggunakan Metode Doc2Vec dan Support Vector Machine (SVM). *e-Proceeding of Engineering*, Vol. 5, No.1.
- [4] Fikri. C. M., Agustin. F. E., & Mintarsih, F. (2017). Pengelompokan Kualitas Kerja Pegawai Menggunakan Algoritma Kmeans++ dan COP-Kmeans Untuk Merencanakan Program Pemeliharaan Kesehatan Pegawai di PT. PLN P2B JB Depok. *Jurnal Pseudocode*, Vol. IV, No. 1.
- [5] Wahyuni, R. T., Prastiyanto, D., & Suprptono, E. (2017). Penerapan Algoritma Cosine Similarity dan Pembobotan TF-IDF pada sistem Klasifikasi Dokumen Skripsi. *Jurnal Teknik Elektro*, Vol. 9, No. 1.