Optimizing Support Vector Machine Classification Based on Semantic-Text Knowledge Enrichment *

Mr. Shadi Diab **

Mr. Nasim Hamaydeh ***

* Received: 12/12/2018, Accepted: 31/12/2018.

** Instructor/Al-Quds Open University/Palestine.
*** Instructor/Al-Quds Open University/Palestine.

DOI: https://doi.org/10.5281/zenodo.2582947

Abstract

In this research, we enhanced the performance of Support Vector Machine (SVM) in text classification by applying semantic-knowledge enrichment. We propose using semantic-knowledge enrichment scheme to inject new concepts into the original contents of the text documents. A pre-processing technique is proposed for cleaning and extracting features for generating semantic concepts through using WordNet database and the open source Natural Language Toolkit (NLTK). Additionally, the combined online variation Bayes algorithm and the Latent Dirichlet Allocation model are used as a dimensionality reduction technique to generate abstract concepts from the raw text. In our experiment, we clarified the process of preparing data for cleaning, transformation and weighting the features vectors in a multi-dimensional space as a step to measure the performance metrics of SVM, before and after applying our proposed approach on two different datasets. K-Fold Cross-Validation technique is used to validate our proposed approach. Moreover, a confusion matrix is implemented to measure the accuracy and macro-averages of precision, recall and f1 measurements. The result of the evaluation showed improvements in term of accuracy from 94% to 98.3% for the dataset-1, and from 88% to 93% for dataset-2. Moreover, the training time of the classifier in terms of seconds was reduced to 32% and 17% for dataset-1 and dataset-2 respectively, in comparison with the training time of the original data before applying our proposed enrichment scheme.

Keywords:

Support Vector Machine; Semantic Enrichment; Text Classification; Latent Dirichlet Allocation; Semantic Concepts Extraction.

" تطوير كفاءة خوارزمية "متّجهات دعم الآلة "

في تصنيف النصوص من خلال إثراء المحتوى الدلالي"

ملخص

يقدم هذا البحث تطويراً لكفاءة خوارزمية (Support Vector Machine) في تصنيف الملفات النصية من خلال تقنية اثراء المعرفة الدلالية للمحتويات النصية، والتي تهدف الى استخراج وبناء مجموعة من المفاهيم ذات العلاقة الدلالية بالمحتوى النصي وحقنها ضمن المحتوى الأصلي للملفات النصية. يقدم البحث طريقة منهجية في تصفية وتنقية البيانات واستخراج الخصائص والميزات منها قبل المعالجة في استخراج المفاهيم الدلالية من النص والتي تمت باستخدام قاعدة بيانات (WordNet) ومكتبة (NLTK). وباستخدام تقنية تخفيض حجم البيانات بإتباع طريقة نموذج (DLA) وخوارزمية (Online Variation Bayes). قدّم البحث إمكانية استخراج مجموعة من المفاهيم الدلالية التي تمثل البيانات الخام في الملفات الأصلية. في تجربة البحث تم استعرض آلية تحضير البيانات، تحويلها بشكل متجهات في الفضاء المتعدد الأبعاد، ثم تطبيق ذلك على خوارزمية (SVM) وقياس أداء الخوارزمية على مرحلتين: قبل وبعد تطبيق وحقن المفاهيم الدلالية في المحتوى النصي لمجموعتي بيانات مختلفتين، تم التحقق من صحة النهج المقترح في تحسين كفاءة خوارزمية (SVM) وذلك باستخدام آلية تحقق (K-Fold Cross Validation)، تم تقييم النهج المقترح باستخدام مصفوفة الارتباك (Confusion-Matrix) لقياس أداء الخوارزمية قبل وبعد تطبيق النهج المقترح، أما نتائج التقييم فأظهرت تحسن في دقة الخوارزمية من 94% الى 98.3% لمجموعة البيانات الأولى، و من 88% الى 93% لمجموعة البيانات الثانية، كما أن وقت المعالجة المستهلك بالثانية انخفض بنسبة 32% و 17% لكلا مجموعتي البيانات الأولى والثانية على التوالي بالمقارنة مع الوقت المستهلك عند استخدام البيانات الأصلية.

الكلمات المفتاحية:

(متجهات دعم الآلة)، الإثراء الدلالي، تصنيف النصوص، نموذج (DLA)، استخراج المفاهيم الدلالية.

Introduction and Literature Review

The massive content of the digitized text documents has increased rapidly in the last decades. Using search engines, web pages, emails and social media in the current era became part in our real routine life, not only at work but also at home, even while walking in the street. The textual contents of these text repositories are growing rapidly, in addition to the growing use of mobile applications, contents sharing, and social media networks. Consequently, the text classification task is becoming more challenging in achieving a more accurate efficient classification model.

The text classification task requires a given set of classes and a learning model to classify new unseen documents based on the learned process [1]. Several studies and research papers have been conducted to address the text classification technique by using different algorithms and methods. The commonly used methods for text classifications are decision tree based, rule-based, Neural Network classifiers, Bayesian classifiers, Support Vector Machine (SVM) classifiers, and some of the other classifiers that can perform text classification such as nearest neighbor classifiers, and genetic algorithm-based classifiers [2]. Different optimization methods have been used on SVM to enhance the performance of the classifier. Some of these methods are based on changing the underlying distribution of the data, while others combine different techniques and classifiers to enhance the overall performance of the classification task [3]. Moreover, many studies have been conducted on SVM in many research areas since 1995, after its introduction by Vapnik [4]. [5] [6] [7] [8] [9] [10] [11] Introduced SMV to improve face, images, and voice recognition. In text classifications, many researchers attempted to enhance the performance of SVM by different approaches and techniques, for instance through combining different kernels to reduce the computation cost of SVM [12], constructing hierarchical structure with a mixing linear SVM classifier [13], using the transductive SVM instead of inductive method [14]. Others developed instance selection approach to improve SVM [15], combining word2vec to improve SVM [16]. [17] Created virtual examples to be used with SVM. [18] Combined sparse representation classifier and (SVM) classifier by using frequency-based kernels. [19] Explored the effect of small sample size on the performance of text classifications, while [20] compared the performance of SVMs using different kernels with conventional learning methods. [21] Described adjusting the thresholds of SVM to enhance the performance. On the other hand, many researchers used semantics-based approaches to enhance the SVM, [22] deployed semantic-preserving dimension reduction by representing text with semi-supervised learning. Their results showed a good enhancement in SVM with some limits in precision and f1 measurements. [23] [24] Deployed Latent Semantic Indexing and Local Relevancy Weighted LSI. Other researchers studied the effects of named entity recognition and Part of Speech Tagging on SVM [25] [26]. In terms of using external resources to improve the text classification, [27] integrated the semantic-background knowledge, extracted from Wikitology, to enrich the text documents. Their experiment showed +6% improvement for the f1-measure of SVM classifier on 20-Newsgroups and Reuter’s datasets. The most relevant research to our work is [28], where the researchers extracted concepts from WordNet by mapping words to concepts, using multivariate chi-square in order to reduce the dimensionality and to create a coordinator profile for the documents. The researchers used different strategies to add, replace or remove concepts and terms, in addition to two strategies to tackle the word sense-disambiguation problem. The cosine distance between profiles was calculated to measure the performance of their approach. Their evaluation showed an improvement in f1-measure increasing from 0.649 to 0.714, and from 0.667 to 0.719, on Reuters-21578 and 20Newsgroup dataset respectively.

In this research, we propose SVM classifier with semantic knowledge-enrichment technique to enhance the performance of the classifier in term of accuracy and the execution time. We investigated the effects of transformation on some selected features semantically to be injected as semantically enriched vectors, and how such transformation could improve the performance of the SVM classifier?.

The order of the paper is as follows:

Section 2 and 3 introduce briefly SVM classifier and semantic knowledge enrichment. Sections 4 and 5 introduce more background details about Latent Dirichlet Allocation, and WordNet. Section 6 introduces more details about our research methodology. Our experiment and validation are introduced in section 7. The final evaluation and results are presented in section 8, and the last section concludes the proposed work and the results.

Support Vector Machine (SVM)

Support Vector Machines (SVMs) are designed based on the structural risk minimization (SRM) principle as an alternative to the traditional empirical risk minimization (ERM) principle [4] [1]. Classifiers based on SVM seek to find the optimal separating hyperplane to separate the data area through using linear or non-linear hyperplane (H) between the classes to be used as a separator for the purposes of classifications. The best separator between the different classes is the one that has the largest normal distance from the nearest data sample (maximum margin of separation).

Figure I illustrates SVM in a two-dimensional space for word1 and word2. In text classification by SVM, each document is associated with a category, ("+" or "-"). It shows the separator (H) between classes (+ and -), and d+ and d- are the shortest distance to the closest positive and negative points.

For any given linearly separable training dataset, we have a set of training samples in form of (xi, yi) for all instances of x from 1...n where x1, x2...xn represent the input sample features (input vectors) with label y. The expected output of SVM is a set of weights Wi vectors, one for each feature, which predicts the value of y, and b denotes the perpendicular distance from the hyperplane to the root. See figure 1.

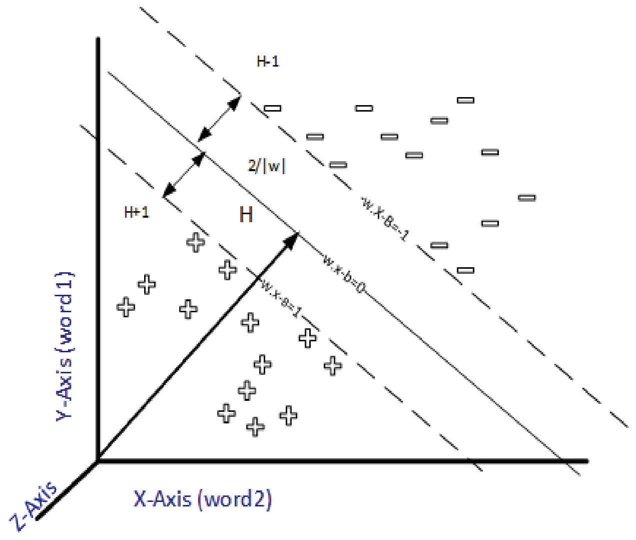

Figure I: Illustration of Linear Support Vector Machine

Subsequently, we can mathematically define the hyperplane (H) of SVM for positive and negative classes such that,

$w \bullet xi + b \geq +1$ if $yi = +1$

$w \bullet xi + b \leq -1$ if $yi = -1$

Then, (H+1) and (H-1) can be calculated as follow:

$H+1 = w \bullet xi + b = +1$

$H-1 = w \bullet xi + b = -1$

While the total distance between H+1 and H-1 is [29], minimizing |w| in a condition that no any point between H+1 and H-2 will maximize the margin, and this will lead to finding the optimal separating hyperplane.

Therefore, the classification decision for any new data (text document) with hyperplane (w, b) can be mathematically calculated through finding sign (w • x +b).

SVM classifier has been considered an extremely fast machine learning algorithm for multi-class classification problems [30]. It performs very well for classification of text data while it is not affected by the high number of features when having high dimensional input space as in text classification. Moreover, it is memory efficient because it uses support vectors, and it is flexible in choosing different kernels (for example Linear, Polynomial, RBF kernels) and custom kernels to be used for the decision function. While most text categorization problems are separable problems linearly [14], we will consider linear SVM classifier in this research.

Semantic-Knowledge Enrichment.

Text classification requires extracting and representing the unstructured data (text) into numerical features applicable to the machine learning algorithms. The use of representation of the bag-of-words is commonly the implemented method. This method has different processing steps such as tokenizing the text into separated tokens, counting the tokens to figure the frequency of each token in the text, and normalizing and weighting the tokens. In fact, the bag of word approach ignores the relation of words and their position information. Therefore, there are a lot of semantic information that will be missed when the bag-of-words approach is implemented. In practice, this approach leads to the most important problem in natural language processing which is identifying the exact meaning of the words. For example, in English language, the word "tank" has different meanings such as a large receptacle or storage chamber, heavy armored fighting vehicle and a cell in a police station or jail. In order to tackle the problem several approaches were used and they can be categorized under three types, knowledge-based approach, supervised approach and unsupervised approach [31]. In this research, an external knowledge-based approach is used to extract, process, and inject a background-knowledge into the text representation by using online variation Bayes approach [32] with Latent Dirichlet Allocation (LDA) model [33] and WordNet [34].

Latent Dirichlet Allocation (LDA)

Several methods are used to generate hidden semantic features from a collection of text documents. In order to achieve this, different sets of algorithms are used. The idea was introduced by [33] as information retrieval technique based on analysis of the term-document matrix to extract the underlying semantics. A probabilistic Latent Semantic Analysis (PLSA) model was introduced by [35] and it adopts the same assumptions of LSI, with different probabilistic generative steps that generate terms from text documents. Whereas PLSI has a large number of parameters that grow linearly with the number of documents. The Latent Dirichlet Allocation (LDA) model generates different suggested topics for each document. The reason it reduces the number of learned parameters is to provide clear topic through discovering and employing patterns of the word occurrences in the text document. LDA is considered the most commonly used model in topic modeling, taking into consideration its ability to generate a wide range of main topics. These topics are needed in our research, in order to have more than one nominated topic for each text document. Moreover, Latent Dirichlet Allocation (LDA) is a Bayesian probabilistic model while the online LDA is based on online stochastic optimization with a natural gradient step. It was introduced by [32].

WordNet Database

Contradictory to the traditional language dictionaries, which are structured alphabetically, WordNet is a lexical database, which is structured semantically. It contains a collection of verbs, nouns, adverbs and adjectives organized into sets of synonyms or synsets to represent the lexical concept which is interlinked with many relations. WordNet has the capability to measure the semantic similarity and relatedness between the concepts. Different approaches are used to achieve semantic similarity values. In our research, we used Wu-Palmer algorithm [36] to calculate similarity score between the concepts in WordNet.

Research Methodology

In this research, we inject related semantic-knowledge into the text documents. Therefore, we designed pre-processing steps to prepare the data before extracting the related knowledge. A comparison-based experiment was implemented on the performance of the classifier before and after applying our proposed approach, in order to evaluate the performance of SVM classifier in the text classification. The following sub-sections describe the research methodology in details.

I. Collecting and Preparing Datasets

Several textual datasets have been used in text classification research. In our research, we used two datasets in our experiment, the first is the 20-Newsgroups dataset, which is a widely used dataset for experiments and research related to text classification and clustering. It contains approximately 20,000 text documents divided into different 20 newsgroups category. Each category is related to a specific subject, but some of them are closely related to each other [37]. The second dataset is Syskill and Webert dataset, which has four different topics (Bands, Bio-Medical, Goats, and sheep). It contains 62, 137, 71, and 66 different HTML documents respectively. Each document has a text related to the topic of the category. The dataset is collected by [38] and downloaded from The UCI Machine Learning Repository [39].

II. Data cleaning and extracting features

Inspecting and deleting unnecessary words from text is required to avoid processing unrelated terms to save time, space and costs. We designed a removal tool to inspect and delete the stop words as well as perform tokenization to extract one feature that describes each category name. The following steps and figure II clarify the data cleaning and feature extraction steps:

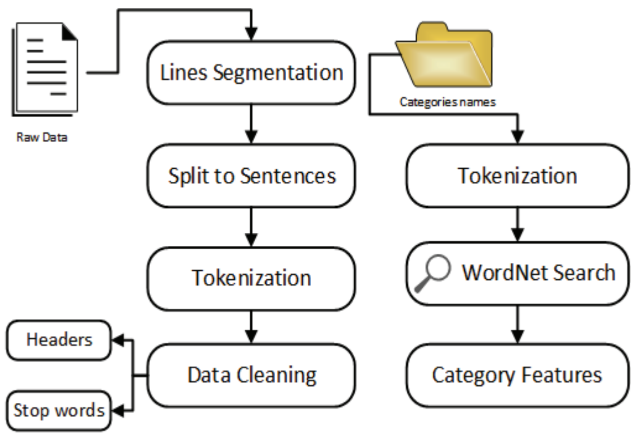

Figure II: Data cleaning and extracting categories features

- ♦ Removing headers: in the 20 newsgroups dataset, each document contains an unstructured text in form of messages and email letters. In order to prepare the data for processing to contain the body text of the email message, including some useful information of the header section, we deleted all header information that does not include specific information related to the category name itself. We noticed that the header fields are not identical for all categories and some of the fields may contain useful information to improve the classification results. Therefore, for the 20 newsgroups, the removal tool inspects the header lines in each file to remove all the headers fields except newsgroup, subject, organization, keywords, summary, follow-up, and archive name fields.
- ♦ Removing stop words based on the built-in stop list of the Natural Language Toolkit (NLTK) library [40], after expanding the stop words list to include all special characters, questions words, pronouns, and auxiliary verbs in the English language.
- ♦ Tokenizing text and extracting features: to split the text into separated words and punctuations, we tokenized each line of the file by using NLTK word tokenizer. Moreover, our tokenization tool tokenized names of all categories to extract at least one feature that represents the category name. The feature name must exist in the WordNet dataset. For example, the category names in 20 newsgroups: [alt.atheism, comp.graphics, comp.os.ms-windows.misc, comp.sys.mac.hardware, misc.forsale] became [atheism, graphics, operting_system, macintosh for_sale,] based on semantically closest concept available in WordNet database.

III. Generating Concepts

As illustrated in our enrichment scheme in figure III, the scheme goal is to extract the top abstract concepts that occur in the documents for each category in the dataset. LDA model is used and configured as a generative bag-of-word topic model to analyze the text, and nominate top 10 concepts for each document. LDA can be considered a dimensionality reduction technique, in the spirit of LSI but with implicit generative probabilistic semantics that makes sense for the type of data that it models [41]. For each document in each category, and after data cleaning and feature extraction as illustrated in figure II, we transformed the data into numeric features as an input in the DLA model. The following clarifies steps of generating concepts in details:

1. Features Extraction: to extract numerical features after preparing and cleaning each document, where a list of cleaned tokens are converted to numerical vectors using a bag of words representation, and the count features are weighed through using TF-IDF (Term Frequency, Inverse Document Frequency) transformation, we transformed the cleaned text to vectors through TF-IDF transformation and normalized the output through Euclidian normalization based on the following equations:

$$\mathbf{TF}\ (\text{term, document})\ \mathbf{x}\ \mathbf{IDF}\ (\text{term})$$

$$\mathbf{IDF}\ (\text{term}) = \log\frac{n_d}{\text{df(dociment,term)}} + 1$$

$$\textbf{Normalization of Vector (v)} =$$

$$\frac{v}{||v||^2} = \frac{V}{\sqrt{V1^2+V2^2+\cdots+Vn^2}}$$

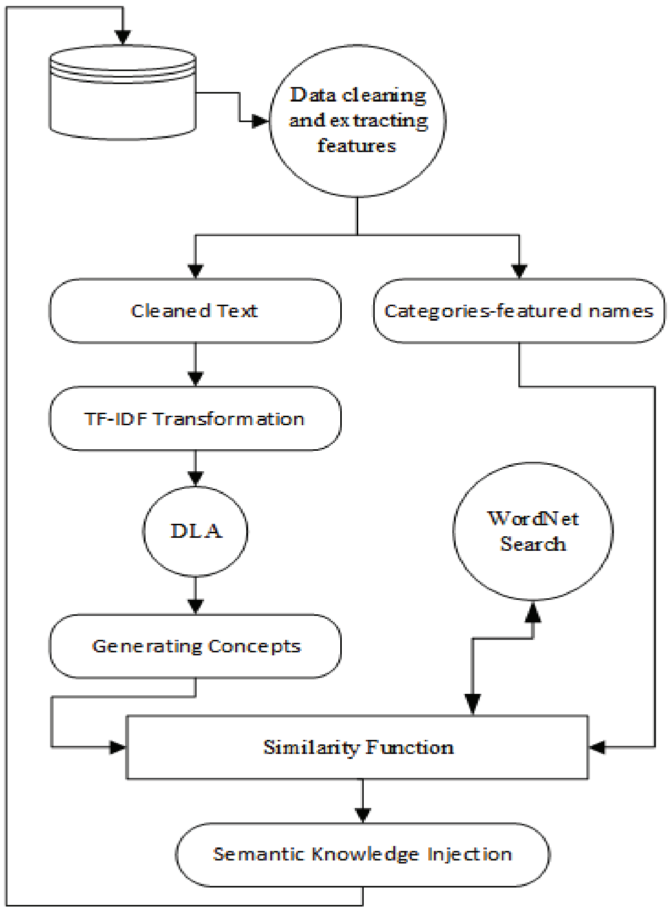
Data cleaning and extracting features
Cleaned Text
Categories-featured names
TF-IDF Transformation
DLA
WordNet Search
Generating Concepts
Similarity Function
Semantic Knowledge Injection

Figure III: Semantic-knowledge enrichment scheme

2. We extracted top 10 concepts for each document by using LDA model and through variation Bayes approach. The outputs of the LDA is stored in a separate list to be used for generating powerful concepts related to the contents of the document.

IV.Injecting Semantic-Knowledge Background

Our approach to enrich the text document is by injecting powerful related concepts into the document prior to performing training on the classifier. Wherefore, we extracted all synsets for each category-featured name by using WordNet, and then a canonical form for each synset returned. A similarity function is designed to calculate the similarity score among the generated top 10 concepts and the categories-featured names, through using Wu-Palmer algorithm. As illustrated in figure III, steps of producing and processing the synsets and lemma names for each concept to generate and inject powerful concepts into the text of the document are briefly explained in the following points:

1. Cleaning the raw data in each file and extracting feature for each category name based on the WordNet database.
2. Extracting the top 10 concepts that represent each file through using LDA.
3. Finding synonyms that share a common meaning (Synsets) for each category-featured name, in additional to the top 10 concepts synonyms, through using WordNet dataset. For example, two forms of new synsets will be generated for the synset (Computer), synset <computer> and synset <compute >.
4. Lemmatizing synsets of each category featured name and the top 10 concepts based on the morphological analysis of the words for each generated synsets. For example, extracting the lemma names for word (computer) will generate the following concepts, 'computer', 'computing_machine', 'computing_device', 'data_processor', 'electronic_computer', 'information_processing_system', 'calculator', 'reckoner', 'figurer', 'estimator', 'computer'.
5. Calculating the semantic similarity values between the synsets of each category-featured name and the synsets of the top 10 concepts through using a designed similarity check function. The function itself generates a list of similarity synsets on the condition that the similarity values between them meet the selected similarity threshold. In our experiment, we used a 90% threshold as a similarity score, where all synsets that have 90% and greater will be considered in the injection approach.
6. Appending the lemma names for each synset that successfully met the condition of similarity check threshold to the end of the processed text file.

VII. Experiment and Validation

Our experiment seeks to check the performance of SVM in the classification of a semantic-knowledge enriched dataset. The dataset was downloaded from The UCI Machine Learning Repository [39]. Fifty percent of the original dataset of the 20 newsgroups has been created as a development set. Our implementation is carried out on Intel(R) Core(TM) i7-6600U CPU @ 2.60GHz, 2 Core(s), 4 Logical Processor(s) and 16.0 GB RAM computer machine. We used python programming language with scipy.sparse data structure [42], and Scikit-learn module [43], and NLTK, in addition to the 10 K-Fold Cross-Validation to validate our approach. Our cleaning and removal tool is used to clean and prepare our data for processing. We performed tokenization, vectorization and transformation of the data to numeric features vectors. Then we normalized and weighed the count features by using TF-IDF transformation. Moreover, a confusion matrix is used to record the macro average of accuracy, precision, recall and f1 measurements to be used for performance comparison. By using our development set, we measured the performance of the linear SVM in two states, the first state is before enriching the dataset by our semantic knowledge enrichment scheme and the second state is when we applied our approach to enrich the semantic-knowledge.

Table I clarifies the performance comparison of linear SVM before and after applying our semantic knowledge enrichment scheme on both datasets (dataset-1, the 20newsgroup, and dataset-2, the Syskill and Webert).

Table I:

Performance of Linear SVM on the development sets

Classifier	Macro-Average				Time / Sec
	Accuracy	Precision	Recall	f1	
Dataset-1 before	0.9429	0.9424	0.9429	0.9426	9.483
Dataset-1 after	0.9796	0.9798	0.9798	0.9797	6.620
Datset-2 before	0.8888	0.8928	0.8888	0.8860	0.240
Datset-2 after	0.9345	0.9385	0.9345	0.9333	0.210

Based on table I, it becomes clear that our knowledge-enrichment approach enhances the accuracy of linear SVM on the development set of the 20 newsgroups dataset, increasing it from 94.3% to 98%, and from 88% to 93% for the second dataset. For validation, we divided the training dataset into 10 different smaller datasets to validate the performance of linear SVM for each state in which the classifier is trained by using K-1 of the folds as a new training set and validated by using the remaining fold. The result of the 10 K-Folds Cross-Validation for datset-1 and datset-2 in term of the mean of accuracy (μ) and standard deviation (σ) before and after implementing our semantic knowledge enrichment is summarized in table II.

Table II:

10 K-Folds Cross-Validation for dataset-1 and datset-2

Linear SVM	Accuracy		Time / Sec.
	μ	σ	
Dataset-1 before enrichment	0.9387	+/- 0.0363	32.717
Dataset-1 after enrichment	0.9787	+/- 0.0188	21.402
Datset-2 before enrichment	0.8992	+/- 0.2489	0.260
Datset-2 after enrichment	0.9326	+/- 0.1952	0.282

Evaluation

We evaluated our approach by using the unseen testing set on the same classifier by comparing the two states mentioned in the experiment and validation section. The result of our evaluation of 30% testing dataset for datset1 and dataset2 is clarified in tables III, where there is improvement in the accuracy and the macro averages of precision, recall and f1 measure after processing the data and injecting the semantic knowledge to the source text in each document in the dataset. In comparison with [28] for the same dataset (the 20newsgroup) the achieved macro-average of f1 after applying their approach reached 71.9%, while in our approach and after applying our semantic knowledge enrichment the f1 measurement reached 98.3% .

Table III:

Evaluation before and after applying the enrichment scheme

Linear SVM	Macro-Average				Time / Sec
	Accuracy	Precision	Recall	f1	
Dataset-1 before	0.9408	0.9413	0.9418	0.9415	10.716
Dataset-1 after	0.9836	0.9838	0.9836	0.9837	7.292
Datset-2 before	0.8834	0.8888	0.8835	0.8796	0.290
Datset-2 after	0.9306	0.936	0.9306	0.9295	0.240

Conclusion

In this research, we proposed using semantic knowledge enrichment to enhance the performance of Support Vector Machine classification. In the experiment, we implemented linear SVM on two different datasets and measured the effects of injecting new semantic-knowledge to the contents of text files as a step to enhance the whole classification technique. In our experiment and validation, we explored different metrics to compare the performance of Linear SVM before and after implementing our proposed semantic knowledge enrichment on the same datasets. Therefore, and based on the evaluation process, we can conclude that the performance of SVM is increased from 94% to 98.3% in term of accuracy for dataset-1, and from 88% to 93% for dataset-2. Moreover, the proposed approach reduced the training time of the classifier to 32% and 17% in terms of seconds for dataset-1 and dataset-2 respectively, in comparison with the training time of the original dataset before applying our proposed approach.

References

1. Catarina Silva, Bernardete Ribeiro, Inductive Inference for Large Scale Text Classification: Kernel Approaches and Techniques, Springer-Verlag Berlin Heidelberg, 2010, p. 3.
2. C.C. Aggarwal and C.X. Zhai, Mining Text Data, Boston: Springer, Boston, MA, 2012, pp. 165-167.
3. C.C. Aggarwal and C.X. Zhai (eds.), Mining Text Data, London: Springer Science+Business Media, LLC , 2012, p. 209.
4. V. Vapnik, The Nature of Statistical Learning Theory, 2 ed., Springer-Verlag New York, 2000, p. 8.
5. P. Shih et al., «Face detection using discriminating feature analysis and Support Vector Machine,» Pattern Recognition, vol. 39, no. 2, pp. 260-276, 2006.
6. M. D. L. Samuel Kadoury, «Face detection in gray scale images using locally linear embeddings,» Computer Vision and Image Understanding, vol. 105, no. 1, pp. 1-20, 2007.
7. S. Lee et. al., «SVM-based feature extraction for face recognition,» Pattern Recognition, vol. 43, no. 8, 2010.
8. Déniz, O et al., «Face recognition using independent component analysis and support vector machines,» Pattern Recognition Letters, vol. 24, no. 13, 2003.
9. Wang, Jiakailin et al., «A Face Recognition System Based on Local Binary Patterns and Support Vector Machine for Home Security Service Robot,» in 9th International Symposium on Computational Intelligence and Design (ISCID), 2016.
10. Cumani, S., & Laface, P., « Large-Scale Training of Pairwise Support Vector Machines for Speaker Recognition,» IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 22, pp. 1590-1600.
11. O. Eray, S. Tokat and S. Iplikci, An application of speech recognition with support vector machines, 2018, pp. 1-6.
12. Liwei Wei et.al, «Text Classification Using Support Vector Machine with Mixture of Kernel,» Journal of Software Engineering and Applications, vol. 5, pp. 55-58, 2012.
13. Di Wang, Xiaoqin Zhang, Mingyu Fan, Xiuzi Ye, «Hierarchical mixing linear support vector machines for nonlinear classification,» Pattern Recognition, vol. 59, 2016.
14. J. Thorsten, «Transductive Inference for Text Classification Using Support Vector Machines,» in Proceedings of the Sixteenth International Conference on Machine Learning, 1999.
15. Tsai, C., & Chang, C., «SVOIS: Support Vector Oriented Instance Selection for text classification,» Inf. Syst, vol. 38, pp. 1070-1083, 2013.
16. J. Lilleberg and Y. Zhu and Y. Zhang, «Support vector machines and Word2vec for text classification with semantic features,» in 2015 IEEE 14th International Conference on Cognitive Informatics Cognitive Computing (ICCI*CC.
17. M. Sassano, «Virtual Examples for Text Classification with Support Vector Machines,» in Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing, Stroudsburg.
18. N. Sharma and A. Sharma and V. Thenkanidiyoor and A. D. Dileep, «Text classification using combined sparse representation classifiers and support vector machines,» in 2016 4th International Symposium on Computational and Business Intelligence (ISCBI), 2016.
19. P. Matykiewicz and J. Pestian, «Effect of small sample size on text categorization with support vector,» in Proceedings of the 2012 Workshop on Biomedical Natural Language Processing (BioNLP 2012), Montreal.
20. J. T, «Text categorization with Support Vector Machines: Learning with many relevant features,» Lecture Notes in Artificial Intelligence, vol. 1398, 1998.
21. Shanahan J.G and Roma N., «Shanahan J.G., Roma N. (2003) Improving SVM Text Classification Performance through Threshold Adjustment. In: Lavrač N., Gamberger D., Blockeel H., Todorovski

L. (eds) Machine Learning: ECML 2003. ECML 2003. Lecture Notes in Computer Science, vol 2837. Sp,» Machine Learning: ECML 2003. ECML 2003. Lecture Notes in Computer Science, vol. 2837, 2003.

22. LU et al, «Enhancing Text Categorization with Semantic-enriched Representation and Training Data Augmentation,» Journal of the American Medical Informatics Association , vol. 13 , no. 5, p. P 526, 2006.

23. Y. Huang, «Support Vector Machines for Text Categorization Based on Latent Semantic Indexing,» 2001.

24. Tao Liu and Zheng Chen and Benyu Zhang and Wei-ying Ma and Gongyi Wu, «Improving text classification using local latent semantic indexing,» in Fourth IEEE International Conference on Data Mining (ICDM'04), 2004.

25. Bam, S. and Shahi, T, «Named Entity Recognition for Nepali Text Using Support Vector Machines,» Intelligent Information Management, vol. 6, pp. 21-29, 2014.

26. Tej Bahadur Shahi, Tank Nath Dhamala and Bikash Balami, «Support Vector Machines based Part of Speech Tagging for Nepali Text.,» International Journal of Computer Applications, vol. 70, no. 24, pp. 38-42.

27. Muhammad Rafi and Sundus Hassan and Mohammad Shahid Shaikh, «Content-based Text Categorization using Wikitology,» CoRR, vol. abs/1208.3623.

28. Z.Elberrichi et al., «Using WordNet for Text Categorization,» The International Arab Journal of Information Technology, vol. 5, no. 1, 2008.

29. «Why does the SVM margin is 2 / ||w||," Mathematics Stack Exchange, [Online]. Available: https://math.stackexchange.com/questions/1305925/why-does-the-svm-margin-is-frac2-mathbfw. [Accessed 7 7 18].

30. Huang and Kecman, «LinearSVM,» Huang and Kecman, [Online]. Available: http://www.linearsvm.com/. [Accessed 06 06 2018].

31. A. Ranjan and D. Saha, «Word Sense Disambiguation: A Survey,» nternational Journal of Control Theory and Computer Modeling (IJCTCM), vol. 5, no. 3, 2015.

32. Papadimitriou, Christos H. and Tamaki, Hisao and Raghavan, Prabhakar and Vempala, Santosh, «Latent Semantic Indexing: A Probabilistic Analysis,» in the Seventeenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, Seattle, 1998.

33. G. A. Miller, «WordNet - A Lexical Database for English,» Communications of the ACM, vol. 38, no. 11, pp. 39-41, 2005.

34. T. Hofmann, «Probabilistic Latent Semantic Indexing,» in the Twenty-Second Annual International SIGIR Conference on Research and Development in Information Retrieval, Berkeley, 1999.

35. Z. W. a. M. Palmer, «VERB SEMANTICS AND LEXICAL SELECTION,» in the 32nd Annual Meeting of the Associations for Computational Linguistics, New Mexico, 1994.

36. L. K, «Newsweeder: learning to filter netnews,» in Proceedings of the 12th international conference on machine learning (ICML'95), 1995.

37. Pazzani, M., Muramatsu J., and Billsus, D, «Identifying interesting web sites,» in Proceedings of the National Conference on Artificial Intelligence, Portland, OR, 1996.

38. Dheeru, Dua and Karra Taniskidou, Efi, «Machine Learning Repository,» University of California, Irvine, School of Information and Computer Sciences, 2017. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Twenty+Newsgroups. [Accessed 11 11 2018].

39. Steven Bird, Natural Language Processing with Python, vol. 1, J. Steele, Ed., O'Reilly Media, Inc., 2009, p. 222.

40. David M. Blei, Andrew Y. Ng and Michael I. Jordan, «Latent Dirichlet Allocation,» Journal of Machine Learning Research 3 (2003) , vol. 3, pp. 993-1022, 2003.

41. Jones E, Oliphant E, Peterson P, et al, «SciPy : Open Source Scientific Tools for Python,» SciPy , [Online]. Available: http://www.scipy.org. [Accessed 11 10 2017].

42. Pedregosa et al., «Scikit-learn: Machine Learning in Python,» Journal of Machine Learning Research, vol. 12, pp. 2825-2830, 2011.