Pengklasifikasian Topik Hadits Terjemahan Bahasa Indonesia Menggunakan *Latent Semantic Indexing* dan *Support Vector Machine*

Hafizh Fauzan[1], Adiwijaya[2], Said Al Faraby[3]

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

Abstrak

Hadits dijadikan sumber hukum dalam agama Islam selain Al-Qur'an, Ijma dan Qiyas, dimana dalam hal ini, hadits merupakan sumber hukum kedua setelah Al-Qur'an. Penelitian ini bertujuan untuk membangun suatu sistem yang dapat mengkategorikan kelas dari hadits shahih Bukhari terjemahan Bahasa Indonesia. Topik ini dipilih untuk membantu masyarakat Islam yang ingin memahami dari setiap hadits merupakan berupa informasi, larangan atau anjuran. *Support Vector Machine* digunakan karena dapat melakukan klasifikasi dengan memberikan performansi yang baik untuk dataset yang memiliki fitur yang sangat banyak. *Latent Semantic Indexing* sebagai metode pemilihan fitur digunakan karena dapat mereduksi fitur dengan menghilangkan fitur yang tidak penting (*noise term*). Penelitian ini juga memanfaatkan metode *Bootstrap Aggregating* (*Bagging*) untuk meningkatkan akurasi. Hasil akurasi yang didapatkan dengan menggunakan *Latent Semantic Indexing* dan *Bagging* pada klasifikasi *Support Vector Machine* sebesar 84.67% pada data hadits *single label*.

Kata kunci : hadits, *support vector machine*, pemilihan fitur, *latent semantic indexing*, *bootstap aggregating*, klasifikasi teks, *single label*

Abstract

Hadith is used as the source of Islamic law other than Qur'an, Ijma and Qiyas, hadith is the second of Islamic law after the Qur'an. This study attempted to build a system than can classify shahih hadith of Bukhari in Indonesian Translation. This topic was chosen to help Muslims who want to understand from each hadith is in the form of informations, prohibitions or suggestions. Support Vector Machine was chosen because it can perform classification by providing good performance for dataset with a large number of features. Latent Semantic Indexing as a feature selection method was chosen because it can reduce features by eliminating unimportant features (noise term). This study also using Bootstrap Aggregating (Bagging) method to improve accuracy. The accuracy results show that by using Latent Semantic Indexing and Bagging on Support Vector Machine classification is 84.67% of single label hadith data.

Keywords: hadith, support vector machine, feature selection, latent semantic indexing, bootstrap aggregating, text classification, single label

1. Pendahuluan

Hadits adalah segala perkataan (sabda), perbuatan dan ketetapan atau persetujuan dari Nabi Muhammad SAW yang dijadikan ketetapan ataupun hukum dalam agama Islam. Hadits dijadikan sumber hukum dalam agama Islam selain Al-Qur'an, Ijma dan Qiyas. Kedudukan hadits merupakan sumber hukum kedua setelah Al-Qur'an[1]. Pada dewasa ini teknologi informasi yang terus berkembang semakin mempermudah masyarakat dalam mendapatkan informasi, terutama dengan menggunakan internet. Informasi yang didapatkan di internet lebih banyak informasi berupa dokumen teks. Hal ini berpengaruh terhadap bertambahnya jumlah hadits shahih Bukhari terjamahan Bahasa Indonesia berupa dokumen teks. Bagi masyarakat yang tidak memiliki cukup waktu, mengakibatkan kesulitan untuk memahami setiap makna dari setiap hadits yang disebabkan jumlah hadits yang sangat banyak. Jumlah dokumen yang sangat banyak mendorong peneliti untuk melakukan penelitian dalam membuat sistem pengolahan terhadap kumpulan dokumen hadits tersebut kedalam kelas topik larangan, anjuran, atau informasi. Mengelola dokumen teks dengan jumlah yang banyak merupakan hal yang sulit untuk dilakukan secara manual. Oleh karena itu, penelitian ini menggunakan teknik yang dapat mengkategorikan kelas dari dokumen secara otomatis. Klasifikasi merupakan salah satu teknik dalam *text mining* dan *data mining* yang digunakan dalam proses mencari model atau fungsi yang menjelaskan atau membedakan kelas-kelas pada data dan konsep yang bertujuan untuk menggunakan model tersebut dalam melakukan prediksi terhadap *data testing*. Salah satu metode klasifikasi adalah *Support Vector Machine* (SVM) yang memiliki kelebihan yaitu mampu memberikan performansi yang baik dalam dataset yang memiliki fitur yang sangat banyak. *Latent Semantic Indexing* (LSI) sebagai metode pemilihan fitur digunakan untuk mereduksi fitur dengan menghilangkan fitur

yang tidak penting dari data (*noise term*). Terdapat dua metode untuk meningkatkan tingkat akurasi prediksi dan model dalam klasifikasi yaitu *Boosting* dan *Bootstrap Agregating* (*bagging*). Sistem klasifikasi pada penelitian ini akan memanfaatkan metode *bagging* dengan tujuan untuk meningkatkan nilai akurasi.

Terdapat beberapa batasan masalah dalam penelitian ini adalah *dataset* yang digunakan adalah data hadits shahih Bukhari terjemahan Bahasa Indonesia dengan jumlah 1650 hadits dengan komposisi sebanyak 500 untuk setiap label informasi, larangan dan anjuran sebagai *data training*, dan sebanyak 50 untuk setiap label informasi, larangan dan anjuran sebagai *data testing*. Dilakukan penghilangan *stopwords* pada proses *preprocessing* dataset.

Pada jurnal ini, disusun sebagai berikut. Pada bagian kedua, terdapat studi terkait yang menjelaskan mengenai metode penelitian sebelumnya yang berkaitan dengan metode penelitian yang diterapkan pada penelitian yang sedang dikerjakan ini. Pada bagian ketiga, menjelaskan tentang sistem klasifikasi yang dibangun dan diimplementasikan pada penelitian ini dengan judul pengklasifikasian topik hadits shahih Bukhari terjemahan Bahasa Indonesia menggunakan *Latent Semantic Indexing* dan *Support Vector Machine*. Pada bagian keempat, terdapat evaluasi dan analisis hasil pengujian yang telah didapatkan. Kemudian pada bagian kelima, menjelaskan kesimpulan yang didapatkan pada penelitian ini.

2. Studi Terkait

Pada dewasa ini metode klasifikasi telah banyak dikembangkan, antara lain *Support Vector Machine* (SVM), *Naïve Bayessian Classifier* (NBC), *Decision Tree* (DT), *K-Nearest Neighbour* (KNN) dan metode klasifikasi lainnya. Dari semua metode klasifikasi yang telah dikembangkan, KNN dan SVM telah diakui menghasilkan akurasi lebih baik dibandingkan dengan metode klasifikasi pada teks lainnya, dimana SVM juga memiliki kemampuan yang baik dalam melakukan klasifikasi pada dimensi yang tinggi[2]. Penelitian terkait yang telah dilakukan oleh Wulandini & Nugroho (2009), membandingkan metode klasifikasi pada teks NBC dengan metode SVM, C4.5 dan KNN. Hasil penelitian tersebut menggambarkan akurasi dari setiap metode, dan metode yang menghasilkan nilai akurasi terbaik adalah SVM dengan nilai akurasi 92%, diikuti dengan NBC dengan nilai akurasi 90%, C4.5 dengan nilai akurasi 77.5% dan KNN dengan nilai akurasi 50%.

Penelitian terkait dalam menggabungkan metode klasifikasi dengan pemilihan fitur *Latent Semantic Indexing* (LSI) dilakukan oleh Ana CardosoCachopom & Arlindo I., Olivera (2007). Pada penelitian tersebut menggambarkan kombinasi antara *K-Nearest Neighbour* (KNN) dan *Support Vector Machine* (SVM) dengan *Latent Semantic Indexing* (LSI), menghasilkan bahwa SVM-LSI adalah metode yang menyajikan kinerja terbaik pada penelitian tersebut, dengan mengetahui bahwa SVM selalu menjadi metode klasifikasi dengan performansi terbaik[3]. Hasil nilai akurasi rata-rata yang didapat oleh SVM-LSI adalah 0.8385, dan KNN-LSI memiliki nilai akurasi rata-rata 0.7904. Terdapat juga hasil klasifikasi tanpa pemilihan fitur, hasil nilai akurasi rata-rata untuk SVM adalah 0.8383, hasil nilai akurasi rata-rata untuk KNN adalah 0.7540, dan hasil nilai akurasi rata-rata untuk LSI adalah 0.7702.

Pada penelitian penerapan teknik *bagging* pada algoritma klasifikasi C4.5 untuk mendeteksi penyakit *liver* dilakukan oleh Aditya Prtama, Fisal Ajang Sidik, Yuliani Dwi Asih, Ita Fitriani dan Retno Dwi Khusnul Khotimah (2017). Penelitian ini menggunakan *bootstrap aggregating* (*bagging*) untuk meningkatkan tingkat akurasi prediksi dan ketepatan dalam klasifikasi dengan masalah ketidakseimbangan kelas sangat menghambat kinerja klasifikasi banyak algoritma klasifikasi[8]. Pada penelitian tersebut terbukti teknik *bagging* efektif pada algoritma klasifikasi *decision tree* dapat meningkatkan hasil akurasi dari klasifikasi dataset *liver* dengan hasil akurasi menggunakan *bagging* 96,2003%, sedangkan tanpa *bagging* hasil akurasi yang didapat hanya 87,9102%.

Penelitian yang menerapkan *Bootstrap Aggregating* (*bagging*) pada algoritma klasifikasi *Support Vector Machine* (SVM) untuk klasifikasi *leukemia* yang dilakukan oleh Billy Eden William Asrul (2014). Penggunaan metode klasifikasi SVM sebagai metode klasifikasi yang efektif untuk dataset berdimensi tinggi dan *bagging* yang memiliki manfaat untuk meningkatkan akurasi dengan menciptakan banyak model klasifikasi[20]. Penerapan *Bagging* melatih setiap SVM secara terpisah dengan menggunakan teknik *bootstrap*, maka mengambil kesimpulan kinerja masing-masing SVM oleh suara terbanyak. Peneran bagging pada SVM memberikan akutasi antara 87,5% - 92,5%, area dibawah kurva ROC antara 98,0% - 99,2%, *F-measure* antara 86,5% - 92,7% dan SVM tunggal memberikan akurasi 87,5%, area bawah kurva ROC antara 98,0% - 98,8%, dan *F-measure* antara 86,5% - 87,8%.

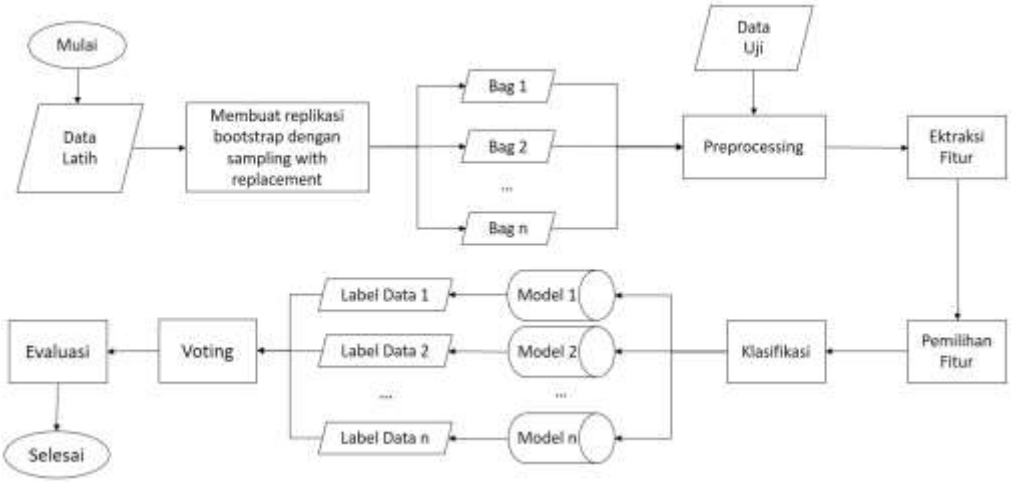
3. Sistem yang Dibangun

Dataset yang digunakan pada penelitian ini adalah hadits shahih Bukhari dari aplikasi lidwa dengan terjemahan Bahasa Indonesia. Data asli hadits ini belum memiliki label terhadap kelas informasi, larangan dan anjuran, sehingga dilkakukan proses pelabelan pada data secara manual. Data yang digunakan sebanyak 1650 data dengan label *single label*, terdiri dari 550 kelas informasi, 550 kelas larangan dan 550 kelas anjuran.

Tabel 3.1. Contoh *dataset* hadits

No.	Bab	Hadits	LEN	Kelas
11	6	Kamu memberi makan, mengucapkan salam kepada orang yang kamu kenal dan yang tidak kamu kenal.	93	informasi
206	147	Jika salah seorang dari kalian mengantuk saat shalat, hendaklah tidur (dahulu) hingga ia mengetahui apa yang ia baca.	117	anjuran
1024	686	Seorang wanita tidak boleh mengadakan perjalanan diatas tiga hari kecuali bersama mahramnya.	92	Larangan

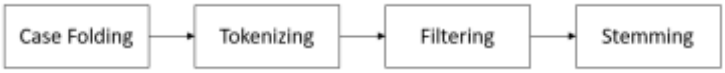
Pada penelitian ini, sistem dibangun untuk mampu melakukan klasifikasi *single label* terhadap hadits shahih Bukhari terjemahan Bahasa Indonesia ke dalam salah satu kelas informasi, larangan atau anjuran. Sistem pada penelitian ini memiliki beberapa proses utama. Proses pertama adalah *bagging* yang melakukan replikasi *data training* sebanyak 20 *bag* dengan *sampling with replacement*. Proses kedua adalah *preprocessing*, setiap *bag* dilakukan proses *preprocessing* yang bertujuan untuk mempersiapkan data teks menjadi lebih terstruktur dan bersih. Proses ketiga adalah proses pembobotan yang akan memberikan nilai bobot untuk setiap fitur terhadap dokumen. Proses keempat adalah proses pemilihan fitur yang digunakan untuk mereduksi jumlah fitur yang akan menjadi masukan di proses klasifikasi. Proses terakhir adalah proses klasifikasi yang akan menentukan kelas dari setiap hadits dengan melakukan prediksi terhadap data *testing* dan kemudian menghitung nilai akurasinya. Alur sistem pada penelitian ditunjukkan pada Gambar 3.1.



Gambar 3.1. Alur sistem klasifikasi

Proses *bagging* melakukan replikasi *data training* menjadi 20 *bag* dengan cara *sampling with replacement*. Kemudian setiap *bag* dilakukan proses *preprocessing*, pembobotan, pemilihan fitur menggunakan *Latent Semantic Indexing* (LSI) dan klasifikasi menggunakan *Support Vector Machine* (SVM). Model-model klasifikasi yang dihasilkan nanti akan menjadi *bagged classifier*, dimana data *testing* yang ingin diklasifikasikan tersebut dicoba untuk masing-masing model klasifikasi, dan hasil klasifikasi akhir untuk data tersebut akan ditentukan berdasarkan *voting* terbanyak berdasarkan prediksi dari model-model klasifikasi tersebut.

Berdasarkan struktur data teks yang tidak terstruktur, maka diperlukan beberapa tahap awal yang pada intinya adalah mempersiapkan agar data teks dapat diubah menjadi lebih terstruktur. Proses *preprocessing* ini meliputi *case folding*, *tokenizing*, *filtering* dan *stemming*.



Gambar 3.2. Alur *Preprocessing* dataset hadits

1. *Case Folding*, tidak semua dokumen teks konsisten dalam penggunaan huruf kapital. Oleh karena itu, peran *case folding* diperlukan dalam mengkonversi keseluruhan teks dalam dokumen menjadi suatu bentuk standar (biasanya huruf kecil). Secara garis besar *case folding* adalah mengubah semua huruf dalam dokumen menjadi huruf kecil.
2. *Tokenizing*, adalah tahap pemotongan *string input* berdasarkan tiap kata yang menyusunnya. Tokenisasi secara garis besar memecah sekumpulan karakter dalam suatu teks ke dalam satuan kata.
3. *Filtering*, adalah tahap mengambil kata-kata penting dari hasil token. Bisa menggunakan algoritma *stopword* (membuang kata kurang penting) atau *worldlist* (menyimpan kata penting). *Stopword* adalah

kata-kata yang tidak deskriptif yang dapat dibuang dalam pendekatan *bag-of-words*. Penghilangan *stopword* ini dapat mengurangi ukuran indeks, waktu pemrosesan dan dapat mengurangi *noise*.

4. *Stemming*, teknik *stemming* diperlukan selain untuk memperkecil jumlah indeks yang berbeda dari suatu dokumen, juga untuk melakukan pengelompokan kata-kata lain yang memiliki kata dasar dan arti yang serupa, namun memiliki bentuk atau form yang berbeda karena mendapatkan imbuhan yang berbeda. Proses *stemming* pada dokumen teks berbahasa Indonesia menghilangkan semua kata imbuhan baik itu sufiks dan prefiks.

Selanjutnya akan dilakukan tahap pembobotan untuk mendapatkan nilai bobot dati setiap fitur terhadap dokumen. Representasi dokumen biasa didapatkan berdasarkan memilih kata-kata yang penting di dalam dokumen berdasarkan frekuensinya. *Term Frequency – Inverse Document Frequency* atau yang biasa disebut dengan TF-IDF merupakan gabungan dari *term frequency* dan *inverse document frequency*. TF-IDF merepresentasikan kedalam sebuah matriks dimana setiap baris pada matriks merupakan dokumen dan untuk setiap kolomnya menyatakan *feature*/*term* (kata). Pembobotan ini merupakan perhitungan statistika yang digunakan untuk mengevaluasi pentingnya suatu kata dalam dokumen di kumpulan dokumen. Pentingnya peningkatan dari nilai kemunculan suatu kata di dalam dokumen, namun diimbangi dengan frekuensi dari kata tersebut dalam kumpulan dokumen. Nilai bobot yang didapat akan menjadi masukan untuk proses *feature selection* dengan *Latent Semantic Indexing*. Perhitungan bobot kata t pada dokumen dirumuskan dalam persamaan berikut :

$$TFIDF(d,t) = TF(d,t)\ .\ IDF(t) = TF(d,t).\log(\frac{N}{df(t)}) \quad (1)$$

Dimana :

$TFIDF(d,t)$ = bobot kata t terhadap dokumen d
$TF(d,t)$ = jumlah kemunculan kata t dalam dokumen d
N = jumlah dokumen
$df(t)$ = jumlah dokumen yang mengandung kata t

Setelah didapatkan matriks bobot yang merepresentasikan bobot kata terhadap dokumen, proses selanjutnya adalah memilih fitur yang akan menjadi masukan pada proses klasifikasi. Proses ini disebut sebagai pemilihan fitur. Pemilihan fitur ini menggunakan metode *Latent Semantic Indexing* (LSI). *Latent Semantic Indexing* adalah metode pengindeksan dan pencarian yang menggunakan matematika yang disebut *Singular Value Decomposition* (SVD) untuk mengidentifikasi pola hubungan antara istilah dan konsep-konsep yang terkandung dalam sebuah koleksi teks yang tidak terstruktur. LSI didasarkan pada prinsip bahwa kata-kata yang digunakan dalam konteks yang sama cenderung memiliki makna yang sama. Fitur utama LSI adalah kemampuannya untuk mengekstrak konteks konseptual dari suatu tubuh teks dengan mendirikan asosiasi antara istilah-istilah yang terjadi dalam konteks yang serupa[21].

Pada SVD, jika diketahui A adalah matriks, dimana m berupa baris yang menunjukkan *feature*/*term* (kata) yang unik dan n berupa kolom yang menyatakan dokumen. Setiap *cell* menyatakan frekuensi kata pada setiap dokumen. Selanjutnya LSI melakukan perhitungan *Singular Value Decomposition* (SVD) terhadap matriks A. Proses perhitungan SVD akan menghasilkan 3 matriks. Matriks pertama menjelaskan *original* dari baris *entity* sebagai *vector* dari nilai orotogonal, matriks kedua sebagai sebuah matrik diagonal yang terdiri dari nilai skala terhadap ke tiga komponen matriks dan matriks ketiga menjelaskan kolom original[7]. Jika A berukuran $m\ x\ n$ dimana $m \geq n$, maka SVD (A) didefinisikan :

$$A = USV^T \quad (2)$$

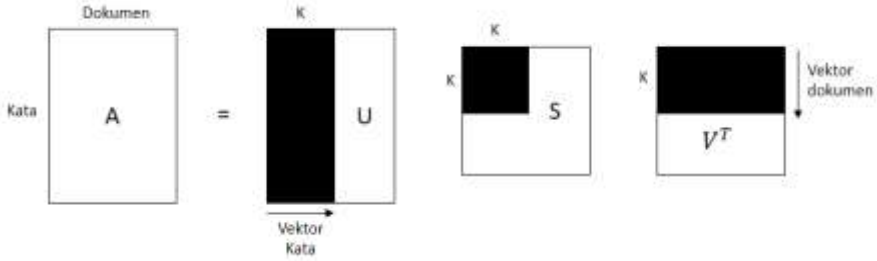
$$\underset{m \times n}{\begin{pmatrix} A \end{pmatrix}} = \underset{m \times n}{\begin{pmatrix} U \end{pmatrix}} \underset{n \times n}{\begin{pmatrix} S \end{pmatrix}} \underset{n \times n}{\begin{pmatrix} V^T \end{pmatrix}}$$

Gambar 3.3. Ilustrasi *Singular Value Decomposition.*

SVD melibatkan pemfaktoran A ke dalam hasil kali USV^T, dimana :

U = matriks kolom *orthogonal* $m\ x\ n$ atau disebut juga dengan *left singular vectors*
S = matriks diagonal $n\ x\ n$
V = matriks *orthogonal* $n\ x\ n$, atau disebut juga dengan *right singular vectors*
V^T = matriks *transpose* dari matriks V

Setelah matriks A di dekomposisi menjadi tiga matriks, selanjutnya dilakukan reduksi dimensi terhadap tiga matriks tersebut. Reduksi dilakukan dengan nilai k yang telah ditentukan, k merupakan besar dimensi matriks yang akan disimpan, dengan syarat k tidak melebihi jumlah dokumen. Reduksi dimensi tidak mengubah nilai dalam matriks. Reduksi dilakukan dengan cara mengambil k baris atau kolom pertama dari ketiga matriks tersebut[3]. Ilustrasi proses reduksi dapat dilihat pada gambar 2-2.



Gambar 3.4, Reduksi dimensi matriks dengan nilai k.

Besaran dari *singular value* menjelaskan tingkat kepentingan terhadap dokumen. Setiap dokumen yang berisi pola kombinasi kata ini direpresentasikan kedalam vektor bentuk tunggal dan dokumen yang merepresentasikan pola ini dengan baik akan memiliki nilai indeks yang paling besar dari vektor. Matriks *right singular vectors* (V^T) menggambarkan tingkat kepentingan tiap kata terhadap dokumen.

Pada penelitian ini, metode klasifikasi yang dibangun adalah *Support Vector Machine* atau biasa disebut SVM. SVM adalah salah satu metode klasifikasi pada *supervised learning*. SVM membagi data menjadi dua kelas dengan memaksimalkan *margin* antara *hyperplane* dengan kedua data terdekat pada kedua kelas[4]. *Hyperplane* yang paling optimal disebut sebagai *support vector*. Untuk memperoleh *hyperplane* yang terbaik diperlukan perhitungan untuk menentukan jarak antara data dengan *hyperplane* yang ada. Pada proses *learning* yang pertama, akan dicari jarak minimum dari setiap data *training* terhadap *hyperplane* dengan menggunakan persamaan berikut :

$$y_i(w.x_i + b) \geq 1 \quad ; i = 1,2,\dots n \tag{3}$$

Dimana x_i merupakan data masukan, i merupakan jumlah data dan y_i merupakan keluaran, sedangkan w dan b merupakan parameter yang akan dicari nilainya.

Feature Space dalam prakteknya biasanya memiliki dimensi yang lebih tinggi vari vektor input (*input space*). Hal ini menyebabkan komputasi pada *feature space* menjadi sangat besar, karena ada kemungkinan *feature space* dapat memiliki jumlah *feature* yang tak terhingga. Selain itu, sulit mengetahui fungsi transformasi yang tepat untuk mengatasi masalah ini, pada SVM digunakan "*kernel trick*". Terdapat beberapa jenis fungsi kernel yang sering digunakan yaitu *polynomial* dan *Radial Basis Function* (RBF), dimana p sebagai parameter *degree* (d) pada kernel *polynomial* dan y sebagai parameter gamma ($\gamma = \frac{1}{2\sigma^2}$) pada kernel RBF [22].

Tabel 3.2. SVM biner *one-against-all*

Jenis Kernel	Defenisi
Polynomial	$K(\overrightarrow{x_i}, \overrightarrow{x_j}) = (\overrightarrow{x_i}.\overrightarrow{x_j} + 1)^p$
RBF	$K(\overrightarrow{x_i}, \overrightarrow{x_j}) = \exp(-\frac{\|\overrightarrow{x_i} - \overrightarrow{x_j}\|^2}{2\sigma^2})$

Pada *dataset* yang digunakan ini terdapat tiga kelas yaitu kelas informasi, kelas anjuran dan kelas larangan. Ada cara untuk mengimplementasikan *multi class* SVM yaitu dengan menggabungkan beberapa SVM biner atau menggabungkan semua data ke dalam sebuah sistem optimasi. Metode "*one against all*" adalah salah satu metode yang dapat diimplementasikan dalam permasalah *multi class* SVM. Pada Penelitian ini membangun tiga buah model SVM biner. Setiap model dilatih dengan semua data untuk dicari solusinya.

Tabel 3.2. SVM biner *one-against-all*

y_i=1	y_i=-1	Hipotesis
Kelas Informasi	Bukan Kelas Informasi	$f_1(x) = (w_1)x + b_1$
Kelas Larangan	Bukan Kelas Larangan	$f_2(x) = (w_2)x + b_2$
Kelas Anjuran	Bukan Kelas Anjuran	$f_3(x) = (w_3)x + b_3$

Gambar 3.5. klasifikasi *one-against-all*

Tahap terakhir adalah mengevaluasi hasil klasifikasi dari sistem yang telah dibangun pada penelitian ini dengan perhitungan *accuracy*. *Accuracy* dipilih sebagai perhitungan evaluasi untuk mengukur performansi dari hasil sistem klasifikasi *single label* yang telah didapatkan pada dokumen *testing* dengan jumlah yang sama pada setiap kelasnya yaitu sebanyak 50 data. *Accuracy* merupakan salah satu perhitungan evaluasi yang umum digunakan untuk pengujian data *single label* dengan menghitung jumlah dokumen *testing* hadits yang diklasifikasikan dengan benar dibagikan dengan jumlah dokumen *testing* hadits keseluruhan. Mengevaluasi banyaknya label prediksi yang sesuai dengan label aktual . Semakin besar nilai *accuracy*, maka performansi *classifier* semakin bagus

$$Accuracy = \frac{Jumlah\ dokumen\ hadits\ yang\ diklasifikaskan\ dengan\ benar}{Jumlah\ dokumen\ hadits}\ x\ 100\ \% \quad (4)$$

4. Evaluasi

Dalam mengevaluasi sistem klasifikasi *single label* pada data hadits shahih Bukhari terjemahan Bahasa Indonesia dengan *Support Vector Machine* yang telah dibangun, dilakukan beberapa skenario pengujian. Pengujian pertama dilakukan untuk mengetahui pengaruh perubahan nilai *degree* dan nilai *gamma* pada kernel *polynomial* dan kernel RBF pada SVM terhadap nilai akurasi. Pengujian kedua dilakukan untuk mengetahui pengaruh pemanfaatan pemilihan fitur LSI pada SVM terhadap nilai akurasi. Pengujian ketiga dilakukan untuk mengetahui pengaruh pemanfaatan *bagging* pada SVM terhadap nilai akurasi. Pengujian keempat dilakukan untuk mengetahui pengaruh pemanfaatan *bagging* dan pemilihan fitur LSI pada SVM terhadap nilai akurasi.

1. Skenario pengujian klasifikasi *Support Vector Machine* (SVM)

   Dilakukan pengujian terhadap klasifikasi *Support Vector Machine* (SVM) untuk mengetahui pengaruh perubahan nilai *degree* dan *gamma* pada kernel *polynomial* dan kernel RBF pada SVM terhadap nilai akurasi. Diperoleh hasil dari pengujian ini dengan nilai *degree* terbaik untuk kernel *polynomial* adalah 1 dengan nilai akurasi yang diberikan sebesar 76.67%, dan nilai *gamma* terbaik untuk kernel RBF adalah 0,02 dengan nilai akurasi yang diberikan sebesar 84.67%.

Tabel 4.1. Hasil pengujian skenario SVM

Kernel	Nilai Parameter	Nilai Akurasi
Polynomial	d=1	**76.67%**
	d=2	65.33%
	d=3	64.00%
RBF	γ =0.01	84.00%
	γ =0.02	**84.67%**
	γ =0.03	78.00%
	γ =0.04	71.33%
	γ =0.05	68.00%

Berdasarkan Tabel 4.1, hasil pengujian skenario diatas dapat dilihat bahwa pada kernel *polynomial* dengan nilai parameter *degree* bernilai 1 memiliki nilai akurasi terbaik sebesar 76.67% dan pada kernel

RBF didapatkan nilai akurasi tertinggi dengan nilai paremeter *gamma* bernilai 0.02 sebesar 84.67%. Hal ini tidak berjalan sesuai dengan prinsip parameter *degree* dan *gamma* pada kernel *polynomial* dan kernel RBF yang pada dasarnya digunakan untuk merepresentasikan data ke dalam *feature space* yang berdimensi tinggi. Jika semakin besar nilai *degree* atau *gamma*, maka semakin tinggi juga *feature space* yang nantinya akan semakin mudah dalam menemukan *hyperplane* terbaik. Namun nilai *degree* atau *gamma* yang terlalu besar akan menyebabkan algoritma kesulitan dalam menentukan *classifier* terbaik karena dimensi data yang terlalu besar dapat mengakibatkan data semakin berjarak dan menyebabkan data kehilangan informasinya[18].

2. Skenario pengujian pengaruh pemilihan fitur *Latent Semantic Indexing* (LSI) pada klasifikasi *Support Vector Machine* (SVM)

Pada skenario pengujian pengaruh pemilihan fitur *Latent Semantic Indexing* (LSI) pada klasifikasi *Support Vector Machine* (SVM) bertujuan untuk melihat pengaruh LSI pada metode SVM dan mengetahui pengaruh perubahan nilai k untuk mereduksi dimensi pada proses pemilihan fitur LSI. Diperoleh hasil dari skenario ini dengan hasil akurasi terbaik pada kernel *polynomial* sebesar 82.67 % dengan nilai k adalah 510, dan hasil akurasi terbaik pada kernel RBF sebesar 87.33% dengan nilai k adalah 520,530 dan 560.

Tabel 4.2. Hasil pengujian skenario SVM-LSI

k dimensi	Polynomial			RBF				
	d =1	d =2	d =3	γ =0.01	γ =0.02	γ =0.03	γ =0.04	γ =0.5
500	82	68.67	50	83.33	84.67	86.67	85.33	81.33
510	**82.67**	68.67	50.67	83.33	84	86.67	86.67	81.33
520	80.67	68.67	50.67	83.33	84	86.67	**87.33**	80.67
530	81.33	68.67	50.67	84	86	86	**87.33**	78.67
540	81.33	68.67	50.67	84.67	86	86.67	85.33	76.67
550	82	68.67	50.67	84.67	86.67	86	85.33	76.67
560	81.33	68.67	51.33	85.33	86.67	**87.33**	80.67	76
570	80.67	68.67	52	84	86	85.33	80	74.67
580	81.33	67.33	52	84.67	86	86	81.33	74.67
590	79.33	66.67	52	85.33	86	86	80	74.67
600	82	66.67	52	84	86.67	85.33	79.33	74.67
Avg	81.33	68.18	51.15	84.24	85.69	86.24	83.51	77.27

Berdasarkan Tabel 4.2, hasil pengujian diatas dapat dilihat bahwa nilai akurasi tertinggi pada kernel *polynomial* sebesar 82.67% dan nilai akurasi tertinggi pada kernel RBF sebesar 87.33%, yang artinya pengaruh pemilihan fitur menggunakan LSI berhasil memberikan akurasi tertinggi dan rata-rata akurasi yang lebih baik daripada skenario 1 untuk kedua kernel pada SVM. LSI dapat memberikan akurasi yang lebih baik karena LSI dapat menghilangkan beberapa *noise* dari data, yang dapat diterjemahkan sebagai *term* yang jarang digunakan maupun tidak penting[3]. Berdasarkan hasil yang didapatkan, dapat dilihat bahwa semakin kecil dimensi k yang digunakan, akurasi yang diperoleh untuk parameter *degree* dan *gamma* semakin meningkat. Pada dasarnya tidak ada cara khusus dalam menentukan nilai parameter k pada LSI sebagai dimensi, dan juga tidak ada pola khusus dari nilai parameter k sehingga tidak ada acuan dalam menentukan nilai k terkecil. Menentukan nilai k setelah didapat pada nilai k tersebut performansi terbaik, walapun mungkin pada nilai k yang lebih besar ditemukan hasil reduksi yang lebih baik terhadap performansi[19].

3. Skenario pengujian pengaruh *Bootstap Aggregating* (*Bagging*) pada klasifikasi *Support Vector Machine* (SVM)

Pada skenario pengujian pengaruh *Bootstap Aggregating* (*Bagging*) pada klasifikasi *Support Vector Machine* (SVM) bertujuan untuk membandingkan pengaruh metode *bagging* pada metode klasifikasi SVM, dan membandingkan nilai akurasi terhadap nilai jumlah *bag* yang digunakan pada metode *bagging*. Diperoleh hasil dengan hasil akurasi terbaik pada kernel *polynomial* sebesar 82% dengan jumlah bag sebesar 10, dan hasil akurasi terbaik pada kernel RBF sebesar 87.33% dengan jumlah bag sebesar 5,10 dan 15.

Tabel 4.3. Hasil pengujian skenario SVM-*Bagging*

n bag	Polynomial (d)			RBF (γ)				
	d =1	d =2	d =3	γ =0.01	γ =0.02	γ =0.03	γ =0.04	γ =0.05
5	81.33	58	58.67	86.67	**87.33**	76	70	68.67
10	**82**	58	56.67	**87.33**	86.67	74	70	66.67
15	81.33	60	59.33	86	**87.33**	73.33	70.67	67.33
20	80	59.33	58.67	86	86	74	71.33	68
Avg	81.165	58.83	58.33	86.5	86.83	74.33	70.5	67.66

Berdasarkan Tabel 4.3, dapat dilihat bahwa hasil akurasi terbaik adalah 82% pada kernel *polynomial*, dan hasil akurasi terbaik sebesar 87.33% pada kernel RBF, dapat disimpulkan bahwa pengaruh *bagging* berhasil memberikan akurasi tertinggi dan rata-rata akurasi yang lebih baik daripada skenario 1 untuk kedua kernel pada SVM, namun tidak lebih baik daripada skenario 2. *Bagging* dapat memberikan akurasi yang lebih baik karena pada dasarnya teknik *bagging* dengan *sampling with replacement* akan memberikan distribusi data *bag* dari setiap data *training* berbeda. Beberapa data *training* bisa muncul lebih dari sekali atau tidak muncul sama sekali. Hal ini yang menyebabkan *bagging* bisa meningkatkan akurasi karena dengan *sampling with replacement* dapat memperkecil *variance* atau *noise* dari *dataset* sehingga *bagging* dapat mengurangi kesalahan dalam melakukan klasifikasi.

4. Skenario Pengujian pengaruh *Bootstrap Aggregating* (*Bagging*) dan *Latent Semantic Indexing* (LSI) pada klasifikasi *Support Vector Machine* (SVM)

Pada skenario pengujian pengaruh *Bootstrap Aggregating* (*Bagging*) dan *Latent Semantic Indexing* (LSI) pada klasifikasi *Support Vector Machine* (SVM) bertujuan untuk membandingkan pengaruh metode *bagging* dan pemilihan fitur LSI terhadap klasifikasi SVM, dan membandingkan nilai akurasi terhadap nilai jumlah *bag* yang digunakan pada metode *bagging*. Pada pengujian ini menggunakan nilai *degree* dan *gamma* terbaik pada skenario 1. Diperoleh hasil akurasi terbaik pada kernel *polynomial* dengan nilai akurasi sebesar 84%, dan hasil akurasi terbaik pada kernel RBF dengan nilai akurasi sebesar 84.67%.

Tabel 4.4. Hasil pengujian skenario SVM-LSI-*Bagging*

k dimensi	Polynomial d=1				RBF γ=0.02			
	5	10	15	20	5	10	15	20
500	80	80.67	81.33	82	83.33	**84.67**	84	84
510	81.33	82	83.33	82	82.67	84	**84.67**	**84.67**
520	82.67	80.67	81.33	80.67	82	84	84	84
530	82.67	82	82	81.33	82.67	84	84	82.67
540	82	82	82	81.33	80.67	83.33	83.33	83.33
550	80	80.67	80.67	81.33	82	83.33	84	83.33

560	81.33	82	80	81.33	81.33	82	82	82.67
570	81.33	**84**	80.67	81.33	82.67	82	82.67	82.67
580	80	82.67	82	80.67	82	82.67	82.67	82.67
590	80	82	80.67	82	80.67	80.67	81.33	81.33
600	79.33	82.67	80	81.33	80.67	80	82	82
Avg	80.96	81.94	81.27	81.39	81.88	82.78	83.15	83.03
Avg Total	81.39				82.71			

Berdasarkan Tabel 4.4, dapat dilihat bahwa nilai akurasi terbaik pada kernel *polynomial* dengan nilai akurasi sebesar 84%. Hal ini membuktikan pengaruh pemanfaatan LSI dan *Bagging* pada SVM degan kernel *polynomial* berhasil memberikan nilai akurasi terbaik dari keempat skenario pengujian. Namun akurasi terbaik pada kernel RBF dengan nilai akurasi sebesar 84.67%. Hal ini membuktikan pengaruh pemanfaatan LSI dan *bagging* pada SVM dengan kernel RBF tidak lebih baik daripada sistem klasifikasi menggunakan SVM saja. Hal ini bisa terjadi diakibatkan kareakteristik *dataset* yang dapat memberikan performansi yang berbeda-beda pada setiap kernel.

5. Kesimpulan

Berdasarkan skenario pengujian yang sudah dilakukan pada penelitian ini, dapat disimpulkan bahwa pada metode klasifikasi *Support Vector Machine*. Nilai *degree* dan *gamma* yang tinggi tidak efektif pada dataset ini, karena nilai *degree* dan *gamma* yang besar yang menyebabkan *feature space* semakin tinggi. Hal ini menyebabkan algoritma kesulitan dalam menentukan *classifier* terbaik. Pemilihan fitur dengan *Latent Semantic Indexing* efektif untuk diimplementasikan dengan metode klasifikasi *Support Vector Machine* karena pada penelitian ini berhasil memberikan nilai akurasi yang lebih baik daripada tanpa pemilihan fitur, dan metode *Bagging* juga efektif untuk diimplementasikan dengan metode klasifikasi *Support Vector Machine* karena berhasil meningkatkan akurasi pada penelitian ini.

Ada saran yang bisa diterapkan untuk penelitian selanjutnya adalah seperti melakukan pelabelan yang dilakukan oleh orang yang lebih memahami topik hadits tersebut yang pada penelitian ini masih banyak label yang mungkin kurang sesuai. Dapat melakukan pengklasifikasian dengan *multi label* dikarenakan mungkin suatu hadits tidak hanya mengandung informasi saja, namun juga adanya makna yang ingin disampaikan seperti anjuran dan larangan tertentu.

Daftar Pustaka

[1] Marwan Hakim, Sistem Pakar Menidentifikasi Hadits Menggunakan Menggunakan Metode Forward Chaining, 2016.
[2] Wulandini F & Nugroho A.N, Text Classification Using Support Vector Machine for Web Mining Based Spation Temporal Analuysis of the Spread of Tropical Diseases, International Conference on Rural Information and Communication Technology 2009, 2009.
[3] Ana Cardoso-Cachopom dan Arlindo I., Olivera, Combining LSI with other Classifiers to Improve Accuracy of Single-lanel Text, 2007.
[4] Durgesh, K.S. and Lekha, B., Data classification using support vector machine. Journal of Theoretical and Applied Information Technology, 12(1), pp.1-7. 2010.
[5] Igg Adiwijaya Ph.D., Text Mining and Knowledge Discovery,2006.
[6] Christianini, Nello and John S. Taylor., An Introduction to Support Vector Machines and Other Kernel-based Learning Merhods. Cambridge University Press, 2000.
[7] Anto Satriyo Nugroho, Arief Budi Witarti dan Dwi Handoko, Support Vector Machine – Teori dan aplikasisnya dalam bioinformatika, 2003.
[8] Aditiya Pratama. Faisal Ajang Sidik dan Yuliani Dwi Asih, Penerapan Teknik Bagging Pada Algoritma Klasifikasi C4.5 Untuk Mendeteksi Penyakit Liver, 2017.
[9] K. Shima, M. Todoriki dan A. Suzuki, SVM-based feature selection of latent semanctic features, 2003.

[10] Hai jin, Xiaoming Ning, Hanhua dan Zuoning Yin, Efficient Query Routing for Information Retrieval in Semantic Overlays, 2006.
[11] Ziqiang Wang dan Dexian Zhang, Feature Selection in Text Classification Via SVM and LSI, 2006.
[12] Khrisna Dini Yunita Sari, Kategorisasi teks dengan metode klasifikasi Support Vector Machine (SVM)i, 2006.
[13] Achmad Ridok, Indriati., Pengklasifikasian Dokumen Berbahasa Indonesia Dengan Penindeksan Berbasis LSI , 2015.
[14] Adiwijaya, Aplikasi Matriks Ruang Vektor, Graha Ilmu (Yogyakarta), 2014.
[15] Hendra Bunyamin, Information Retrieval system Dengan Metode Latent Semantic Indexing, 2009.
[16] Yang Y., & Liu X. A re-examination of text categorization methods In Proceedings 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR-99), 1999.
[17] Ramana, B. V., Babu, S. p ., dan Venkateswarlu, N. B., A Critical Study Of Selected Classification Algorithms For Live Disease Diagnosis. International Journal Of Database Management Systems, Vol. 3(2), 2011.
[18] S. Prangga, Optimasi Parameter pada Support Vector Machine menggunakan Pendekatan Metode Taguchi untuk Data High-Dimensional, pp. 33-37, 2017.
[19] Deerwester, Scott et al, Indexing by Latent Semantic Indexing, Journal of The American Society for Information Science., 1990.
[20] Billy Eden William Asrul, Bagging Support Vector Machines for leukemia Classification., 2014.
[21] Bian, Rian., Pengertian Latent Semantic Indexing, 2010.
[22] A. S. Nugroho, A. B. Witarto dan D. Handoko, Support Vector Machine (Teori dan Aplikasinya dalam Bioinformatika), 2003.

Lampiran