Klasifikasi Teks Multi Label pada Hadis dalam Terjemahan Bahasa Indonesia Berdasarkan Anjuran, Larangan dan Informasi menggunakan TF-IDF dan KNN

Ilham Kurnia Syuriadi[1], Adiwijaya[2], Widi Astuti[3]

[1,2,3]Fakultas Informatika, Universitas Telkom, Bandung
[1]ilhamkurnias@students.telkomuniversity.ac.id, [2]adiwijaya@telkomuniversity.ac.id,
[3]widiwdu@telkomuniversity.ac.id

Abstrak

Hadis adalah segala sesuatu yang dinisbatkan kepada Nabi Muhammad SAW baik berupa perkataan, berbuatan atau sikap. Hadis merupakan pedoman hidup kedua bagi umat muslim setelah AL Qur'an. Sebagai pedoman hidup, umat muslim sangat diharuskan mempelajari Hadis. Melakukan kategorisasi pada Hadis adalah salah satu cara untuk mempermudah dalam mempelajari Hadis. Penelitian ini bertujuan untuk melakukan klasifikasi terhadap Hadis. Hadis yang digunakan adalah Hadis shahih Imam Bukhari. Klasifikasi yang dilakukan adalah klasifikasi multi label. Kelas yang digunakan adalah kelas anjuran, larangan dan informasi. Ekstraksi fitur yang digunakan adalah *N-Gram* dengan nilai n = 1 *(unigram)* dan n = 2 *(bigram)*. Sedangkan fitur seleksi yang digunakan adalah *TF-IDF*. Klasifikasi digunakan menggunakan metode *k-NN*. Skenario percobaan dilakukan dengan mencoba beberapa nilai *k* pada *k-NN*, penambahan *threshold* pada *df (document frequency)* untuk perhitungan *TF-IDF* dan melakukan beberapa perubahan pada tahap *preprocessing*. Untuk mendapatkan hasil evaluasi yang dapat dipercaya, digunakan *cross validation*. Sedangkan untuk evaluasi dari sistem yang telah dibangun, dihitung menggunakan nilai *F1-score*. Berdasarkan beberapa percobaan yang dilakukan didapatkan nilai *F1-score* terbaik sebesar 0.853. Hasil tersebut menunjukan metode *k-NN* lebih baik dibanding metode *ANN* dan metode *baseline* pada klasifikasi hadis multi label.

Kata kunci : Klasifikasi multi label, Hadis, *k-NN*, *TF-IDF*, *N-gram*

Abstract

Hadith is everything that is attributed to the Prophet Muhammad either in the form of words, works or attitudes. Hadith is the second life guide for Muslims after the Qur'an. As a way of life, Muslims are strongly required to visit the Hadith. Categorizing the Hadith is one of many way to ease people learn Hadith. This study aims to make a classification of the Hadith. The hadith used is the Imam Bukhari Sahih Hadith. The classification carried out is a multi label classification. The class used is a class of recommendations, prohibitions and information. Feature extraction used is N-Gram with a value of n = 1 (unigram) and n = 2 (bigram). While the feature selection used is TF-IDF. For classification method used is the k-NN method. The trial scenario is done by trying several k values on k-NN, the threshold on df (document frequency) for calculating TF-IDF and do some changed at the preprocessing stage. To get a reliable evaluation result, cross validation used. Evaluation of the system that has been built, it is calculated using the F1-score. Based on some of experiments conducted, the best F1-score was 0.853. These results indicate that the k-NN method is better than the ANN method and the baseline method in the multi label hadith classification.

Keywords: Multi label classification, Hadith, kNN, TF-IDF, N-gram

1. Pendahuluan

Hadis adalah segala sesuatu yang dinisbatkan kepada Nabi Muhammad SAW baik berupa perkataan, perbuatan, sikap dan lain sebagainya. Hadis merupakan sumber hukum tersendiri bagi umat muslim yang tidak dijelaskan dalam Al Qur'an[1]. Setiap Hadis terdiri dari 2 bagian yaitu Sanad dan Matan. Sanad adalah untaian nama para penyampai Hadis yang menjamin keaslian dari Hadis itu. Matan adalah isi dari tersebut. Setiap hadis diawali dengan Sanad[2]. Umumnya hadis di koleksi oleh imam besar, salah satunya koleksi Hadis yang disusun oleh Imam Bukhari yang memiliki nama lengkap Abu Abdullah Muhammad bin Ismail bin Ibrahim bin alMughirah al-Ju'fi. Imam Bukhari hidup antara 194 hingga 256 hijriah. Sebagai seorang muslim, sangat dianjurkan untuk mempelajari Hadis. Oleh karena itu, dibutuhkan kategorisasi pada Hadis untuk mempermudah seorang muslim dalam mempelajari Hadis.

Pada dasarnya, melakukan kategorisasi pada Hadis tidaklah berbeda dengan klasifikasi teks. Klasifikasi teks menggunakan *TF-IDF* dan *k-NN* sudah pernah dilakukan oleh Bruno Trstenjak, Sasa Mikac dan Dzenana Donko[3]. Pada penelitian tersebut mereka melakukan klasifikasi terhadap teks berita. Sedangkan untuk klasifikasi Hadis sendiri juga sudah pernah beberapa kali pada penelitian [4] dan [5]. Pada penelitian [4],

dilakukan klasifikasi Hadis menggunakan beberapa metode diantaranya yaitu *ANN* dan *baseline*. Ekstraksi fitur menggunakan *N-Gram* dilakukan pada penelitian tersebut. Selain itu, Muhammad Romi Ario Utomo dan Yuliant Sibaroni juga menggunakan ekstraksi fitur yang serupa pada penelitian[6]. Penelitian tersebut memaksimalkan hasil evaluasi yang didapat dengan cara menambahkan nilai *threshold* pada nilai *df* untuk penghitungan bobot *TF-IDF*[6]. Klasifikasi yang akan dilakukan pada penelitian ini adalah klasifikasi multi-label. Oleh karena itu dibutuhkan *problem transformation* agar klasifikasi bisa tetap dilakukan. *Problem transformation* yang akan digunakan adalah *binary relevance*[7]. Hal tersebut serupa dengan penelitian pada[8] dan [9].

Berdasarkan latar belakang yang sudah disebutkan, dapat dirumuskan masalah dari penelitian ini adalah (1) bagaimana melakukan klasifikasi multi label terhadap Hadis (2) bagaimana mengetahui pengolahan data terbaik hingga data bisa digunakan untuk klasifikasi, (3) bagaimana mendapatkan evaluasi terbaik dari klasifikasi yang dilakukan.

Berdasarkan masalah yang sudah disebutkan, dapat dirumuskan tujuan dari penelitian ini adalah (1) membangun klasifikasi teks multi label pada Hadis shahih Bukhari menggunakan metode *k-NN* , (2) melakukan ekstraksi fitur menggunakan *n-gram* dan menganalisa nilai *n* terbaik pada ekstraksi fitur tersebut, (3) melakukan analisa terhadap nilai *k* pada *k-NN*.

2. Studi Terkait

Klasifikasi Hadis multi-label berdasarkan anjuran, larangan dan informasi sudah pernah dilakukan pada penelitian [4]. Pada penelitian tersebut dilakukan percobaan klasifikasi Hadis menggunakan metode *ANN* dan *baseline*. Ekstraksi fitur dan seleksi fitur yang digunakan pada metode *ANN* adalah *N-Gram* dan *TF-IDF*. Pada percobaan menggunakan metode *ANN* dilakukan dengan beberapa percobaan yaitu (1) mencoba nilai *n* pada *N-Gram* dengan nilai 1 *(unigram)*, 2 *(bigram)* dan 3 *(trigram)*, (2) mencoba jumlah *hidden layer* yaitu 10 dan 20 *hidden layer*. Pada percobaan (1) didapatkan nilai *F1-Score* untuk masing-masing *unigram*, *bigram*, dan *trigram* adalah sebesar 0.79, 0.70 dan 0.48. Pada percobaan tersebut terlihat penurunan nilai *F1-Score* apabila *n* pada *N-Gram* semakin besar. Sehingga pada percobaan selanjutnya mereka hanya menggunakan ekstraksi fitur *N-Gram* dengan nilai *n* = 1 *(unigram)*. Pada percobaan (2) didapatkan nilai *F1-Score* masing-masing pada 10 dan 20 *hidden layer* sebesar 0.79 dan 0.85. Pada percobaan tersebut terlihat peningkatan nilai *F1-Score*. Pada percobaan dengan metode *baseline*, dilakukan dengan cara *string matching* yaitu mencocokan kata-kata pada satu Hadis dengan *list*. Pada percobaan tersebut digunakan dua *list* yaitu *list* kata-kata anjuran dan *list* kata-kata larangan. Misal pada satu Hadis mengandung satu atau lebih kata-kata pada *list* larangan maka Hadis tersebut akan diklasifikasi sebagai larangan demikian juga untuk anjuran. Apabila Hadis tersebut diak mengandung kata-kata pada *list* maka Hadis tersebut akan diklasifikasi sebagai informasi. Namun metode ini mendapatkan hasil evaluasi *F1-Score* yang cukup rendah dibanding percobaan sebelumnya, yaitu sebesar 0.69. Pada penelitian [4], terdapat kelemahan pada metode yang digunakan. Pada metode *ANN* menggunakan nilai probabilitas densitas. Pada metode *baseline* terdapat kelemahan karena metode ini hanya memperhitungkan kata-kata yang berada pada *list*. Sedangkan pada hasil penelitian [6] diketahui bahwa setiap kata-kata pada data dapat mempengaruhi hasil klasifikasi.

Penelitian[6] melakukan klasifikasi terhadap bahasa inggris. Dimana data yang digunakan memiliki dua kelas yaitu *english america* dan *english british*. Hasil evaluasi penelitian [6] diukur menggunakan nilai akurasi. Pada penelitian tersebut juga dilakukan percobaan pada nilai *n* pada ekstraksi fitur *N-Gram*. Serupa dengan penelitian[4], pada penelitian[6] juga terjadi penurunan hasil evaluasi apabila nilai *n* semakin besar. Nila akurasi untuk masing-masing *n* pada *N-Gram* yaitu 1 *(unigram)*, 2 *(bigram)* dan 3 *(trigram)* adalah sebesar 87.1%, 84.9% dan 55.5%. Selain itu, pada penelitian tersebut juga dilakukan percobaan penambahan nilai *threshold* terhadap nilai *df* pada penghitungan *TF-IDF*. Percobaan tersebut dicobakan pada lima nilai *threshold* yaitu 1, 2, 3, 5, dan 10, nilai akurasi yang didapatkan untuk masing-masing nilai *threshold* tersebut adalah 93.5%, 94%, 91.1%, 75% dan 73.5%. Tidak hanya itu, percobaan pada penelitian [6] juga meliputi tahap pada *preprocessing* yaitu *stemming*. Percobaan dilakukan dengan penggunaan *stemming* dan tanpa penggunaan *stemming*, masing-masing percobaan mendapatkan nilai akurasi sebesar 91.2% dan 92.1%. Pada hasil penelitian [6], dapat diketahui bahwa ekstraksi fitur dan seleksi fitur yang digunakan dapat mempengaruhi hasil klasifikasi.

Pada penelitian[3] dilakukan klasifikas terhadap teks. Data yang digunakan adalah teks berita. Pada penelitian ini, digunakan kombinasi *TF-IDF* dan *k-NN* untuk klasifikasi berita. Pada penelitian ini fitur yang digunakan adalah kata-kata unik dari seluruh dokumen kemudian dilakukan pembobotan menggunakan *TF-IDF*. Penelitian [3] menggunakan akurasi sebagai evaluasi hasil dari sistem yang dibangun. Berdasarkan percobaan yang dilakukan pada penelitian tersebut, didapatkan akurasi terbaik sebesar 92%. Pada hasil penelitian [3] dapat diketahui bahwa penggunaan *TF-IDF* dan *k-NN* mendapatkan hasil evaluasi yang bagus pada klasifikasi teks.

Klasifikasi multi-label adalah klasifikasi yang berbeda dengan klasifikasi single-label. Oleh karena itu dibutuhkan penangan khusus seperti *problem transformation*[7]. Pada *problem tansformation* terdapat beberapa cara, salah satunya adalah *binary relevance*[7]. Hal serupa juga dilakukan pada penelitian [8] dan [9]. Pada *binary relevance* label pada data akan direpresentasikan kedalam matriks. Dimana jumlah kolom

merepresentasikan kelas dan baris merepresentasikan nilai dari kelas tersebut. Pada penelitian yang dilakukan terdapat tiga kelas yang digunakan sehingga kolom dari matriks ada tiga juga. Nilai dari baris pada matriks berupa angka 0 dan 1. Nilai 0 berarti data tersebut tidak termasuk dalam suatu kelas, begitu sebaliknya pada nilai 1 yang berartu data tersebut termasuk dalam suatu kelas. Berdasarkan studi [7] dapat diketahui bahwa penggunaan *probelm transformation* yaitu *binary relevance* bisa digunakan dan sudah digunakan pada beberapa penelitian [8][9].

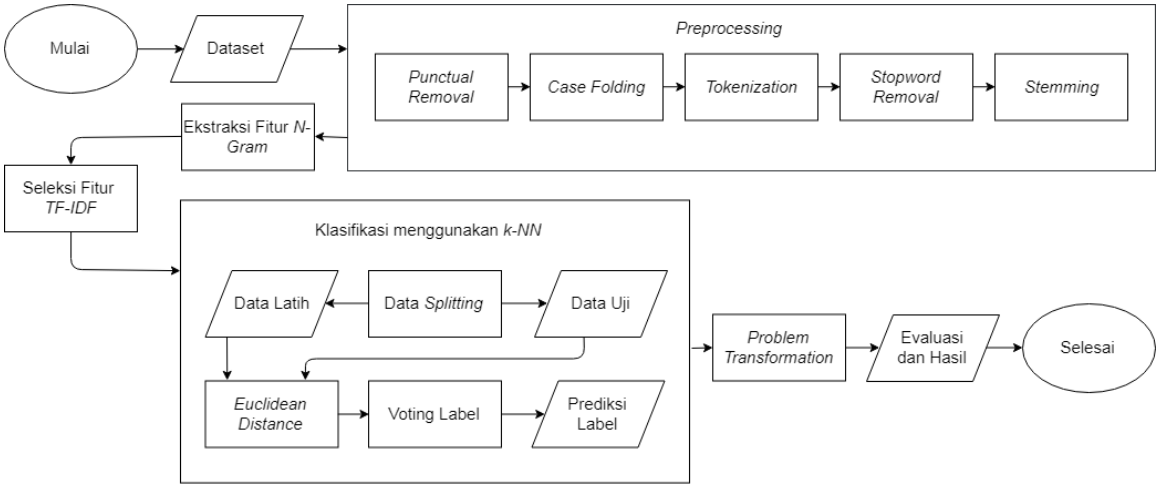
3. Sistem yang Dibangun

Pada penelitian ini data yang digunakan adalah data Hadis shahih Bukhari dalam bentuk file excel yang telah diberi label dengan cara *hand labeling* oleh Muhammad Yuslan Abu Bakar[8]. Kelas yang digunakan adalah anjuran, larangan dan informasi. Pada data yang digunakan sudah diterapkan *problem transformation* yaitu *binary relevance* seperti pada penelitian[9] dan yang disebutkan pada[7]. Berikut bentuk data yang digunakan.

Tabel 1 Contoh dataset

Data	Anjuran	Larangan	Informasi
Iman memiliki lebih dari enam puluh cabang, dan malu adalah bagian dari iman.	0	0	1
Siapa yang Kaum Muslimin selamat dari lisan dan tangannya.	0	0	1
Aku datang untuk menjelaskan Lailatul Qodar kepada kalian, namun fulan dan fulan saling berdebat sehingga akhirnya diangkat (lailatul qodar), dan semoga menjadi lebih baik buat kalian, maka itu intailah (lailatul qodar) itu pada hari yang ketujuh, enam dan lima .	1	0	1

Pada penelitian ini dibangun sistem untuk melakukan klasifikasi multi label terhadap Hadis shahih Bukhari. Setiap sistem yang dibangun pasti harus memiliki rancangan, begitu juga dengan sistem yang dibangun pada penelitian ini. Gambar 1 berikut merupakan gambaran sistem yang dibangun untuk klasifikasi multi label Hadis.



Gambar 1 Gambaran umum sistem

3.1 *Preprocessing*

Preprocessing merupakan tahapan yang umum dalam melakukan klasifikasi terhadap teks[10]. Tahap ini bertujuan untuk mengolah data yang tadinya hanya berupa teks menjadi data yang siap untuk diklasifikasi. Adapun *preprocessing* yang digunakan pada penelitian ini adalah *punctual removal, case folding, tokenization, stopword removal* dan *stemming*.

Punctual removal adalah proses untuk menghilangkan tanda baca. Pada proses ini setiap karakter selain angka dan huruf akan dihapus. Misalnya pada kalimat “Tanda iman adalah mencintai (kaum) Anshar dan tanda nifaq adalah membenci (kaum) Anshar.” menjadi “Tanda iman adalah mencintai kaum Anshar dan tanda nifaq adalah membenci kaum Anshar”.

Case folding adalah proses untuk mengubah setiap huruf besar menjadi huruf kecil. Misalnya pada kalimat "Tanda iman adalah mencintai kaum Anshar dan tanda nifaq adalah membenci kaum Anshar" menjadi "tanda iman adalah mencintai kaum anshar dan tanda nifaq adalah membenci kaum anshar".

Tokenization adalah proses untuk mengubah kalimat menjadi *token* atau kata. Misalnya pada kalimat "tanda iman adalah mencintai kaum anshar dan tanda nifaq adalah membenci kaum anshar" menjadi ['tanda', 'iman', 'adalah', 'mencintai', 'kaum', 'anshar', 'dan', 'tanda', 'nifaq', 'adalah', 'membenci', 'kaum', 'anshar'].

Stopword removal adalah proses untuk menghapus kata yang dianggap tidak memiliki pengaruh pada suatu kalimat. Misalnya pada ['tanda', 'iman', 'adalah', 'mencintai', 'kaum', 'anshar', 'dan', 'tanda', 'nifaq', 'adalah', 'membenci', 'kaum', 'anshar'] menjadi ['tanda', 'iman', 'mencintai', 'kaum', 'anshar', 'tanda', 'nifaq', 'membenci', 'kaum', 'anshar'].

Stemming adalah proses untuk mengubah kata dengan imbuhan menjadi kata dasar. Misalnya pada ['tanda', 'iman', 'mencintai', 'kaum', 'anshar', 'tanda', 'nifaq', 'membenci', 'kaum', 'anshar'] menjadi ['tanda', 'iman', 'cinta', 'kaum', 'anshar', 'tanda', 'nifaq', 'benci', 'kaum', 'anshar'].

3.2 Ekstraksi Fitur *N-Gram*

Fitur dapat diartikan sebagai objek yang keberadaannya memiliki karakteristik signifikan dalam proses klasifikasi. Pada klasifikasi teks, fitur dapat berupa kata-kata yang ada pada suatu teks. Ekstraksi fitur adalah proses menghasilkan fitur yang akan digunakan pada proses klasifikasi. Pada penelitian ini ekstraksi fitur yang digunakan adalah *N-Gram* dengan nilai n = 1 *(unigram)* dan nilai n = 2 *(bigram)*.

N-Gram memisahkan kata berdasarkan urutan kata tersebut pada kalimat. Misal pada *unigram*, kalimat "tanda iman cinta kaum anshar tanda nifaq benci kaum anshar" menjadi ['tanda', 'iman', 'cinta', 'kaum', 'anshar', 'tanda', 'nifaq', 'benci', 'kaum', 'anshar']. Sedangkan pada *bigram* menjadi ['tanda iman', 'iman cinta', 'cinta kaum', 'kaum anshar', 'anshar tanda', 'tanda nifaq', 'nifaq benci', 'benci kaum', 'kaum anshar'].

3.3 Seleksi Fitur *TF-IDF*

TF-IDF adalah pemberian bobot pada fitur. Pada *TF-IDF* terdapat *tf (term frequency), idf (inverse document frequency* dan *df (documen frequency)*. *Tf* merupakan jumlah kemunculan suatu fitur *(term)* dalam satu dokumen. *Df* merupakan jumlah kemunculan dari suatu *term* dari seluruh dokumen. Sedangkan untuk *idf* muncul dari *df* dan digunakan untuk mengurangi bobot suatu term apabila term tersebut muncul dari hampir sebagian besar dokumen yang digunakan. Adapun penghitungan bobot menggunakan *TF-IDF* dilakukan seperti persamaan berikut[11].

$$tf = f_{t,d} \tag{1}$$

Nilai $f_{t,d}$ pada persamaan (1) menyatakan frekuensi kemunculuan suatu *term* pada satu dokumen.

$$idf_j = \log(\frac{N}{df_j}) \tag{2}$$

Pada persamaan (2), *N* menyatakan jumlah dokumen sedangkan df_j menyatakan jumlah dokumen yang mengandung term *j*. Setelah nilai *tf* dan *idf* didapatkan, dihitung niliat bobot suatu *term* yang dinyatakan dengan *w* menggunakan persamaan (3) berikut.

$$w = tf \times idf_j \tag{3}$$

Seleksi fitur dilakukan untuk mengurangi fitur yang dianggap tidak memiliki pengaruh signifikan terhadap proses klasifikasi. Proses ini dilakukan dengan cara memberi nilai batas ukur *(threshold)* pada nilai *documen frequency (df)* dari suatu fitur. Suatu fitur dengan nilai *df* di bawah atau sama dengan nilai *threshold* yang diujikan akan dihapus. Asumsi yang diterapkan adalah dengan semakin sedikitnya kemunculan suatu kata maka tidak akan mempengaruhi performansi secara global [12]. Hal tersebut karena apabila suatu kata muncul semakin sedikit maka kata tersebut hanya menggambarkan ciri-ciri dari data tertentu sehingga kata tersebut tidak akan menggambarkan ciri dari suatu kelas atau dengan kata lain *overfit*.

3.4 Klasifikasi Menggunakan *k-NN*

Sebelum melakukan proses klasifikasi, data terlebih dahulu dibagi menjadi data latih dan data uji. Pembagian data dilakukan dengan perbandingan 80% untuk data latih dan 20% untuk data uji. Pembagian dilakukan empat kali dengan keterangan pada iterasi pertama yang menjadi data uji adalah 20% pertama dari

data set sisanya menjadi data latih, pada iterasi kedua data uji adalah 20% kedua dari data set sisanya menjadi data latih dan begitu seterusnya. Hal ini dilakukan untuk melakukan *cross validation* pada hasil evaluasi. Hasil evaluasi pada satu rangkaian percobaan didapat dari rata-rata hasil evaluasi keempat iterasi pembagian data. Pembagian data digambarkan pada gambar berikut.

Data ke 1-266	Data ke 267-532	Data ke 533-798	Data ke 799-1064
Testing	Training	Training	Training

Data ke 1-266	Data ke 267-532	Data ke 533-798	Data ke 799-1064
Training	Testing	Training	Training

Data ke 1-266	Data ke 267-532	Data ke 533-798	Data ke 799-1064
Training	Training	Testing	Training

Data ke 1-266	Data ke 267-532	Data ke 533-798	Data ke 799-1064
Training	Training	Training	Testing

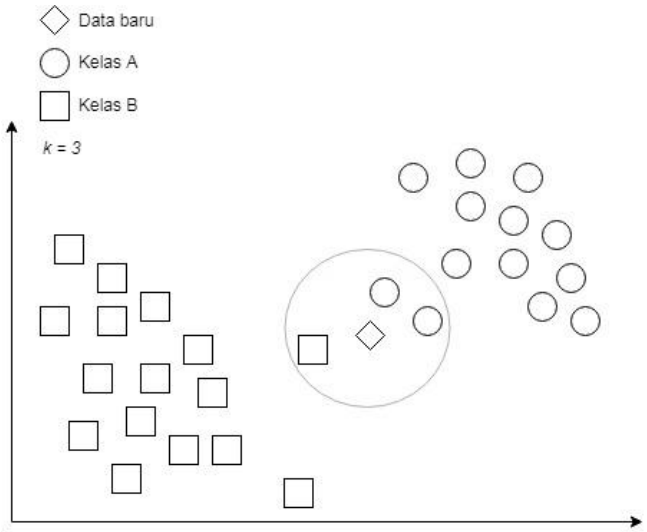
Gambar 2 Ilustrasi pembagian data

Pada penelitian ini, digunakan metode *k-NN* untuk melakukan klasifikasi seperti penelitian[3]. Pada metode *k-NN* klasifikasi dilakukan dengan cara menghitung *euclidean distance* dari satu data uji yang akan diklasifikasi terhadap seluruh data latih. Penghitungan *euclidean distance* dilakukan seperti pada persamaan(4).

$$d(a,b) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + \cdots + (a_n - b_n)^2} \quad (4)$$

Pada persamaan(4), a dan b adalah data Hadis yang digunakan. Pada persamaan tersebut a dan b adalah data uji dan data latih. Sedangkan a_1 sampai a_n merupakan fitur dari data uji begitu juga dengan b_1 sampai b_n yang juga merupakan fitur dari data latih.

Setelah itu dihitung data sebanyak nilai *k* terdekat dengan data uji yang sedang diklasifikasi. Setelah data sebanyak nilai *k* terdekat didapatkan, dilakukan penghitungan terhadap kelas pada sebanyak *k* data yang didapatkan dan kelas terbanyak menjadi hasil klasifikasi dari data tersebut. Ilustrasi klasifikasi menggunakan *k-NN* dengan nilai *k* = 3 ditunjukan pada gambar 3 berikut[13].



Gambar 3 Gambaran klasifikasi *k-Nearest Neighbour*

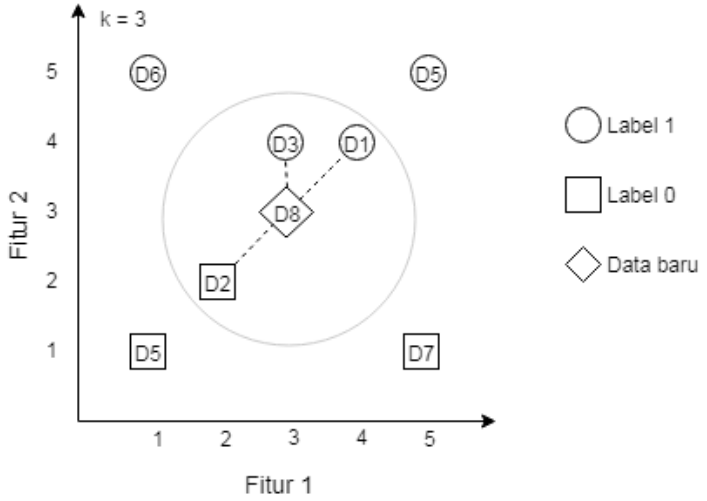
Nilai *k* yang digunakan dicobakan secara empiris[13][14]. Pada awalnya penelitian ini menggunakan 30 nilai *k* (3-33). Namun setelah beberapa percobaan yang dilakukan tidak terjadi perubahan terhadap hasil yang didapat pada nilai *k* = 17 dan seterusnya. Pada percobaan selanjutnya hanya digunakan 14 nilai *k* (3-16). Sehingga yang dibandingkan hanya 14 nilai *k* (3-16).

Penerapan *k-NN* pada klasifikasi teks tidak jauh berbeda dengan penerapan pada umumnya. Pada tahap *preprocessing* hingga pembobotan dengan *TF-IDF* serta penerapan nilai *threshold* didapatkan fitur pada data seperti yang dicontohkan pada tabel 2 berikut.

Tabel 2 Contoh data untuk penerapan *k-NN*

Data	Fitur 1	Fitur 2	Label
D1	4	4	1
D2	2	2	0
D3	3	4	1
D4	5	5	1
D5	1	1	0
D6	1	5	1
D7	5	1	0
D8	3	3	?

Berdasarkan contoh data diatas, terdapat data D1 – D7 yang telah memiliki label dan D8 yang belum memiliki label. Seperti yang telah disebutkan sebelumnya, pada proses klasifikasi akan dihitung *euclidean distance* datri D8 keseluruh data. Gambar berikut ilustrasi proses klasifikasi jika menggunakan nilai *k* = 3.



Gambar 4 Contoh penerapan *k-NN*

Setelah *euclidean distance* dihitung, maka diambil sebanyak *k* data terdekat. Pada ilustrasi diatas nilai *k* yang digunakan adalah 3. Berdasarkan ilustrasi penghitungan diatas, tiga data terdekat dengan D8 adalah D1, D2 dan D3. Selanjutnya akan dihitung label terbanyak, berdasrakan contoh diatas terdapat dua label 1 dan satu label 0 sehingga D8 akan diklasifikasi sebagai label 1. Hal tersebut dilakukan untuk masing-masing kelas.

3.5 *Problem Transformation*

Setelah klasifikas dilakukan, didapatkan prediksi untuk tiap label. Namun label masih terpisah antar satu dan yang lainnya. Contoh hasil prediksi tiap label digambarkan pada tabel 3 berikut.

Tabel 3 Hasil prediksi sebelum *problem transformation*

Data	Label
D1	0
D2	1
D3	0

Agar penghitungan nilai *hamming loss* dapat dilakukan, diperlukan *problem transformation*. Pada penelitian ini *problem transformation* yang digunakan adalah *binary relevance*[7]. Sehingga hasil prediksi menjadi seperti pada tabel 4 berikut.

Tabel 4 Hasil prediksi setelah *problem transformation*

Data	Label 1	Label 2	Label 3
D1	0	1	1
D2	1	0	0
D3	0	0	1

3.6 Evaluasi dan Hasil

Pada klasifikasi multilabel salah satu evaluasi yang umum digunakan adalah *hamming loss*[15]. Nilai *hamming loss* sendiri adalah untuk menghitung berapa banyak kesalahan pada klasifikasi yang dilakukan[16]. Sehingga semakin kecil nilai *hamming loss* yang didapat semakin baik. Penghitungan *hamming loss* dilakukan seperti persamaan(5) berikut.

$$h = 1 - \left(\frac{1}{n}\sum_{i=1}^{n}\frac{y_1 \cap \hat{y}_1}{y_1 \cup \hat{y}_1} + \frac{y_2 \cap \hat{y}_2}{y_2 \cup \hat{y}_2} + \cdots + \frac{y_n \cap \hat{y}_n}{y_n \cup \hat{y}_n}\right) \quad (5)$$

$$F_1 = 2 \times \frac{precision \times recall}{precision + recall} \quad (6)$$

Pada persamaan(5) *h* merupakan nilai dari *hamming loss*. Sedangkan y_n merupakan label asli pada satu data, $\hat{y}_n$ merupakan label prediksi pada satu data dan *n* merupakan jumlah seluruh data. Selain itu, evaluasi juga akan dihitung menggunakan *F1-Score* untuk nilai *k* terbaik dari masing-masing percobaan berdasarkan penghitungan *hamming loss*. Penghitungan *F1-Score* dilakukan seperti persamaan(6).

4. Evaluasi

4.1 Hasil Pengujian

Pada percobaan pertama digunakan *feature extraction* yaitu *n-gram* dengan nilai n = 1 *(unigram)* dan n = 2 *(bigram)*. Pada *tf-idf* dicobakan dengan satu nilai *threshold* yaitu 2 dan tanpa *threshold*. Hasil pada percobaan pertama digambarkan pada tabel 5 berikut, nilai dengan penulisan tebal merupakan nilai terbaik.

Tabel 5 Percobaan pertama

k	Unigram		Bigram	
	Tanpa threshold	Dengan threshold	Tanpa threshold	Dengan threshold
3	0.1848	0.1818	0.1511	0.8403
4	0.1965	0.1934	0.5178	0.9161
5	0.1602	0.1597	0.1506	0.3488
6	0.1605	0.1589	0.1542	0.3651
7	0.1519	**0.1497**	0.1510	0.1591

8	0.1520	0.1508	0.1510	0.1600
9	0.1525	0.1499	0.1506	0.1519
10	0.1527	0.1503	0.1510	0.1519
11	0.1552	0.1511	0.1510	0.1510
12	0.1516	0.1506	0.1510	0.1510
13	0.1516	0.1506	0.1510	0.1510
14	0.1510	0.1500	0.1510	0.1510
15	0.1514	0.1500	0.1510	0.1510
16	0.1510	0.1510	0.1510	0.1510
Rata-rata	0.1588	0.1570	0.1774	0.2857

Pada percobaan kedua *feature extraction* yang digunakan hanya satu, yaitu *n-gram* dengan nilai n = 1 *(unigram)*. Pada *tf-idf* dicobakan dengan nilai *threshold* sebanyak 4 nilai (1, 2, 3 dam 4). Pada percobaan ini dicobakan dengan penggunaan *stemming* dan tanpa *stemming* pada *preprocessing*. Hasil pada percobaan kedua digambarkan pada tabel 6 berikut , nilai dengan penulisan tebal merupakan nilai terbaik.

Tabel 6 Percobaan kedua

k	Dengan *stemming*				Tanpa *stemming*			
	Threshold = 1	**Threshold = 2**	**Threshold = 3**	**Threshold = 4**	**Threshold = 1**	**Threshold = 2**	**Threshold = 3**	**Threshold =4**
3	0.1701	0.1818	0.1632	0.1737	0.1906	0.1597	0.1632	0.1724
4	0.1881	0.1934	0.1680	0.1793	0.2611	0.1746	0.1680	0.1871
5	0.1536	0.1597	0.1539	0.1538	0.1640	0.1533	0.1539	0.1561
6	0.1513	0.1589	0.1539	0.1542	0.1542	0.1530	0.1539	0.1716
7	0.1499	0.1497	0.1511	0.1497	0.1535	0.1511	0.1511	0.1527
8	0.1505	0.1508	0.1510	0.1492	0.1533	0.1500	0.1510	0.1566
9	0.1500	0.1499	0.1506	0.1488	0.1589	0.1500	0.1506	0.1525
10	0.1505	0.1503	0.1505	0.1500	0.1544	0.1505	0.1505	0.1525
11	0.1514	0.1511	0.1502	0.1497	0.1539	0.1505	0.1502	0.1506
12	0.1510	0.1506	0.1502	0.1497	0.1492	0.1505	0.1502	0.1514
13	0.1505	0.1506	0.1502	0.1503	0.1497	0.1505	0.1502	0.1506
14	0.1510	0.1500	0.1502	0.1502	**0.1491**	0.1510	0.1502	0.1506
15	0.1510	0.1500	0.1502	0.1497	0.1495	0.1510	0.1502	0.1492
16	0.1510	0.1510	0.1502	0.1505	0.1519	0.1510	0.1502	0.1502
Rata-rata	0.1550	0.1570	0.1531	0.1542	0.1638	0.1533	0.1531	0.1574

Pada percobaan ketiga *feature extraction* yang digunakan hanya satu, yaitu *n-gram* dengan nilai n = 1 *(unigram)*. Pada *tf-idf* dicobakan dengan nilai *threshold* sebanyak 4 nilai (1, 2, 3 dan 4). Pada percobaan ini dicobakan dengan penggunaan *stopword removal* dan tanpa *stopword removal* pada *preprocessing*. Hasil pada percobaan kedua digambarkan pada tabel 7 berikut, nilai dengan penulisan tebal merupakan nilai terbaik.

Tabel 7 Percobaan ketiga

k	Dengan *stopword removal*				Tanpa *stopword removal*			
	Threshold = 1	**Threshold = 2**	**Threshold = 3**	**Threshold = 4**	**Threshold = 1**	**Threshold = 2**	**Threshold = 3**	**Threshold =4**
3	0.1701	0.1818	0.1632	0.1737	0.1893	0.1885	0.1785	0.1710
4	0.1881	0.1934	0.1680	0.1793	0.2261	0.2233	0.1909	0.1791
5	0.1536	0.1597	0.1539	0.1538	0.1475	0.1475	0.1510	0.1492
6	0.1513	0.1589	0.1539	0.1542	0.1533	0.1533	0.1552	0.1528
7	0.1499	0.1497	0.1511	0.1497	0.1478	0.1480	0.1480	0.1492
8	0.1505	0.1508	0.1510	0.1492	0.1478	0.1474	0.1475	0.1463
9	0.1500	0.1499	0.1506	0.1488	0.1473	0.1474	0.1470	**0.1461**
10	0.1505	0.1503	0.1505	0.1500	0.1477	0.1473	0.1480	0.1483
11	0.1514	0.1511	0.1502	0.1497	0.1492	0.1480	0.1480	0.1480
12	0.1510	0.1506	0.1502	0.1497	0.1483	0.1480	0.1488	0.1480
13	0.1505	0.1506	0.1502	0.1503	0.1477	0.1473	0.1478	0.1481
14	0.1510	0.1500	0.1502	0.1502	0.1491	0.1486	0.1486	0.1486

15	0.1510	0.1500	0.1502	0.1497	0.1492	0.1488	0.1486	0.1486
16	0.1510	0.1510	0.1502	0.1505	0.1495	0.1491	0.1486	0.1486
Rata -rata	0.1550	0.1570	0.1531	0.1542	0.1571	0.1566	0.1540	0.1523

Pada percobaan keempat *feature extraction* yang digunakan hanya satu, yaitu *n-gram* dengan nilai n = 1 *(unigram)*. Pada *tf-idf* dicobakan dengan nilai *threshold* sebanyak 4 nilai (1, 2, 3 dan 4). Pada percobaan ini dicobakan dengan penggunaan dan tanpa penggunaan *stemming* dan *stopword removal* pada *preprocessing*. Hasil pada percobaan kedua digambarkan pada tabel 8 berikut, nilai dengan penulisan tebal merupakan nilai terbaik.

Tabel 8 Percobaan keempat

k	**Dengan *stemming* dan *stopword removal***				**Tanpa *stemming* dan *stopword removal***			
	Threshold = 1	**Threshold = 2**	**Threshold = 3**	**Threshold = 4**	**Threshold = 1**	**Threshold = 2**	**Threshold = 3**	**Threshold =4**
3	0.1701	0.1818	0.1632	0.1737	0.1870	0.1668	0.1671	0.1735
4	0.1881	0.1934	0.1680	0.1793	0.2017	0.1837	0.1734	0.1950
5	0.1536	0.1597	0.1539	0.1538	0.1538	0.1522	0.1525	0.1578
6	0.1513	0.1589	0.1539	0.1542	0.1524	0.1519	0.1516	0.1680
7	0.1499	0.1497	0.1511	0.1497	0.1503	0.1484	0.1489	0.1519
8	0.1505	0.1508	0.1510	0.1492	0.1488	0.1499	0.1505	0.1577
9	0.1500	0.1499	0.1506	0.1488	0.1483	**0.1472**	0.1478	0.1533
10	0.1505	0.1503	0.1505	0.1500	0.1497	0.1508	0.1486	0.1577
11	0.1514	0.1511	0.1502	0.1497	0.1500	0.1480	**0.1472**	0.1511
12	0.1510	0.1506	0.1502	0.1497	0.1483	0.1495	0.1491	0.1502
13	0.1505	0.1506	0.1502	0.1503	0.1481	0.1491	**0.1472**	0.1502
14	0.1510	0.1500	0.1502	0.1502	0.1500	0.1500	0.1477	0.1486
15	0.1510	0.1500	0.1502	0.1497	0.1506	0.1491	0.1477	0.1475
16	0.1510	0.1510	0.1502	0.1505	0.1519	0.1505	0.1491	**0.1472**
Rata -rata	0.1550	0.1570	0.1531	0.1542	0.1565	0.1534	0.1520	0.1578

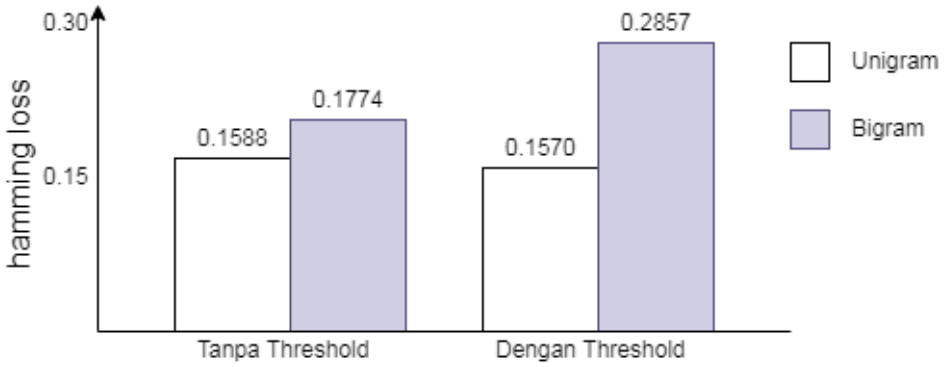
Masing-masing hasil terbaik pada percobaan yang telah disebutkan diatas, akan dihitung nilai *F1-Score*. Penghitungan dilakukan terhadap percobaan metode dan nilai *k* terbaik dilihat berdasarkan nilai *hamming loss*, Hasil penghitungan *F1-Score* dapat dilihat pada tabel 9 berikut, nilai dengan penulisan tebal merupakan nilai terbaik.

Tabel 9 Nilai *F1-score* pada *k* terbaik

Percobaan	***k***	**Ekstraksi Firtur**	**Threshold**	***Stemming***	***Stopword Removal***	***F1-Score***
1	7	*Unigram*	2	Ya	Ya	0.8503
2	14	*Unigram*	1	Tidak	Ya	0.8509
3	9	*Unigram*	4	Ya	Tidak	**0.8539**
4	9	*Unigram*	2	Tidak	Tidak	0.8528
4	11	*Unigram*	3	Tidak	Tidak	0.8528
4	13	*Unigram*	3	Tidak	Tidak	0.8528
4	16	*Unigram*	4	Tidak	Tidak	0.8528

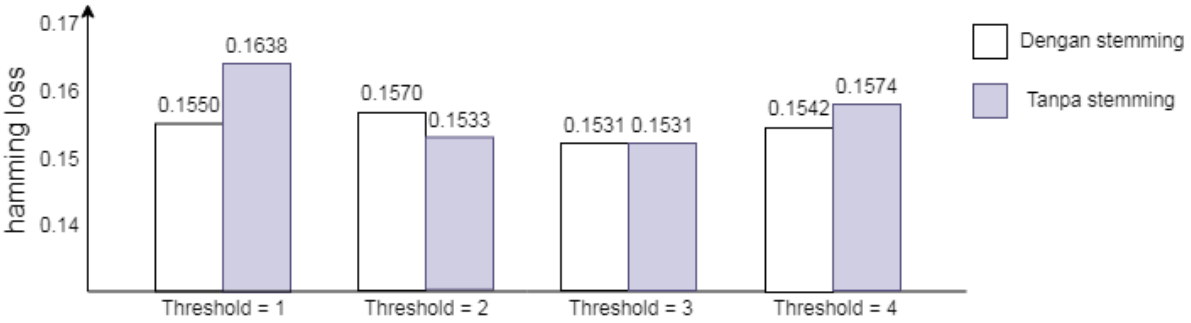
4.2 Analisis Hasil Pengujian

Pada percobaan pertama didapatkan nilai *hamming loss* terbaik sebessar 0.1497. Dari percobaan pertama diketahui penggunaan *unigram* mendapat nilai *hamming loss* lebih baik dibanding *bigram*. Hal ini dikarenakan *bigram* terlalu menjadi ciri khusus dari suatu data sehingga sulit untuk diklasifikasikan atau bisa dibilang penggunaan *bigram* menyebabkan *overfit* pada suatu data. Pada saat suatu fitur *overfit* terhadap data, maka fitur tersebut tidak dapat menggambarkan ciri dari suatu kelas. Selain itu, dapat diketahui juga bahwa penggunaan *threshold* dapat meningkatkan hasil evaluasi. Dua hal tersebut juga didukung dengan didapatkannya nilai rata-rata *hamming loss* terbaik pada *unigram* dengan menggunakan *threshold.*. Perbandingan hasil percobaan pertama dapat dilihat pada gambar 5 berikut.



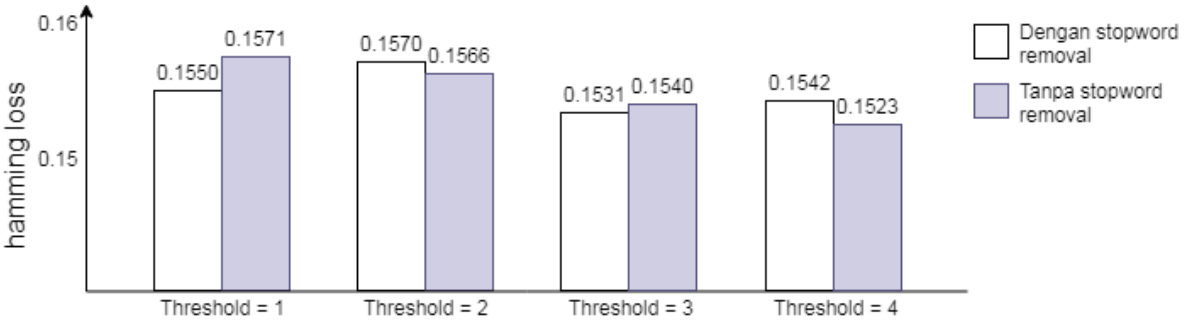
Gambar 5 Perbandingan *unigram* dan *bigram*

Berdasarkan hasil percobaan pertama, pada percobaan berikutnya dicobakan dengan menggunakan *unigram* sebagai *feature extraction* dan penggunaan *threshold* yang nilainya akan diuji coba ke beberapa nilai. Pada percobaan kedua ini, didapatkan nilai *hamming loss* terbaik sebesar 0.1491. Pada percobaan ini dilakukan pengujian terhadap pengaruh *stemming* terhadap hasil evaluasi. Sedangkan untuk nilai *threshold* tetap digunakan. Hasil percobaan kedua mendapatkan hasil seperti yang ditunjukan pada gambar 6 berikut.



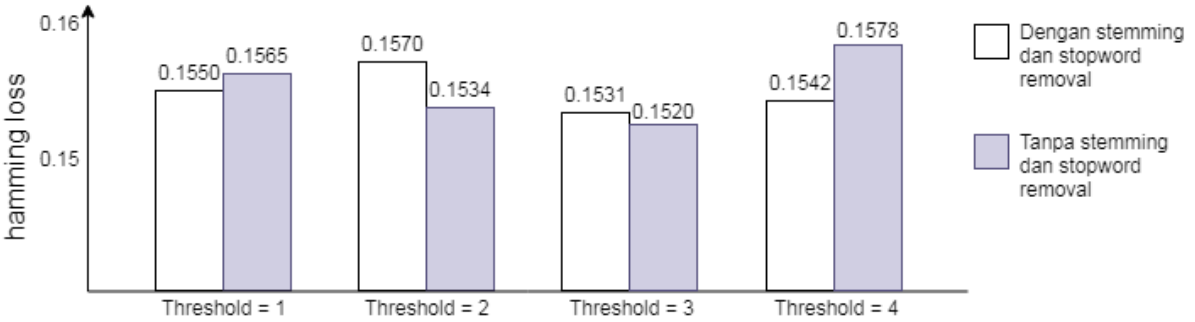
Gambar 6 Perbandingan penggunaan *stemming*

Pada percobaan ketiga didapatkan didapatkan nilai *hamming loss* yang tidak hanya terbaik pada percobaan ini, tapi juga terbaik dibanding percobaan-percobaan sebelumnya yaitu 0.1461. Nilai ini didapat dari pada nilai *threshold* = 4 dan penghapusan *stopword removal* pada *preprocessing*. Perbandingan hasil pada percobaan ketiga digambarkan seperti pada gambar 7 berikut.



Gambar 7 Perbandingan penggunaan *stopword removal*

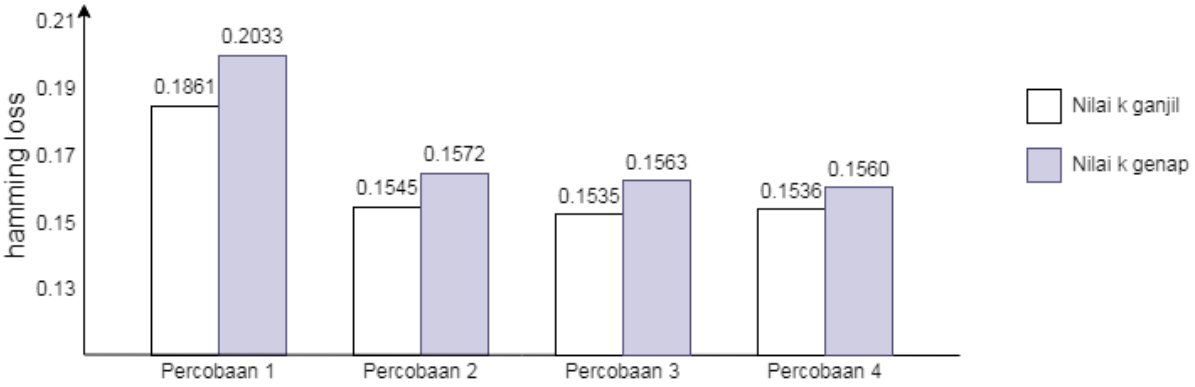
Pada percobaan keempat didapatkan nilai *hamming loss* terbaik sebesar 0.1472. Nilai ini ditemukan pada empat kondisi yaitu (1) k =*9* dan nilai *threshold* = 2, (2) k = 11 dan nilai *threshold* = 3, (3) k = 13 dan nilai *threshold* = 3 dan (4) k = 16 dan nilai *threshold* = 4. Nilai tersebut didapatkan pada penghapusan *stemming* dan *stopword removal* pada tahap *preprocessing*. Perbandingan hasil percobaan keempat dapat dilihat seperti pada gambar 8 berikut.



Gambar 8 Perbandingan penggunaan *stemming* dan *stopword removal*

Pada percobaan yang dilakukan, penggunaan nilai *thresohold* yang lebih kecil tidak selalu mendapatkan hasil yang lebih baik dan hal ini tidak sesuai dengan asumsi yang telah disebutkan. Hal ini bisa terjadi karena seluruh nilai *threshold* yang digunakan cenderumg kecil.

Pada empat percobaan yang dilakukan diapatkan sebanyak enam nilai *k* terbaik pada tiap percobaan yaitu 7, 9, 11, 13, 14, 16. Nilai *k* terbaik pada percobaan pertama adalah 7. Nilai *k* terbaik pada percobaan kedua adalah 14. Nilai *k* terbaik pada percobaan ketiga adalah 9. Sedangkan pada percobaan keempat didapatkan didapatkan empat nilai *k* terbaik yaitu 9, 11, 13, 16. Berdasarkan *k* terbaik dari tiap percobaan, dapat dilihat bawa ganjil atau genapnya nilai *k* yang digunakan dapat meningkatkan hasil evaluasi. Hal tersebut karena dari enam nilai *k* terbaik yang didapatkan, empat diantaranya adalah angka ganjil. Pernyataan tersebut senada rata-rata *hamming loss* pada nilai *k* ganjil selalu lebih kecil dibanding *k* genap pada tiap percobaan. Perbandingan dari pernyataan tersebut dapat dilihat pada gambar 9 berikut.



Gambar 9 Perbandingan nilai *k* ganjil dan genap

Pada gambar 9 dapat dilihat bahwa *k* dengan nilai yang ganjil memiliki nilai *hamming loss* lebih baik dibanding genap. Hal ini dikarenakan nilai *k* genap memiliki kemungkinan keliru pada proses klasifikasi. Hal tersebut terjadi apabila data terdekat dengan data yang sedang diklasifikasi berjumlah sama. Saat jumlah data sama saat *vote*, hasil prediksi tidak akan akurat. Hal ini karena, proses prediksi kelas akan di-*handle* oleh bahasa pemograman yang digunakan. Pada bahasa pemograman yang digunakan penelitian ini (python 3.6), apabila jumlah kelas terdekat dengan data sama maka data akan diklasifikasi sebagai kelas yang berada pada posisi pertama pada proses *sorting*. Sehingg klasifikasi tidak berjalan sebagaimana semestinya.

Pada tiap-tiap nilai *k* terbaik yang didapat pada masing-masing percobaan dihitung *F1-Score* sebagai pembanding dengan penelitian sebelumnya. Berdasarkan penghitungan yang dilakukan (dapat dilihat pada tabel 8), didapatkan *F1-Score* terbaik sebesar 0.8539. Nilai tersebut bisa dikatan berhasil karena lebih baik dibanding penelitian sebelumnya[4].

5. Kesimpulan

Berdasarkan penelitian yang dilakukan dan analisis hasil percobaan yang didapat, dapat disimpulkan bahwa pembangunan klasifikasi Hadis multi label menggunakan *k-NN* berhasil dilakuan dan mendapat hasil yang lebih baik dari penelitian sebelumnya. Hal tersebut dibuktikan dengan nilai *F1-Score* yang didapatkan lebih baik dibanding penelitian sebelumnya yaitu sebesar 0.8539 sedangkan nilai *F1-Score* yang didapat pada penelitian sebelumnya adalah sebesar 0.85. Sedangkan untuk *feature extraction* terbaik yang didapatkan bedasarkan hasil analisa adalah *unigram*. Hal tersebut dibuktikan pada perbandingan nilai *hamming loss* pada

klasifikasi dengan *feature extraction unigram* mendapat nilai yang lebih baik dibanding klasifikasi dengan *feature extraction bigram*. Sedangkan untuk analisis nilai *k* pada *k-NN* diketahui bahwa nilai *k* terbaik didapatkan dengan beberapa percobaan. Percobaan tersebut meliputi mengubah *feature extraction*, penambahan nilai *threshold* pada *TF-IDF* dan melakukan perubahan pada tahap *preprocessing*. Nilai *k* terbaik disini disini adalah nilai *k* dengan nilai *hamming loss* terbaik. Nilai *hamming loss* terbaik adalah 0.1461 didapatkan pada nilai *k* = 9, *feature extraction unigram*, *threshold* pada *TF-IDF* dengan nilai 4, dan penghilangan *stopword removal* pada tahap *preprocessing*. Disisi lain, pada percobaan yang dilakukan didapatkan masing-masing nilai *k* terbaik adalah 7, 11, 13, 14 dan 16. Sehingga, dapat disimpulkan bahwa untuk mendapatkan nilai *k* terbaik pada klasifikasi teks adalah dengan cara melakukan (1) percobaan pada penggunaan *feature extraction*, (2) penambahan *threshold* pada *feature selection*, (3) perubahan pada tahap *porcessing* dan (4) menggunakan nilai *k* yang ganjil.

Daftar Pustaka

[1] Zein, Muhammad Ma'shum, 2008, Ummul Hadits dan Musthalah Hadits, Darul Hikmah.

[2] Naji Al-Kabi, M., Kanaan, G., Al-Shalabi, R., Al-Sinjilawi, S. I., dan AlMustafa, R. S., 2005, Al-Hadith text classifier, Journal of Applied Sciences 5 (pp. 584- 587).

[3] Bruno Trstenjak, Sasa Mikac dan Dzenana Donko, 2014, KNN with TF-IDF Based Framework for Text Categorization, Procedia Engineering 69 (pp. 1356-1364). Elsevier.

[4] Al Faraby, S., Jasin, E.R.R. dan Kusumaningrum, A., 2018, Classification of hadith into positive suggestion, negative suggestion, and information. In Journal of Physics: Conference Series (Vol. 971, No. 1, p. 012046). IOP Publishing.

[5] Fauzan, H., Adiwijaya, A. and Al-Faraby, S., 2018. Pengklasifikasian Topik Hadits Terjemahan Bahasa Indonesia Menggunakan Latent Semantic Indexing dan Support Vector Machine. JURNAL MEDIA INFORMATIKA BUDIDARMA, 2(4), (pp.131-139).

[6] Muhammad Romi Ario Utomo, Yuliant Sibaroni, 2018, Text Classification of British English and American English Using Support Vector Machine, ICoICT.

[7] Min-Ling Zhang dan Zhi-Hua Zhou, 2013, A Review on Multi-Label Learning Algorithms, IEEE.

[8] Bakar, M.Y.A., Adiwijaya, Al Faraby, S., 2018, Multi-Label Topic Classification of Hadith of Bukhari (Indonesian Language Translation) Using Information Gain and Backpropagation Neural Network. In 2018 International Conference on Asian Language Processing (IALP) (pp. 344-350). IEEE.

[9] Gugun Mediamer, Adiwijaya dan Al Faraby, S., 2018, Development of Rule-Based Feature Extraction in Multilabel Text Classification for Bukhari Hadith (Indonesian Translation), ICADEIS.

[10] J. K. Srict, 2015, A Study of Text Classification Natural Language Processing Algorithms for Indian Languages, VNSGU Journal Of Science and Technology vol. 4, no. 1 (pp. 163–164). VNSGU.

[11] L. H. Patil dan M. Atique, 2013, A novel approach for feature selection method TF-IDF in document clustering, in Proceedings of the 2013 3rd IEEE International Advance Computing Conference, IACC.

[12] Y. Yang dan J. O. Pedersen, 1997, A Comparative Study on Feature Selection in Text Categorization, Proceedings of the Fourteenth International Conference on Machine Learning (pp. 412–420).

[13] Suyanto, 2017, Data Mining Untuk Klasifikasi dan Klasterisasi Data, 1st ed. Penerbit Informatika (pp. 211-212).

[14] Adiwijaya, Aulia, M.N., Mubarok, M.S., Novia, W.U. and Nhita, F., 2017, May. A comparative study of MFCC-KNN and LPC-KNN for hijaiyyah letters pronounciation classification system. In 2017 5th International Conference on Information and Communication Technology (ICoIC7) (pp. 1-5). IEEE.

[15] Jose M. Moyano, Eva L. Gibaja, Krzysztof J. Cios, Sebastian Ventura, 2018, Review of Ensembles of Multi-Label Classifier: Models, Experimental Study and Prospects¸ Information Fusion Vol 44 (pp. 11).

[16] Eva Gibaja dan Sebastian Ventura, 2015, A Tutorial on Multi-Label Learning, ACM Computing Surveys.