Semantic Enrichment of Text Representation with Wikipedia for Text Classification

Hiroki Yamakawa, Jing Peng, and Anna Feldman
Dept. of Computer Science
Montclair State University
Montclair 07043, USA
yamakawah1@mail.montclair.edu; pengj@mail.montclair.edu; feldmana@mail.montclair.edu

Abstract—**Text classification is a widely studied topic in the area of machine learning. A number of techniques have been developed to represent and classify text documents. Most of the techniques try to achieve good classification performance while taking a document only by its words (e.g. statistical analysis on word frequency and distribution patterns). One of the recent trends in text classification research is to incorporate more semantic interpretation in text classification, especially by using Wikipedia. This paper introduces a technique for incorporating the vast amount of human knowledge accumulated in Wikipedia into text representation and classification. The aim is to improve classification performance by transforming general terms into a set of related concepts grouped around semantic themes. In order to achieve this goal, this paper proposes a unique method for breaking the enormous amount of extracted Wikipedia knowledge (concepts) into smaller pieces (subsets of concepts). The subsets of concepts are separately used to represent the same set of documents in a number of different ways, from which an ensemble of classifiers is built. Experimental results show that an ensemble of classifiers individually trained on a different representation of the document set performs better with increased accuracy and stability than that of a classifier trained only on the original document set.**

Keywords—**text classification, text representation, semantics, ensemble, voting, Wikipedia**

I. Introduction

As the number of online text documents increases, the demand for automated text classification that can be performed by computers grows, especially in the field of information retrieval (IR) and online search of documents. Many studies have therefore been conducted in order to achieve better classification performances.

The basic method for representing a text document is to use a column vector whose rows represent the words contained in the document [1]. When dealing with a set of documents, document vectors are combined to create a term-document (TD) matrix (1), where $x_{(i,j)}$ denotes the weight (usually term frequency, TF) for i_{th} term (t_i) in j_{th} document (d_j) :

$$TD = X = \begin{bmatrix} x_{(1,1)} & x_{(1,2)} & \cdots & x_{(1,n)} \\ x_{(2,1)} & x_{(2,2)} & \cdots & x_{(2,n)} \\ \vdots & \vdots & \ddots & \vdots \\ x_{(m,1)} & x_{(m,2)} & \cdots & x_{(m,n)} \end{bmatrix} \quad (1)$$

The above so-called "bag-of-words" (BOW) approach with TF, or other weight adjustment measures such as term frequency - inverse document frequency (TF-IDF), faces a number of problems that stem from its underlying assumption (e.g. the independence among the terms in the document set) [2][3]. Latent Semantic Analysis (LSA) is developed to address the problems that arise from such an assumption [4]. In LSA, the terms in TD are replaced by "latent semantic concepts" that form the basis of latent semantic space. Latent semantic concepts, however, are difficult to interpret, since they do not convey explicit word senses as the original terms do. In addition, since it is purely statistical, LSA may not be able to detect true semantic relatedness among terms or documents when the document corpus fails to provide enough instances of co-occurrences that indicate the semantic relatedness.

Recently, several techniques in text representation have been developed that focus on extracting true semantic concepts from text documents [2]. Here, true semantic concepts mean those concepts that are explicitly defined and understood by humans in general. One example of true concept extraction is the use of Wikipedia (an online encyclopedia) [5] .

This paper presents a unique approach to representing text documents so as to improve the performance of text classification. Based on the assumption that adding more semantic attributes to plain text representation leads to better text classification performances, the concepts described in Wikipedia are incorporated in the process of creating semantically rich text representations. Furthermore, our technique systematically transforms the enormous amount of extracted Wikipedia knowledge (concepts) into smaller pieces (subsets of concepts). The subsets of concepts are separately used to represent the same set of documents in a number of different ways, from which an ensemble of classifiers is built. Experimental results show that an ensemble of classifiers individually trained on a different representation of the document set performs better with increased accuracy and stability than that of a classifier trained only on the original document set.

II. Text Representation Using Wikipedia Concepts

A. *Wikipedia Profile*

Wikipedia is a free online encyclopedia. It is the largest encyclopedia available online and offline [5]. Its contents are contributed, edited, and updated by hundreds of thousands of active registered users every month. It contains over three million English articles and about 13 million articles in all languages. For a comparison, Britannica Online has over 122,264 articles as of August 1, 2009 [6].

One of the most peculiar characteristics of Wikipedia is that it contains a great number of proper nouns, domain-specific technical terms, and many topics that are usually not covered in other encyclopedias. For example, Wikipedia can tell that the term "A330", which is found in RCV1 [7] (one of the datasets used in this study to conduct experiments), is a passenger airliner built by Airbus and provide much more information about the aircraft [8]. Another characteristic of Wikipedia is the timeliness of its coverage; articles describing new scientific discoveries, current events, etc. quickly show up in Wikipedia. It is also constantly growing and evolving, as thousands of people post, edit, revise, and update the contents of all kinds of topics imaginable on a daily basis [5].

Wikipedia can also boost the amount of information contained in short documents/sentences by providing a number of associated words [2]. For example, a short sentence, such as "Obama meets Sarkozy to discuss subprime crisis," cannot adequately be represented by BOW approaches that only consider the words contained in the sentence. If Wikipedia were used to augment the sentence with such words (concepts) as President, the United States of America, France, politician, financial markets, subprime mortgage crisis, lending, etc., the sentence would contain much more information that might be very helpful in text classification.

B. Text Representation Using Wikipedia Concepts

1) Wikipedia Concepts Extraction: A regular TD matrix is first created before retrieving Wikipedia concepts for each of the terms in the TD matrix. In this paper, Wikipedia Thesaurus [9] is used to retrieve associated Wikipedia concepts. Wikipedia Thesaurus returns top candidate concepts that may closely describe a given query word and 30 associated Wikipedia concepts for each of those candidate concepts. It also produces "relatedness scores" of the associated Wikipedia concepts returned, indicating the degree to which the returned Wikipedia concepts are associated with the candidate words chosen for the query word.

Wikipedia Thesaurus calculates relatedness scores by examining the link structures of Wikipedia articles. A relatedness score has the following properties: A relatedness score between two articles (concepts) is high if there are many paths (which consist of direct/indirect hyper links) between them [10].

Path Frequency – Inversed Backward Link Frequency (pfibf) [10] is a measure developed to determine relatedness scores between two articles. It is expressed as follows: given two articles (concepts) v_i and v_j and a set of all paths from article v_i to v_j (using T to represent a set of all n paths from article v_i to v_j: $T = \{t_1, t_2, \ldots, t_n\}$), the relatedness score (pfibf) between the two articles is: $pfibf(v_i,v_j) = pf(v_i,v_j) \cdot ibf(v_j)$, where pf represents path frequency

$$pf(v_i, v_j) = \sum_{k=1}^{n} \frac{1}{d(|t_k|)} \qquad (2)$$

and $d(|t_k|)$ denotes a function that monotonically increases with increasing length of path t_k (e.g. logarithmic function). Here *ibf* represents inversed backward link frequency:

$$ibf(v_j) = \log \frac{N}{bf(v_j)} \qquad (3)$$

N is the total number of articles, and $bf(v_j)$ is the number of backward links of v_j (links pointing to v_j).

2) Enhancing TD with Extracted Wikipedia Concepts: After Wikipedia concepts are extracted, the next step is to update the TD matrix. In this paper, the TD matrix is updated in two different ways: the augmentation of the TD matrix with Wikipedia concepts, and the projection of the TD matrix in the Wikipedia concept space.

TD Matrix Augmented with Wikipedia Concepts: In the augmentation method, the dimension of a TD matrix (i.e. the total number of terms/features) is expanded by the extracted Wikipedia concepts. If $r_{(i,c)}$ denotes the relatedness score between a term t_i and a corresponding Wikipedia concept w_c (w_c: $c = 1, 2, 3, \ldots, m'$: m' is the number of Wikipedia concepts extracted: $m \le m'$ or $m > m'$), and $X((m + c), :)$ denotes the row vector for the Wikipedia concept w_c, then $X((m + c), :)$ is expressed as: (n is the number of documents)

$$X((m+c),:) = [x_{(m+c,1)}, x_{(m+c,2)}, x_{(m+c,3)}, \ldots, x_{(m+c,n)}]$$

$$= r_{(i,c)} \times [x_{(i,1)}, x_{(i,2)}, x_{(i,3)}, \ldots, x_{(i,n)}] \qquad (4)$$

That is, a vector of each Wikipedia concept is a product of the relatedness score ($r_{(i,c)}$ between t_i and w_c) and the vector of the original term t_i ($=[x_{(i,1)}, x_{(i,2)}, x_{(i,3)}, \ldots, x_{(i,n)}]$).

All the new vectors for Wikipedia concepts are vertically concatenated to form a WikiConcept-document (WD) matrix

$$WD = X' = \begin{bmatrix} x_{(1,1)} & x_{(1,2)} & \cdots & x_{(1,n)} \\ x_{(2,1)} & x_{(2,2)} & \cdots & x_{(2,n)} \\ \vdots & \vdots & \ddots & \vdots \\ x_{(m',1)} & x_{(m',2)} & \cdots & x_{(m',n)} \end{bmatrix} \qquad (5)$$

of size m' by n. The TD matrix and the WD matrix are vertically concatenated to form a TD&WD matrix, which is expressed as:

$$TD\&WD = X_{aug} = \begin{bmatrix} TD \\ WD \end{bmatrix}$$

$$= \begin{bmatrix} x_{(1,1)} & x_{(1,2)} & \cdots & x_{(1,n)} \\ x_{(2,1)} & x_{(2,2)} & \cdots & x_{(2,n)} \\ \vdots & \vdots & \ddots & \vdots \\ x_{(m,1)} & x_{(m,2)} & \cdots & x_{(m,n)} \\ x_{(m+1,1)} & x_{(m+1,2)} & \cdots & x_{(m+1,n)} \\ x_{(m+2,1)} & x_{(m+2,2)} & \cdots & x_{(m+2,n)} \\ \vdots & \vdots & \ddots & \vdots \\ x_{(m+m',1)} & x_{(m+m',2)} & \cdots & x_{(m+m',n)} \end{bmatrix} \qquad (6)$$

While using WD alone without TD as a new text representation may sound reasonable, our investigation and others [2][11] have found the importance of keeping the TD matrix when adding semantic concepts from external knowledge, such as Wikipedia. Therefore, the WD matrix is simply attached at the bottom of the TD matrix in this work.

The degree to which the attached WD matrix exerts influence in the new representation of a TD&WD matrix can be adjusted by multiplying WD with a scalar variable, α. This value can be adjusted to suit the nature of a document corpus. For the Reuters dataset used in this paper, α = 1 gives rise to better performance.

Projection of TD in WikiConcept Space Using WikiConcept-To-Term Matrixs: A WikiConcept-to-Term (WT) matrix represents a set of relatedness scores among WikiConcepts and the original terms. It is constructed and used as a factor to project the TD matrix in the Wikipedia concept space. Different types of WT matrices are examined bellow.

One-to-One Correspondence WT Matrix

In this WT matrix, each WikiConcept corresponds to only one of the original terms, and all the correspondences are unique. That is, two or more WikiConcepts never relate to the same term, and two or more original terms never relate to the same WikiConcept.

This results in a creation of a single matrix with the dimension expressed as: #WikiCons × #Terms, where #WikiCons = #Terms (here, #WikiCons denotes the number of Wikipedia concepts extracted. #Terms represent the number of the original terms, which never changes in all WT types).

The one-to-one correspondence WT ($WT_{1:1}$) matrix is expressed as:

$$WT_{1:1} = R = \begin{bmatrix} r_{(1,1)} & r_{(1,2)} & \cdots & r_{(1,m)} \\ r_{(2,1)} & r_{(2,2)} & \cdots & r_{(2,m)} \\ \vdots & \vdots & \ddots & \vdots \\ r_{(m',1)} & r_{(m',2)} & \cdots & r_{(m',m)} \end{bmatrix} \quad (7)$$

where $r_{(c,i)}$ represents the relatedness score between WikiConcept w_c (c = 1, 2, 3, … , m') and term t_i (i = 1, 2, 3, … , m). Note that $m = m'$ in this case.

The advantage of a $WT_{1:1}$ matrix is that after projecting a TD matrix in the WikiConcept space, the dimension of the resulting WD matrix is the same as that of the original TD matrix. The disadvantages are that this type of WT matrix fails to capture inter-relationship among original terms. It also fails to capture inter-relationship among WikiConcepts.

One-to-Many Relational WT Matrix

In this WT matrix, a WikiConcept may relate to one or more original terms, but each original term can relate to at most one WikiConcept. This results in a single matrix with the dimension expressed as #WikiCons × #Terms, where #WikiCons ≤ #Terms (note that #Terms is always the same; even if no related WikiConcept is found for some terms, those terms are still kept as column vectors with all the elements set to 0.).

The one-to-many relational WT ($WT_{1:m}$) matrix is expressed as:

$$WT_{1:m} = R = \begin{bmatrix} r_{(1,1)} & r_{(1,2)} & \cdots & r_{(1,m)} \\ r_{(2,1)} & r_{(2,2)} & \cdots & r_{(2,m)} \\ \vdots & \vdots & \ddots & \vdots \\ r_{(m',1)} & r_{(m',2)} & \cdots & r_{(m',m)} \end{bmatrix} \quad (8)$$

where $m \geq m'$.

The advantages of this WT are: (1) it captures inter-relationship among original terms via the shared WikiCondepts; and (2) it is possible to keep the dimension of the projected matrix WD the same as that of the original TD ($m \geq m'$). The disadvantage of this WT matrix is that it fails to capture inter-relationship among WikiConcepts.

Many-to-One Relational WT Matrix

In this WT matrix, one or more WikiConcepts may relate to the same original term. However, each WikiConcept can relate to at most one original term. This results in a single matrix with the dimension expressed as #WikiCons × #Terms, where it is most likely that #WikiCons ≥ #Terms. It is possible that #WikiCons < #Terms if related WikiConcepts are found for very few original terms. #Terms is always the same as discribed in the $WT_{1:1}$ matrix section above.

The many-to-one relational WT ($WT_{m:1}$) matrix is expressed as:

$$WT_{m:1} = R = \begin{bmatrix} r_{(1,1)} & r_{(1,2)} & \cdots & r_{(1,m)} \\ r_{(2,1)} & r_{(2,2)} & \cdots & r_{(2,m)} \\ \vdots & \vdots & \ddots & \vdots \\ r_{(m',1)} & r_{(m',2)} & \cdots & r_{(m',m)} \end{bmatrix} \quad (9)$$

where usually $m \leq m'$.

The advantage of $WT_{m:1}$ is that it can capture inter-relationship among WikiConcepts via the shared terms. The disadvantages are that this WT matrix fails to capture inter-relationship among original terms, and the dimension of the projected matrix WD most likely changes upward as the number of WikiConcepts extracted tends to far exceed that of the original terms.

Many-to-Many Relational WT Matrix

In this WT matrix, one or more WikiConcepts may relate to the same original term, and one or more original terms may relate to the same WikiConcept. Most of the time, the extraction of WikiConcepts from a set of original terms results in this type of relation (WT matrix). The dimension of the many-to-many relational WT matrix ($WT_{m:m}$) is expressed as: #WikiCons × #Terms, where it is most likely that #WikiCons ≥ #Terms. However, it can be that #WikiCons < #Terms if related WikiConcepts are found for very few original terms. Again, #Terms is always the same as discribed above.

The $WT_{m:m}$ matrix is expressed as:

$$WT_{m:m} = R = \begin{bmatrix} r_{(1,1)} & r_{(1,2)} & \cdots & r_{(1,m)} \\ r_{(2,1)} & r_{(2,2)} & \cdots & r_{(2,m)} \\ \vdots & \vdots & \ddots & \vdots \\ r_{(m',1)} & r_{(m',2)} & \cdots & r_{(m',m)} \end{bmatrix} \quad (10)$$

where $m \leq m'$ in general.

The advantages are that it can capture the inter-relationship among WikiConcepts via the shared terms and the inter-relationship among original terms via the shared WikiConcepts. The disadvantage is that the dimension of the projected matrix WD most likely changes significantly upward as the number of WikiConcepts extracted tends to far exceed that of the original terms.

The ratio of the total number of WikiConcepts extracted to the total number of the original terms usually exceeds 10/1. If this huge $WT_{m:m}$ is used to project the TD matrix in the WikiConcept space, the size of the resulting WD matrix most likely becomes too big for machine computation. In order to avoid this issue, some strategies to transform it into several smaller, yet effective, WT matrices are explored next.

Breakdown of Many-to-Many Relational WT Matrix

A many-to-many relational WT ($WT_{m:m}$) matrix can be broken down into several one-to-many relational WT ($WT_{1:m}$) matrices. Each $WT_{1:m}$ matrix represents the relationship among a subset of Wikipedia concepts and the set of the entire original term dictionary. For any $WT_{1:m}$, each original term relates to at most one of the WikiConcepts in the subset.

If there is no relation between one original term and any of the WikiConcepts in the subset, a value, 0, is entered as the element for that relation in the matrix.

This breaking down process keeps the dimension of each $WT_{1:m}$ matrix to: #WikiConcs × #Terms; where #WikiCons ≤ #Terms. Again, #Terms is always the same as discribed earlier in the $WT_{1:1}$ matrix section.

Each one of the one-to-many relational matrices created from the $WT_{m:m}$ matrix is expressed as:

$$e^{th}WT_{1:m} = R_e = \begin{bmatrix} r_{e(1,1)} & r_{e(1,2)} & \cdots & r_{e(1,m)} \\ r_{e(2,1)} & r_{e(2,2)} & \cdots & r_{e(2,m)} \\ \vdots & \vdots & \ddots & \vdots \\ r_{e(m_e,1)} & r_{e(m_e,2)} & \cdots & r_{e(m_e,m)} \end{bmatrix} \quad (11)$$

where $r_{e(c,i)}$ represents relatedness score between WikiConcept w_c (w_c: $c = 1, 2, 3, \ldots, m_e$) and the original term t_i (t_i: $i = 1, 2, 3, \ldots, m$) in e_{th} $WT_{m:m}$ matrix, and $m \geq m_{e.}$

If #WikiCons < #Terms (that is, $m > m_e$), using the $WT_{1:m}$ matrix to project the TD matrix in the WikiConcept space results in a WD matrix with lower dimension than that of the TD matrix. For computational flexibility (e.g. when merging all the projected WD matrices (and the TD matrix) by summation or taking the average), one may choose to set #WikiCons = #Terms all the time for all $WT_{1:m}$ matrices and rearrange the order of WikiConcepts so that the order corresponds with that of the original terms. The rearragnement ensures the followings: a) The order in which the WikiConcepts are arranged vertically in $WT_{1:m}$ matrices is the same as the order in which the original terms are arranged horizontally in $WT_{1:m}$ matrices. That is, a WikiConcept that relates to the first original term is placed in the first row. A WikiConcept that relates to the second original term is placed in the second row, and so on. b) If i_{th} original term does not have any related WikiConcept extracted, i_{th} row of the $WT_{1:m}$ matrix is set to be a zero vector (as an empty WikiConcept).

Since each $WT_{1:m}$ matrix is a one-to-many relational matrix, the same WikiConcept can relate to more than one original term. In such a case, more than one row represents the same WikiConcept, and each of the rows contains more than one non-zero element. The elements (relatedness scores) are divided by the total number of relations the WikiConcept has with original terms.

For example, suppose the 3rd WikiConcept (w_3) has a relatedness score of 0.4 with the third term and a score of 0.6 with the fourth term:

$$1stWT_{1:m}(3,:) = [0.0 \quad 0.0 \quad 0.4 \quad 0.6]$$

If the first WikiConcept has a relatedness score of 0.5 with the first term:

$$1stWT_{1:m}(1,:) = [0.5 \quad 0.0 \quad 0.0 \quad 0.0]$$

and the second WikiConcept has a relatedness score of 0.7 with the second term:

$$1stWT_{1:m}(2,:) = [0.0 \quad 0.7 \quad 0.0 \quad 0.0]$$

then $1stWT_{1:m}$ matrix is expressed as:

$$1stWT_{1:m} = \begin{bmatrix} 0.5 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.7 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.4 & 0.6 \end{bmatrix}$$

After the rearrangemtn performed to set $m = m_e$, the third row and the fourth row of $WT_{1:m}$ matrix are expressed as:

$$WT_{1:m}(3,:) = [0.0 \quad 0.0 \quad 0.2 \quad 0.3]$$
$$WT_{1:m}(4,:) = [0.0 \quad 0.0 \quad 0.2 \quad 0.3]$$

then the rearranged $1stWT_{1:m}$ matrix is expressed as:

$$1stWT_{1:m} = \begin{bmatrix} 0.5 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.7 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.2 & 0.3 \\ 0.0 & 0.0 & 0.2 & 0.3 \end{bmatrix}$$

where $m = m_e = 4$. Now, the dimension of each $WT_{1:m}$ matrix is expressed as: #WikiConcs x #Terms; where #WikiCons = #Terms. Note that #Terms is always the same as described ealier. Also, #WikiCons is always the same; even if no WikiConcept is extracted for an original term, an empty WikiConept is filled in for the term as a row vector with all the elements set to 0.

When projecting the TD matrix in the space of Wikipedia concepts, this makes it possible to keep the dimension of each of the projected WD matrices the same as that of the original TD, and the correspondence between the order of the original terms and the order of the WikiConcepts is maintained.

One benefit of transforming the huge $WT_{m:m}$ matrix into much smaller $WT_{1:m}$ matrices is that the computation of smaller separate WD matrices can easily be distributed among multiple processors/computers to significantly reduce the time required for classification.

Algorithm for Creating Several One-to-Many Relational WT Matrices

Step 1: From Wikipedia Thesaurus, extract Wikipedia concepts and their relatedness scores for all the terms contained in the training document set. In this study, the relatedness scores for top candidate WikiConcepts are set to be twice as much as the highest relatedness score extracted from each of those top candidate concepts.

Step 2: Normalize each set of relatedness scores returned for top candidate concepts, so that the top candidate concept receives a relatedness score of 1.0, the highest associated concept receives 0.5 (as a result of Step 1), and the rest of the associated concepts receive proportionally adjusted scores.

Assign a progressively smaller weight to the top candidate concept and its associated concepts in the order they are returned by Wikipedia Thesaurus. That is, the first top candidate concept returned and its associated concepts are weighted more than the second top candidate concept and its associated concepts are (for example, weight = 1.0 for the first top candidate and its associated concepts, weight = 0.9 for the second top candidate set, and so on). This weight is used to reflect the tendency of Wikipedia Thesaurus to return more appropriate top candidates first and less appropriate ones last.

Step 3: Arrange WikiConcepts based on the relatedness scores from the highest to the lowest. Many WikiConcepts are extracted multiple times for different terms. Combine all the scores for the same Wikipedia concepts by summation. Once it is done, Wikipedia concepts are arranged in such a way that most relevant WikiConcepts in the context of the entire document corpus are placed in the top part of the list, and the least relevant concepts are placed at the bottom of the list.

Step 4: Create $WT_{1:m}$ matrices one at a time. Each row in the $1^{st}WT_{1:m}$ matrix consists of the one-to-many relation among one Wikipedia concept and either zero (in case of an empty WikiConcept to maintain the dimention), one, or more original terms that correspond with the Wikipedia concept. Every one-to-many relation of each row is derived so that the selected Wikipedia concept for the $1^{st}WT_{1:m}$ matrix has "the highest" relatedness score for the corresponding original terms. The dimension and the order of the WikiConcepts are arranged to correspond with those of the original terms.

Each row in the $2^{nd}WT_{1:m}$ matrix consists of the one-to-many relation among one Wikipedia concept and either zero, one, or more original terms that correspond with the Wikipedia concept. Every one-to-many relation of each row is derived so that the selected Wikipedia concept for the $2^{nd}WT_{1:m}$ matrix has "the second highest" relatedness score for the corresponding original terms. The dimension and the order of the WikiConcepts are arranged to correspond with those of the original terms.

$3^{rd}WT_{1:m}$, $4^{th}WT_{1:m}$, $5^{th}WT_{1:m}$, and all the subsequent $WT_{1:m}$ matrices are created in the same way.

<u>Multiple WD Creation</u>

Once the $WT_{m:m}$ matrix is broken down into several $WT_{1:m}$ matrices, the next step is to project the TD matrix in the extracted WikiConcept space. The projection is performed by left-multiplying the original TD matrix by each of the $WT_{1:m}$ matrices.

$$WD_e = e^{th}WT_{1:m} \times TD$$
$$= \begin{bmatrix} r_{e(1,1)} & r_{e(1,2)} & \cdots & r_{e(1,m)} \\ r_{e(2,1)} & r_{e(2,2)} & \cdots & r_{e(2,m)} \\ \vdots & \vdots & \ddots & \vdots \\ r_{e(m_e,1)} & r_{e(m_e,2)} & \cdots & r_{e(m_e,m)} \end{bmatrix} \begin{bmatrix} x_{(1,1)} & x_{(1,2)} & \cdots & x_{(1,n)} \\ x_{(2,1)} & x_{(2,2)} & \cdots & x_{(2,n)} \\ \vdots & \vdots & \ddots & \vdots \\ x_{(m,1)} & x_{(m,2)} & \cdots & x_{(m,n)} \end{bmatrix} \tag{12}$$

$$WD_e = X_e = \begin{bmatrix} x_{e(1,1)} & x_{e(1,2)} & \cdots & x_{e(1,n)} \\ x_{e(2,1)} & x_{e(2,2)} & \cdots & x_{e(2,n)} \\ \vdots & \vdots & \ddots & \vdots \\ x_{e(m_e,1)} & x_{e(m_e,2)} & \cdots & x_{e(m_e,n)} \end{bmatrix} \tag{13}$$

where $m = m_e$

$$x_{e(i,j)} = \begin{bmatrix} r_{e(i,1)} & r_{e(i,2)} & \cdots & r_{e(i,m)} \end{bmatrix} \begin{bmatrix} x_{(1,j)} \\ x_{(2,j)} \\ \vdots \\ x_{(m,j)} \end{bmatrix} \tag{14}$$

For each $WT_{1:m}$ matrix to retain stronger aspects of the original TD matrix, one can choose to combine each $WT_{1:m}$ matrix and the TD matrix by addition or taking their mean values. In this paper, the combination by summation is applied to each $WT_{1:m}$ matrix.

III. Text Classification with Wikipedia-Enriched Text Representation

A. *Classification with TD Matrix Augmented with Wikipedia Concepts*

Classification with a TD matrix augmented with WikiConcepts basically replaces the TD matrix with the TD&WD matrix. The same classifier models used with the TD matrix (e.g. kNN and SVMs) can be used with the TD&WD matrix in the same way. The degree to which the WD part of the matrix exerts influence can be adjusted by changing the value of the scalar variable α that sets the weight of WD.

The advantage of using this text representation is that there is no need to change the classification approaches used with the regular TD matrices. The disadvantage is that the size of a TD&WD matrix may become too big for computation.

B. *Classification with Multiple WikiConcept-To-Document Matrices*

For each one of the WD matrices, the same classifier models used with the TD matrix (e.g. kNN and SVMs) are used. To take advantage of the multiple classifiers running on multiple WD matrices, some applications of voting techniques are explored in the following.

1) Classification Approaches with Multiple Classification: In the procedure described in the previous section, multiple WD matrices are generated from the original TD matrix. Each WD matrix is a projection of the TD matrix in the space defined by a subset of Wikipedia concepts. Each subset of WikiConcepts is different from any of the others, as they are extracted in the order from the highest relatedness score down for $1^{st}WT$, $2^{nd}WT$, $3^{rd}WT$, and so on (Although each of the subsets are different from the others, some WikiConcepts may appear in more than one of those subsets for different terms; for example, one WikiConcept may be the concept with the highest relatedness score for one term, and the same WikiConcept may be the concept with the second highest relatedness score for another term).

The next step is to train classifiers on those WD matrices. One unique classifier is trained on each WD matrix, and yet another unique classifier is trained on the TD matrix. All of these classifiers are separately run on the same test documents, which results in an ensemble of predictions. There are a number of different strategies developed to make use of an ensemble of predictions. This study focuses on voting techniques to make final predictions given the ensemble.

2) Ensemble of Classifiers: Voting is usually conducted when an ensemble of classifiers produces multiple predictions for a given instance. There are a number of different reasons to build an ensemble of classifiers in the first place: for statistical reason; for effectively handling large volumes of data; for effectively handling small volumes of data; for applying "divide-and-conquer" methods; and for data fusion [12].

The statistical reason for creating an ensemble of classifiers is that classifiers that perform similarly at a training stage may perform differently from each other when testing on a new set of data. Having an ensemble of classifiers and taking vote or average from the multiple predictions helps avoid selecting only one classifier that turns out to be a poor predictor [12].

When the amount of data is so huge that it takes too much time for any classifier to deal with, the data may be divided into smaller subsets and an ensemble of classifiers (one unique classifier for each subset) may be trained on them. The multiple predictions made by the ensemble are combined into a single prediction by voting or other combination techniques [12].

In this study, the above two are the main reasons for adopting the ensemble strategy.

Voting Techniques: The two voting techniques usually adopted for ensemble approaches are plurality voting and majority voting [13]. This study uses only the plurality voting. The majority voting was initially included in the set of techniques used, but for this study, the plurality voting performed consistently better than the majority voting. The better performance of plurality voting over majority voting may be attributable to the rule of majority voting adopted. In this study, the majority voting was implemented so that if no prediction receives the "majority" of votes, the prediction made by the classifier trained only on TD matrix was used as the final prediction. The assumption made with this implementation was that the probability of outputting correct predictions by the classifier trained only on TD matrix is higher than that of the rest. The experiments (in the next section) show that that is not always the case.

A study conducted by [14] investigates in detail some advantages of plurality voting over majority voting. It calls the advantage "higher rejection efficiency." Simply put, fewer rejections made by plurality voting lead to a better reliability. Whether this advantage applies to this study is a question that requires further analysis in the future.

In general, for any voting techniques to perform well, the diversity of classifier performances (prediction accuracy) and/or the diversity of the training data sets separately used for each of the classifiers are important [12].

3) Procedures for Classification with Multiple WD Matrices:

<u>Training</u>

Step 0: Train the classifier (CL_0) on TD matrix

(This step is taken if the information represented in TD matrix is too important to be discarded.)

Step 1: Extract *k* WT matrices from a training data set and create the same number of WDs by multiplying TD matrix by each WT matrix. There are now *k* separate training data sets.

TrainingD = {1stWD, 2ndWD, … , *k*thWD}

Step 2: Divide each of the training data sets into *k* subsets (*k*-fold data split)

1stWD = {Subset(1stWD,1), Subset(1stWD,2), … , Subset(1stWD,*k*)}
2ndWD = {Subset(2ndWD,1), Subset(2ndWD,2), … , Subset(2ndWD,*k*)}
and so on…

Step 3: Train the first classifier (CL_1) on 1stWD matrix minus the first subset,

$CL_1 \rightarrow$ {1stWD - Subset(1stWD,1)}

Train the second classifier (CL_2) on 2ndWD matrix minus the second subset.

$CL_2 \rightarrow$ {2ndWD - Subset(2ndWD,2)}

And so on…

This step ensures that each classifier is trained on slightly different data set (different combination of training documents).

At the end of this step, the ensemble of classifiers is created.

$E = \{CL_1, CL_2, \dots, CL_k\}$

If Step 0 is taken, then the ensemble also includes CL_0.

$E = \{CL_0, CL_1, CL_2, \dots, CL_k\}$

<u>Testing</u>

Step 1: Run each classifier in the ensemble on the testing set of documents and pool the results (predictions) for each instance *i*.

$PredE_i = \{(pred_0), pred_1, pred_2, \dots pred_k\}$

where $i = 1, 2, \dots, n$ (n = number of documents)

Step 2: For each instance *i*, count the number of each prediction made and choose the prediction with the highest count. If more than one prediction has the highest count, break the tie by choosing the one made by a higher set of classifiers. The first classifier (CL_1) is higher than the second classifier (CL_2), the third classifier (CL_3), and so on. The second classifier (CL_2) is higher than the third classifier (CL_3), the fourth classifier (CL_4), and so on. The classifier CL_0 is higher than all the others. One set of classifier is higher than another, if it has the highest classifier among all the classifiers contained in both sets.

IV. Experiments

A. Dataset and Pre-Processing

Three well-know datasets are used for the experiments: Reuters – 21578 [15], RCV1 [7], and 20 Newsgroups (20NG) [16]. They have been used by many studies involving text classification.

To conduct the experiments, the documents need to be processed first: a) Documents with only one category are selected for the training and testing sets. Documents that belong to multiple categories are discarded for simplicity of implementation/experimentation: b) Words are lemmatized for Reuters and 20NG datasets. (Dragon Toolkit [17] was used.): c) Words are stemmed for RCV1 dataset. (Only the stemmed texts are readily available for the dataset, and no lemmatization from the original documents could be performed.): d) Only nouns and verbs are retained after the lemmatization for Reuters and 20NG (using Dragon Toolkit [17]): e) For Reuters, only documents that belong to one of the 5 categories are selected. (acq, crude, earn, grain, trade) [15]. The entire document set contains 2000 documents (400 for each category). For training, 100 documents (out of 400) are selected for each category. For testing, 100 documents are randomly selected each time for each category with replacement from the 400 documents that belong to the same category: f) For RCV1, the categories are grouped into 10 root categories (I0, I1, I2, I3, I4, I5, I6, I7, I8, I9) [7]. For training, 100 documents are selected for each category (1000 documents in total). For testing, 100 documents are randomly selected each time for each category with replacement from 500 documents that belong to the same category: g) For 20NG, all 20 categories are used as is. For training, 100 documents are selected for each category (2000 documents in total). For testing, 100 documents are randomly selected each time for each category with replacement from 350 documents that belong to the same category: h) Numbers, stop words, words with fewer than 3 characters or more than 30 characters are removed: i) Words that do not separately appear in three or more documents in the document set are removed.

B. *Wikipedia Concept Extraction Procedures*

The procedures to extract Wikipedia concepts from Wikipedia Thesaurus are as follows: a) The number of top candidate Wikipedia concepts returned is limited to five at most. If more than five top candidate WikiConcepts are returned, only the first five of them are retained: b) The number of associated WikiConcepts returned for each candidate word is set to 10. Wikipedia Thesaurus by default returns 30 associated WikiConcepts for each candidate term.

From the above two procedures, the maximum number of Wikipedia concepts Wikipedia Thesaurus can extract from one term is set to 55 (5 top candidates plus 50 associated concepts (10 for each candidate)) .

C. *Training/Testing Procedures (For Reuters)*

The training procedures are as follows: a) For classifiers, kNN (k = 3) is used: b) Each training set consists of 500 documents (100 documents for each category). Training set is fixed for each testing set. c) CL$_0$ is trained on TD (of the entire training set) with 10-fold cross validation. d) CL*i* in the ensemble (*E*) of *k* CLs (*k* = total number of CLs: *k* = 7 in this study) is trained on $\{i_{th}$WD - Subset(i_{th}WD,*i*)$\}$. e) Two CL$_0$ are included in the ensemble: *E* = {CL$_0$, CL$_0$, CL$_1$, CL$_2$, … , CL$_k$} (*k* = 7 in this study; the reason that the ensemble contains the classifiers in that proportion (two CL$_{0s}$) is to reflect the relative importance of the information contained in TD. The rest are given the same weight.): f) CL$_0$ and *E* are retained for future testing (the performance of *E* is to be compared against the performance of CL$_0$ as a base classifier).

The testing procedures are as follows: a) For classifiers, kNN (k = 3) is used: b) Each test set consists of 500 randomly selected documents (100 documents for each category): c) For each test set, 10-fold cross validation is performed to measure the performance of CL$_0$ and *E*: d) For *E*, a plurality voting technique is used.

D. *Experimental Results*

TABLE I shows the classification performances for three different text representation approaches: TD alone, TD augmented with WD (TD&WD), and 7 WD matrices with plurality voting techniques. The top number in each cell represents the accuracy measure achieved by TD matrix alone. The number in the middle in bold is the accuracy measure for the corresponding text representation/classification method, whose label is provided above in the corresponding column. The percentage values indicate the degree of change in the accuracy from TD alone to the corresponding methods.

For the comparison between TD and TD&WD for Reuters (1st column), TD and 7WDs for 20NG (3rd column), and TD and 7WDs for RCV1 (4th column), ten rounds of tests are performed, and the results are averaged. For the comparison between TD and 7WDs for Reuters (2nd column), thirty rounds of tests are performed (TABLE II.).

The thirty-round test recorded the accuracy measures achieved by each classifier (CL$_0$, CL$_1$, CL$_2$, … , CL$_7$, and E). The classifier CL$_0$ is trained on TD; the classifier CL$_1$ is trained on 1stWD; and so on. The last column represents an ensemble of 9 classifiers: *E* = {CL$_0$, CL$_0$, CL1, CL$_2$, …, CL$_7$}.

I. Analysis

A. *Interpretations*

The results show relatively small improvements achieved by the method using multiple WD matrices compared with the classifier trained only on the TD matrix. One probable explanation for the small improvement is that the word usage patterns and distributions are fairly consistent throughout the document corpora used; therefore, without the help of Wikipedia reference, the original text representation of TD matrix can capture a good generalization of the relations between a set of terms and a true category. The corpora used consist of news articles written by professional writers, and the consistency in their writing styles is usually observable in news articles covering the same topics, industries, cultures, etc.

Also many studies [2][11] have discovered the relative importance of information captured in TD matrix, even when text representations constructed with the aid of external knowledge are available. Whether a semantically projected matrix (such as WD) can only play a supplemental role, or it is possible to construct a standalone text representation effectively projected in true conceptual space is a challenging, yet very rewarding, issue that deserves further investigation.

TABLE I. Accuracy Comparison

	Reuters		20 NG	RCV1
TD/	TD&WD	7 WDs Plurality	7 WDs Plurality	7WDs Plurality
Accuracy	0.9252/ **0.9331** (+0.85%)	0.9228/ **0.9312** (+0.91%)	0.7451/ **0.7563** (+1.50%)	0.5102/ **0.5198** (+1.88%)

TABLE II. 30 Test Results for Reuters (Accuracy)

Test Set	*TD*	*1stWD*	*2ndWD*	*3rdWD*	*4thWD*	*5thWD*	*6thWE*	*7thWD*	*WDs (Ens)*
1	0.9420	0.9240	0.9280	0.9380	0.9400	0.9240	0.9240	0.9340	**0.9440**
2	0.9120	0.9020	0.9160	0.9200	0.9260	0.9080	0.9280	0.9260	**0.9300**
3	0.9240	0.9140	0.9200	0.9260	0.9140	0.9060	0.9120	0.9140	**0.9300**
4	0.9300	0.9160	0.9280	0.9240	0.9220	0.9140	0.9220	0.9060	**0.9340**
5	0.9140	0.9200	0.9200	0.9100	**0.9280**	0.9140	0.9040	0.9120	**0.9280**
6	0.9300	0.9120	0.9340	0.9360	0.9340	0.9360	0.9320	0.9340	**0.9480**
7	0.9260	0.9120	0.9380	0.9360	**0.9520**	0.9260	0.9180	0.9320	0.9440
8	0.9260	0.9220	0.9260	0.9060	0.9240	0.9260	**0.9280**	0.9200	**0.9280**
9	0.9240	0.9180	0.9200	0.9120	0.9220	0.9140	0.9000	0.9100	**0.9260**
10	0.9140	0.9040	0.9140	**0.9240**	0.9080	0.9120	0.9060	0.9020	0.9140
11	0.9320	0.9180	0.9340	0.9120	0.9280	0.9260	0.9240	0.9180	**0.9400**
12	0.9140	0.9020	0.9180	0.9160	**0.9260**	0.9200	0.9080	0.9020	**0.9260**
13	0.9180	0.8960	0.9200	0.9120	0.9220	0.9140	**0.9300**	0.9040	0.9280
14	0.9400	0.9360	0.9520	0.9360	0.9540	0.9400	0.9380	0.9380	**0.9580**
15	0.9060	0.9200	0.9180	0.8980	0.9080	0.9100	0.9100	0.8980	**0.9220**
16	**0.9400**	0.9180	**0.9400**	**0.9400**	0.9280	0.9340	0.9280	0.9280	0.9360
17	0.9260	0.9060	**0.9400**	0.9240	0.9280	0.9160	0.9220	0.9320	0.9320
18	0.9280	0.9260	0.9160	0.9140	0.9360	0.9220	**0.9380**	0.9180	0.9280
19	**0.9240**	0.9180	**0.9240**	0.9220	0.9200	0.8980	0.9140	0.9020	**0.9240**
20	0.9280	0.9260	0.9140	0.9240	0.9320	0.9160	0.9280	0.9260	**0.9360**
21	0.9120	0.9120	0.9220	0.9180	**0.9280**	0.9140	0.9240	0.9120	0.9260
22	0.9020	0.9160	0.9140	0.9220	0.9140	0.9200	**0.9240**	0.9140	0.9160
23	**0.9220**	0.9080	0.9100	0.9100	0.9080	0.9000	0.9100	0.9120	**0.9220**
24	0.9320	0.9400	0.9240	0.9280	0.9240	0.9220	0.9240	0.9140	**0.9440**
25	0.9160	**0.9240**	0.9180	0.9080	0.9180	0.9140	0.9140	0.9180	**0.9240**
26	0.9200	0.9160	0.9240	0.9180	**0.9320**	0.9140	0.9300	0.9060	0.9280
27	0.9140	0.9120	**0.9300**	0.9100	0.9220	0.9040	0.9220	0.9160	0.9260
28	0.9040	0.8900	0.8980	0.9020	0.9140	0.8960	0.9060	0.9000	**0.9200**
29	0.9320	0.9200	0.9240	0.9260	0.9200	0.9320	0.9360	0.9240	**0.9400**
30	0.9320	0.9260	0.9340	0.9260	**0.9380**	0.9120	0.9260	0.9220	0.9340
Average	0.9228	0.9158	0.9239	0.9199	0.9257	0.9168	0.9210	0.9165	**0.9312**
Ranking	4th	9th	3rd	6th	2nd	7th	5th	8th	**1st**

More specifically, the small improvements achieved by the multiple WD matrix representation could be attributed to its partial reliance on the voting strategy (it is not a total reliance, as the results listed in TABLE II. show that there are two WDs (excluding the ensemble) that performed better on average than TD on their own). In hindsight (with posteriori knowledge), it is easy to say that a construction of the best performing WD matrix alone should be made and used for classification. Posteriori knowledge, of course, is not available when testing in a new circumstance; hence comes the reliance on the voting technique (and the rationale for dividing TD matrix into multiple WDs). In fact, the voting strategy resulted in a higher average accuracy measure than that achieved by the best performing classifier (4thWD) in the ensemble.

B. *Strengths and Weaknesses*

The strengths of the multiple WD matrix strategy are: Stable performance improvement can be achieved by multiple WDs over TD alone. (In 27 out of 30 test sets for Reuters, *E* performed better than CL_0: in 19 out of 30, *E* performed better than all the other classifiers (including ties with one or two other classifiers).); Wikipedia is a work in progress; it is continuously updated every hour every day.

The weaknesses are: Extra work is required to extract Wikipedia concepts from original texts; it is faced with the difficulty in retrieving only those Wikipedia concepts that are strongly related and relevant to query terms; more performance improvement is desirable.

II. Conclusion

This paper discussed how Wikipedia can help improve text classification performance by providing semantic contexts and structures to transform plain texts into more semantically rich document representation.

This paper proposed a novel method for implementing the Wikipedia-assisted text classification. The method mainly consists of: 1) Multiple WD matrix creation. 2) Classification with multiple WD matrices. The experiments conducted show that the method provides improvement in the classification performance, when compared with the results of a classifier trained on a regular term-document matrix.

The experiments also found that still more improvements are needed, especially for Wikipedia concept extraction accuracy. Given the nature of Wikipedia, which is ever expanding and improving, further investigations into its better exploitation in text classification may prove more than worthwhile.

References

[1] G. Salton, A. Wong, and C.S. Yang, “A vector space model for automatic indexing,” *Commun. ACM*, vol. 18, 1975, pp. 613-620.

[2] Gabrilovich, Evgeniy and Shaul Markovitch, “Wikipedia-based Semantic Interpretation for Natural Language Processing,” *Journal of Artificial Intelligence Research X (YYYY) 1-1.*

[3] H. Chu, *Information representation and retrieval in the digital age*, Information Today, Inc., 2003.

[4] T.K. Landauer, P.W. Foltz, and D. Laham, “Introduction to Latent Semantic Analysis,” *Discourse Processes*, vol. 25, 1998, pp. 259–284.

[5] “Wikipedia - Wikipedia, the free encyclopedia.” [online]. Available: http://www.wikipedia.org. [Accessed: Aug. 14, 2009].

[6] “Encyclopedia - Britannica Online Encyclopedia.” [online]. Available: http://www.britannica.com. [Accessed: Aug. 14, 2009].

[7] D.D. Lewis, “RCV1.” [online]. Available: http://www.ai.mit.edu/projects/jmlr/papers/volume5/lewis04a/lyrl2004_rcv1v2_README.htm. [Accessed: Aug. 14, 2009].

[8] “Airbus A330 - Wikipedia, the free encyclopedia.” [online]. Available: http://en.wikipedia.org/wiki/A330. [Accessed: Aug. 14, 2009].

[9] “Wikipedia-Lab.” [online]. Available: http://wikipedia-lab.org/en/index.php/Main_Page. [Accessed: Aug. 15, 2009].

[10] M. Ito, K. Nakayama, T. Hara, and S. Nishio, “Association thesaurus construction methods based on link co-occurrence analysis for wikipedia,” *Proceeding of the 17th ACM conference on Information and knowledge management*, Napa Valley, California, USA: ACM, 2008, pp. 817-826.

[11] G. Ifrim, M. Theobald, and G. Weikum, “Learning Word-to-Concept Mappings for Automatic Text Classification,” *Proceedings of the 22nd International Conference on Machine Learning - Learning in Web Search (LWS 2005)*, L.D. Raedt and S. Wrobel, Eds., Bonn, Germany: 2005, pp. 18–26.

[12] R. Polikar, “Ensemble based systems in decision making,” *Circuits and Systems Magazine, IEEE*, vol. 6, 2006, pp. 21-45.

[13] M. Van Erp, L. Vuurpijl, and L. Schomaker, “An overview and comparison of voting methods for pattern recognition,” *HOBOKEN(NJ), IEEE. PROCEEDINGS OF THE 8TH INTERNATIONAL WORKSHOP ON FRONTIERS IN HANDWRITING RECOGNITION (WFHR02*, 2002, pp. 195--200.

[14] X. Lin, S. Yacoub, J. Burns, and S. Simske, “Performance analysis of pattern classifier combination by plurality voting,” *Pattern Recogn. Lett.*, vol. 24, 2003, pp. 1959-1969.

[15] “Reuters-21578 Text Categorization Collection.” [online]. Available: http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html. [Accessed: Aug. 18, 2009].

[16] “20 Newsgroups Data Set.” [online]. Available: http://people.csail.mit.edu/jrennie/20Newsgroups/. [Accessed: Aug. 18, 2009].

[17] X. Zhou, X. Zhang, and X. Hu, “Dragon Toolkit: Incorporating Auto-Learned Semantic Knowledge into Large-Scale Text Retrieval and Mining,” *ICTAI '07: Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence - Vol.2 (ICTAI 2007)*, IEEE Computer Society, 2007, pp. 201, 197.