EAI Endorsed Transactions
on Industrial Networks and Intelligent Systems
Research Article EAI.EU

Transforming Data with Ontology and Word Embedding for an Efficient Classification Framework

Thi Thanh Sang Nguyen[1,*], Pham Minh Thu Do[1], Thanh Tuan Nguyen[2], Thanh Tho Quan[3]

[1]School of Computer Science and Engineering, International University, VNU-HCMC, Hochiminh City, Vietnam
[2]School of Computing and Mathematical Sciences, University of Greenwich, UK
[3]Faculty of Computer Science and Engineering, Ho Chi Minh City University of Technology, VNU-HCMC, Hochiminh City, Vietnam

Abstract

Transforming data into appropriate formats is crucial because it can speed up the training process and enhance the performance of classification algorithms. It is, however, challenging due to the complicated process, resource-intensive and preserved meaning of the data. This study proposes new approaches to building knowledge representation models using word-embedding and ontology techniques, which can transform text data into digital data and still keep semantic/context information of themselves in order to enhance modeling data later. To evaluate the effectiveness of the built models, a classification framework is proposed and performed on a public real dataset. Experimental results show that the constructed knowledge representation models contribute significantly to the performance of classification methods.

Received on 25 September 2022; accepted on 05 February 2023; published on 01 June 2023

Keywords: Onto2Vec; Doc2Vec; Ontology; Classification

doi:10.4108/eetinis.v10i2.2726

1. Introduction

Data classification is very popular in data mining. Several techniques in this fields are Naïve Bayes, Decision tree induction, Support Vector Machines, Bayesian Networks, or Neural networks, etc. To enhance the performance of classifiers, the data preparation step is extremely important because it involves cleaning the data and transforming it into efficient formats for effective measurements and classifications. In recommender systems, this processing is very challenging because the input data, often being user transactions or profiles, is daily collected, large and appears in various types. It is, therefore, necessary to carefully analyze and transform data to make it simple before training. Many different approaches have been investigated to deal with this issue. For example, user profile and item data are vectorized, and combined through Neural Collaborative Filtering layers before training [1]. Sequential user preference data are able to be captured by RNN-based models and integrated with the key-value memory network of knowledge base, which is then useful for the recommendation [2]. On the other hand, some vector-based transformation methods have been applied to the Book-crossing dataset to improve Naïve Bayes-based classification in [3]. In particular, the text data is transformed to vectors using the Word2Vec model [4], and numbers. As a result, the accuracy of classification has been improved when using the transformed data. Besides, feature selection also plays an important role in recognize good features and lift up classification accuracy [5, 6], so it is a means of validating whether the transformed features are more useful than the original ones.

In this paper, we present three approaches to knowledge base construction serving for a semantic recommendation/prediction. The first approach is to use deep learning methods to encode text data to vectors which are efficient to compare and retrieve information. The second approach is a new one to building automatically ontologies of information objects for the semantic inference. The third approach is to vectorize the built ontologies for saving search space and speed up the information retrieval. To examine

*Corresponding author. Email: nttsang@hcmiu.edu.vn

these approaches, a classification framework has been newly constructed and includes four phases: (1) data preprocessing, (2) knowledge base construction, (3) feature selection, and (4) classification. In that, Phase 2 and 3 play an important role in enhancing making recommendations/predictions. Several experimental cases, using the Book-crossing dataset, are carried out to evaluate the proposed approaches as well as the constructed framework.

The following presents related work in Section 2, the details of proposed methods in Section 3, experiments in Section 4, and conclusions in Section 5.

2. Related work

Evolving technologies are generating enormous raw data in various data types such as numeric, image, video, text, etc. How can we process, analyze, and convert these data into the useful information for us? In this work, we focus on text data and the method to deal with text-related problems which is called the Natural Language Processing (NLP). It refers to the ability of a computer to understand and communicate with humans in the human language and other language-related tasks. Sophisticated machine learning techniques are exploiting due to the evolving of large text data and their complex and dynamic contents such as ambiguity, syntax, semantics, co-reference, normalization, etc. We will review several of them as below.

2.1. Word embeddings

Word2Vec model is a word embedding model proposed by Mikolov et al. [4] which is built using Neural Network to acquire the semantic meaning of the human language [7]. The input is the text body and outputs are low dimensional space vectors. Skip-gram model and Continuous Bag of Words (CBOW) are two architectures of Word2Vec models. The Skip-gram model, introduced as an efficient model for huge numbers of word vector representations in unstructured text, tries to predict the context words based on current words, whereas the CBOW model tries to predict current words based on context words. Hierarchical Softmax and Negative Sampling methods can be used to evaluate Word2Vec models [4, 8, 9].

Like Word2Vec models, sentences and paragraphs can also be vectorized, called Doc2Vec models. The main idea is to treat a paragraph/document as a word then embedded in a vector space. An embedding is a fixed-length vector to encode and represent a sentence, paragraph, or document. There are two approaches called Distributed Memory Model of Paragraph Vector (PV-DM) and Distributed Bag of Words version of Paragraph Vector (PV-DBOW) [10]. The PV-DBOW architecture is similar to the Skip-gram architecture but it works in the same way as CBOW. In the PV-DBOW model, a text window is sampled, from which a random word is sampled and classified or predicted. On the other hand, the PV-DM model considers concatenating the paragraph vector with the word vectors to predict the next word in the next window.

2.2. Ontology

According to Antoniou et al. [11], ontology is a formal representation of the knowledge bases, in that, concepts and relations among them are defined in one or different domains. It is a powerful tool to express the semantic information of a knowledge base. That is the reason Ontology is widely used to enhance semantically recommender systems. In such recommender systems, ontologies are designed in appropriate ontology models for the knowledge representation of specific domains. OWL (Web ontology language, http://www.w3.org/TR/owl-features/) is a main Web ontology language which satisfies the requirements of building a domain ontology, including a well-defined syntax, a well-defined semantics, efficient reasoning support, sufficient expressive power, and convenience of expression [12].

Ontologies can appear in some graphical or formal visualizations but they are encoded in an ontology language which enables machine-processable. Graphical appearance of an ontology might be semantic networks with interlinked conceptual nodes, or the taxonomic hierarchy of domain concepts and the customary relations between them. This graphical appearance, however, cannot express complex axioms in ontologies. Ontology languages like OWL can precisely define the meaning of an ontology in term of logic. Therefore, an ontology often appears as a set of logical formulas representing a set of axioms formalizing the represented knowledge in the ontology. For storage on disk or on the Web, an ontology needs to be expressed in some machine-processable serialization format, such as OWL/RDF/XML.

Ontology has made many valuable productions in many fields in terms of understanding text/domain data and express semantic information in the way machine-understandable. For example, semantic partitioning domain data would help more effective text classification and sentiment analysis [13].

2.3. Onto2Vec

Reasoning information from an ontology is time-consuming. Onto2Vec [14] is an approach to learn feature vectors for biological entities which are elements in biomedical ontologies. By applying the approach, we can generate vectors of entities or individuals in an ontology. Therefore, an ontology-based knowledge representation can be transformed

into vector spaces. The benefits of this approach are to facilitate retrieving information.

3. Methodology

In this study, we propose a number of solutions for transforming data with the purpose to improve the performance of classification, a crucial and useful step in recommender systems. Two main approaches taken into account are ontology and deep learning, making a novel data transformation and classification more efficient. Based on that, a novel classification framework is proposed as presented at Fig. 1. The framework consists of four units: (1) data preprocessing, (2) knowledge base construction, (3) feature selection, and (4) classification.

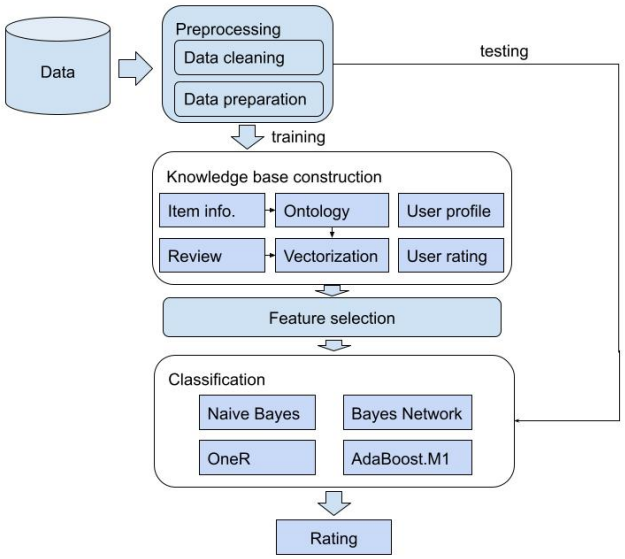


Figure 1. Proposed framework of the classification system.

The knowledge base construction unit plays an important role in data transformation and lifting the data quality. The raw input data are collected from e-commerce sites. They include the Item (Product) information, User (Customer) profile, and User transactions (Sessions). User transactions, e.g. rating, shopping, or reviewing (commenting), are important to discover customer behavior. The data process is described as follows:

1. Data preprocessing is the first unit to clean the raw data. It includes removing invalid words, stop words, or invalid records. In this study, sessions are made by user ratings showing how users are interested in items. Data types are often numeric, nominal, and string. To easily compare records, string values should be vectorized to become numbers in the next step. However, in some cases, numeric values considered as nominal are effective in data modeling, this is shown in [3]. Then, given the pre-processed data, data objects, e.g. items, users, and sessions, or data features are identified to be used in the Knowledge base construction step.

2. The Knowledge base construction unit is necessary to enhance the performance of classification models. In this process unit, several data types are considered to be transformed into vectors or numbers to facilitate comparison and classification. The text data, e.g. reviews, is able to be transformed into vectors using word embedding. The object data, e.g. books, is represented semantically by an ontology and then also encoded into vectors. These approaches will be presented in detail later. Besides, the location attribute in the user dataset (if any) is also converted to numeric data, i.e. their latitude and longitude values can be found with the help of the OpenStreetMap API (https://wiki.openstreetmap.org/wiki/API).

3. The Feature selection unit is performed before classification. Its goal is to select best features/attributes for building classifier models. The features which most affect the rating class/label should be involved in training learned models [3]. It is necessary to analyse attribute values in order to remove outliers and normalize data before selecting attributes. Several tools for the attribute selection, available in the Weka library [15], have been used such as OneR, Symmetrical Uncertainty Ranking Filter, Gain Ratio, Information Gain, and WrapperSubsetEval.

4. The Classification unit is used to train and classify data as well as evaluate the proposed framework. The outputs of this unit are user ratings. For training and testing the classifier models, the raw data after preprocessing is splitted into two parts: the training part is gone through all units to obtain the learned models, but the testing part is passed to the learned models and to output user ratings. A number of typical classification algorithms are employed from basic to advanced, such as OneR, decision trees (e.g. C4.5 [16]), Naïve Bayes (NB), Bayesian Networks (BN) [17] and Adaboost.M1 [18] in this unit. At present, there are many machine learning libraries available for building classifiers, but good input records (well-prepared) are essential for those methods. Therefore, this study mainly focuses on the knowledge base construction for the classification.

3.1. Knowledge base construction

Doc2vec. In this study, the found text attributes are object titles and reviews. Several different Neural Network methods in NLP are utilised to vectorize these datasets as described below.

Word2Vec-based Text data is transformed to vectors using Word2Vec [4]. In particular, each word w_i in a document is able to be represented by a real-valued vector v_i. Therefore, a document, a set of n words, is presented as a vector, which can be computed to get a real value, as follows [3]:

- The length of sum vector of the word vectors: $|\sum_{i=1}^{n} v_i|$; or
- The dot product of the word vectors: $\prod_{i=1}^{n} v_i$; or
- The length of mean vector of the word vectors: $|\frac{1}{n}\sum_{i=1}^{n} v_i|$.

In [3], these values are computed for each title object, defined as:

1. The length of the sum vector of the word vectors in a title, namely, **titleVecSumLen**;
2. The dot product of the word vectors in a title, namely, **titleDotPro**; and
3. The length of the mean vector of the word vectors in a title, namely, **titleVecMean**.

Moreover, item reviews are also transformed into vectors, and then the length of the mean vector of word vectors in each review is calculated, i.e., **Review_Vec**. These transforming jobs had been done in the previous study [3], so this study utilizes their results and compares them with the following new transformations.

Doc2Vec-based On the other hand, in this study, the two Doc2Vec methods PV-DM and DBOW (i.e., PV-DBOW) are also applied to transform review data into vectors. Similarly, the length of the vector of a review is computed. Therefore, each review is now represented as a number, namely **Review_Vec_plus**.

Doc2Vec model is trained with labels and the actual data. The labels can be anything, but in this implementation, they are unique ids automatically generated and assigned to each item review, or a document. Each document is normalized to remove unnecessary words, email addresses, or URL links.

Doc2Vec is implemented in a python environment with hyperparameters:

- Dimensionality of the feature vectors is set to 300 to embed each document.
- The maximum distance between the current and predicted word within a sentence is set to 5.
- The maximum epoch is 30.
- The initial learning rate is set to 0.025. At each training epoch, the learning rate is reduced by 0.0002.

This Doc2Vec model is used to transform the review data to the Review_Vec_plus data.

Ontology. In this study, ontology is studied to design the domain of items, e.g. books or movies, and their features in the domain application. The ontology model of items is presented in Fig. 2. It is considered as a foundational ontology [19], then a domain ontology, e.g. the entities of books, will be added later.

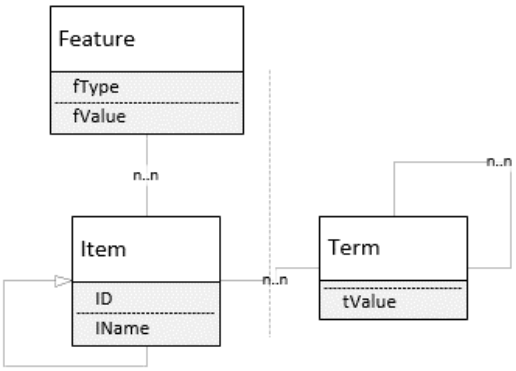


Figure 2. Ontology model of items.

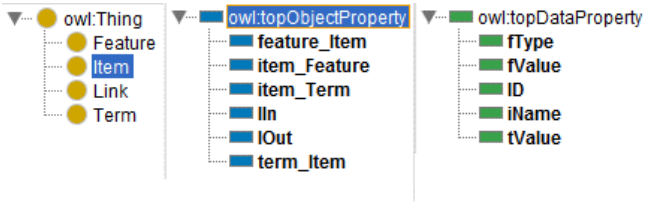


Figure 3. The built ontology.

As shown in Fig. 2, Item class has ID and IName attributes. IName referring to the title attribute of an item is tokenized into meaningful keywords. Since we want to manage keywords in titles in order to describe the relationships among items and keywords, each keyword is defined by the Term class. Moreover, the collocation of keywords in titles is considered, so Term should associate with itself. Term has the tValue attribute storing an item value (or keyword). Other features, such as book author, publisher, are defined by the Feature class, because these features are able to be shared by many items. Feature has the fType and fValue attributes. For example, an author is a feature

of a book item, author name 'abc' has the author type (fType = 'author') and the value of 'abc' (fValue = 'abc'). The ontology model is built in Protégé as Fig. 3. There are four classes: Feature, Item, Link, and Terms. The Link class is the association of two terms. The object properties are described in Table 1.

Table 1. Ontology description

Object property	Description
feature_Item	Domain: Feature; Range: Item
item_Feature	Inverse of feature_Item
item_Term	Domain: Item; Range: Term
term_Item	Inverse of item_Term
lIn	Domain: Link; Range: Term
lOut	Domain: Link; Range: Term

This ontology model is designed generally, so that it can be applied to many application domains, e.g. books, movies, or products. Moreover, the ontology population will be easier. Although this model is simple, ontology data will increase significantly if the amount of instances is huge. Therefore, the model should be decomposed into three parts: Item-feature, Item-term and Term as shown in Fig. 4. The Item-feature part keeps the information of items and their features (OntoModel). The Item-term part presents the association of Item and Term (ItemTerm). The Term part presents term collocations (TermMap).

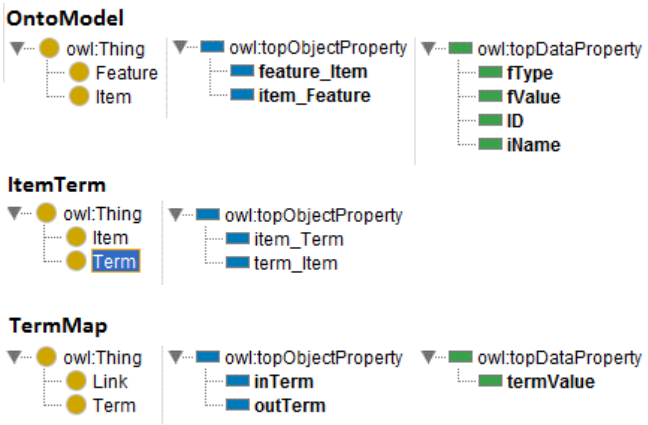


Figure 4. Fragmented Ontology model.

The ontology model is used to populate automatically ontologies of items and concepts/ terms connected to items from a large dataset efficiently.

Onto2Vec To improve the speed of matching ontology instances, the ontology is proposed to be represented in vectors by ultilising Doc2vec. This process model is called Onto2Vec. Because an ontology is like a semantic network, its data can be expressed in plain text. To achieve that, a random-walk algorithm (Algorithm 1) is developed to traverse through the ontology instances. We can obtain many sequences by walking through the ontology. One instance is recorded by each walk step. In one session, we go through a number of instances randomly as a sentence.

Algorithm 1 Random walk algorithm

Require: Ontology data (OntoModel, ItemTerm, TermMap)
Ensure: A text corpus
Set the max length of sentence randomly in range {8, 11}
Randomly walk to a node in item, term, feature in the ontology to generate sentences

1. if walking into an item, randomly walk to one of the features or terms
2. if walking into a feature, randomly walk to one of the items
3. if walking into a term, randomly walk to the next term or an item

Repeat (1) or (2) or (3) until the length = the max or walking to the same object
Save the generated sentences into the output corpus.

After generating the corpus of sentences from the ontology, the Doc2vec model, based on Word2Vec, is applied to vectorize these sentences. As a result, the ontology is vectorized. Item IDs are identified to find the corresponding vectors. The vector of each item ID is considered as the representative information of the item. In this way, we can calculate the mean of the vector of each item ID, namely ID2VecMean. After that, these values are fed into the training data as a new feature for classification and prediction. In this way, we can also solve the problem of big data, avoid storing text, save memory, and compute the distance between two item IDs easier.

3.2. Evaluation methods

The 10-fold cross-validation [20] with performance indicators is used to evaluate and compare among the classification models. The accuracy metric defining the ratio of correct classification over the total number of classified instances is used mostly in our experiments.

Additionally, two error metrics, e.g. Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE), are also taken into account to measure the performance of classifiers in this study. They are computed as follows [21]:

$$MAE = \frac{|p_1 - a_1| + .. + |p_n - a_n|}{n} \quad (1)$$

$$RMSE = \sqrt{\frac{(p_1 - a_1)^2 + .. + (p_n - a_n)^2}{n}} \quad (2)$$

Where n is the number of classified instances; p_1,p_2,..,p_n are the classified class values; and a_1,a_2,.., a_n are the actual class values.

4. Experiments

4.1. Data preparation

Experiments use the Book-crossing dataset (available at http://www2.informatik.uni-freiburg.de/~cziegler/BX/) containing 278,858 users, 271,379 books, and 1,149,780 ratings, including some missing values. In order to take into account book reviews which might be influential on book-rating, the Amazon book review dataset, downloaded at http://jmcauley.ucsd.edu/data/amazon/, is explored. Book reviews are extracted and selected corresponding to ISBNs in the Book-crossing dataset. As a result, 40,642 book reviews have been found, and there are 169,834 book-rating records having book reviews and being able to be used in experiments. However, the reviewerID in the Amazon book review dataset is different from the userId, and there are many reviews on a book. Thus, a long enough review is selected as the reference for each book. Similarly, the average of overall ratings in this dataset is also computed for each book, namely overall_score.

For this book dataset, an item is a book, so the data of books, their features (ISBN, author, year, publisher) and titles, is fed into the proposed ontology model in order to generate vectors using Onto2Vec. As a result, each book is represented by an **ISBN2VecMean** (or a number).

As a result, the book-rating records discovered in the experiments have 17 attributes: “ISBN”, “titleDotPro", “titleVecSumLen", “titleVecMean", “author”, “year", “publisher", “ISBN2VecMean", “Review_Vec", “Review_Vec_plus", “overall_score", “userId", “location", “latitude", “longitude", “age", and “rating". The “rating" feature is the class attribute. In that,

- The values of "titleDotPro", "titleVecSumLen", and "titleVecMean" attributes are transformed from book titles.
- The value of "Review_Vec" attribute is transformed from book reviews.
- The value of "Review_Vec_plus" attribute is transformed from book reviews as mentioned above.
- The values of "latitude" and "longitude" attributes are transformed from "location" in the user data.

In the study of [5], classifying string attributes was not efficient. Therefore, this study considers only nominal and numeric attributes. That is why the transformed titles and reviews are used in steads of the original ones. There are two types of studied datasets: the nom one (all attribute values are set to nominal), and the nom.num one (numeric attribute values are kept as they are). Moreover, rating values should be scaled to binary values in order to obtain higher classification performance. Rate values 1-5 are scaled to 0 for ‘bad’, and rate values 6-10 are scaled to 1 for ‘good’.

4.2. Data analysis

Table 2 describes the 17 attributes of the book data used in this study. Most of the attributes are numeric, except for ISBN, author, publisher, userid, location, and rating which are nominal. The latitude and longitude attributes have 2% missing values, and the age attribute has 27% missing values.

Table 2. Book data analysis

Attribute	Distinct	Type
ISBN	23958	Nominal
titleDotPro		Numeric
titleVecSumLen		Numeric
titleVecMean		Numeric
author	14426	Nominal
year		Numeric
publisher	2290	Nominal
ISBN2VecMean		Numeric
Review_Vec		Numeric
Review_Vec_plus		Numeric
overall_score		Numeric
userid	31586	Nominal
location	10152	Nominal
latitude	Missing 2%	Numeric
longitude	Missing 2%	Numeric
age	Missing 27%	Numeric
rating	2	Nominal

The numeric attributes are analyzed in the following. The distributions of transformed titles are depicted in Fig. 5. TitleVecMean is nearly normal but titleDotPro and titleVecSumLen are not normal. Fig. 6 presents the distributions of latitude and longitude which are left-skewed. Normal attributes are considered to be good features for classification. These arguments will be proved in later experiments.

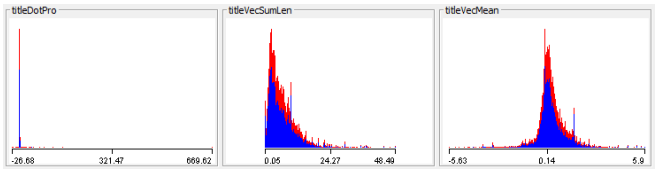


Figure 5. Distribution of transformed titles.

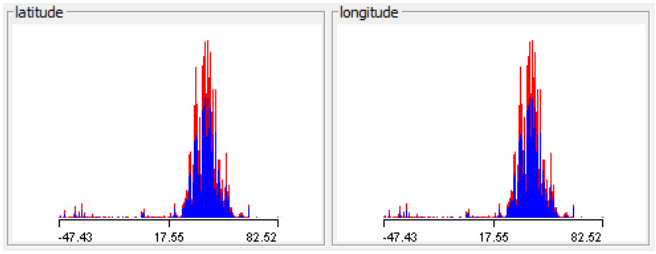


Figure 6. Distribution of transformed locations.

Since it is not reasonable to have Age values greater than 100 and 200 in the dataset, this attribute is normalized as shown in Fig. 7, i.e. invalid values are replaced with the mean. Fig. 8 shows the distribution of ISBN2VecMean, it is nearly normal.

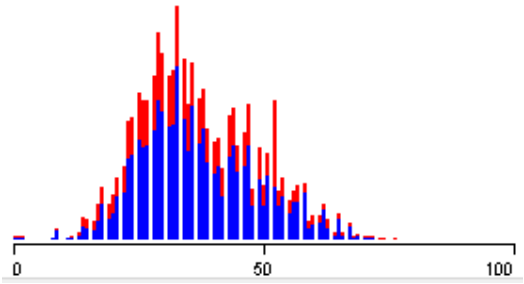


Figure 7. Normalized Age.

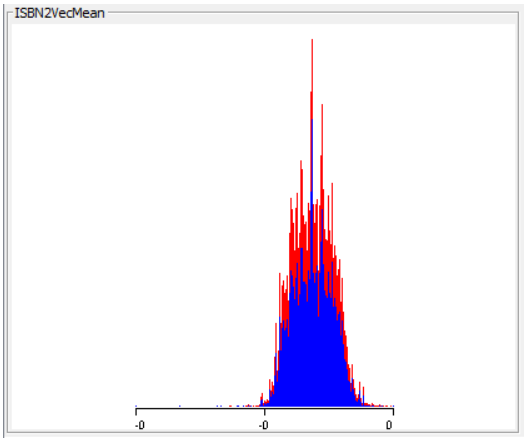
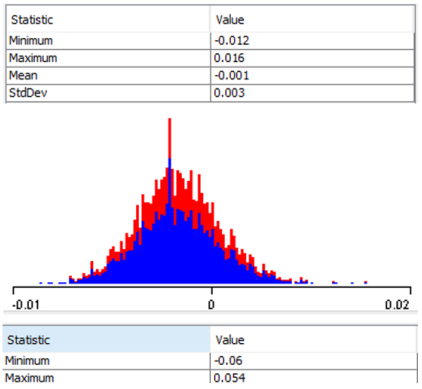
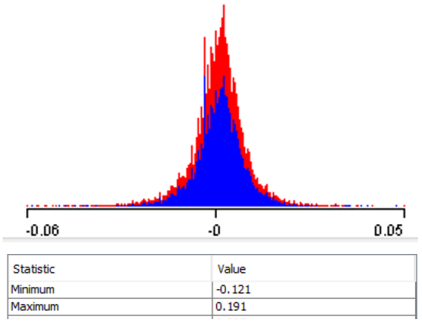
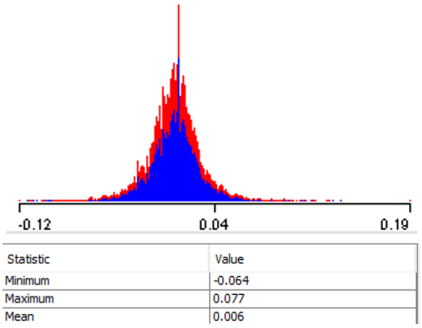
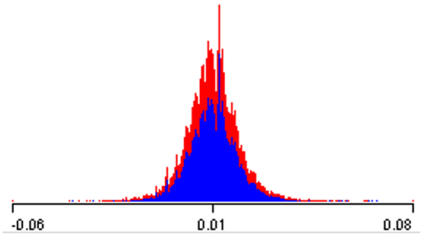


Figure 8. ISBN2VecMean distribution.

Table 3 presents the distributions of Review_Vec and Review_Vec_plus, mentioned in Sub-section 3.1. Review_Vec_plus values generated using PV-DM and DBOW with vector size 100 are noted as PVDM_100 and DBOW_100, respectively. When increasing vector size to 300, applied to PV-DM, the distribution of Review_Vec_plus is more normal (PVDM_300). It is expected that Review_Vec_plus should be the best selection for classification.

Table 3. Distributions of Review_Vec and Review_Vec_plus

Attribute	Distinct
Review_Vec	
Review_Vec_plus (PVDM_300)	
Review_Vec_plus (PVDM_100)	
Review_Vec_plus (DBOW_100)	

As shown, the data after transforming and normalizing is more effective for feature selection and modeling. Therefore, given this data, the Feature selection unit is performed, followed by the Classification unit.

4.3. Experimental results

In the following experiments, four cases are carried out to evaluate how the transformed attributes impact classification. The used classification algorithms are OneR, C4.5, AdaBoost.M1, Naïve Bayes (NB) and Bayesian Networks (BN). OneR is the base classification algorithm. C4.5 has been tested but taken too much time for training and given bad results, hence it will not be examined more.

Since the 10-fold cross-validation is used in all experimental cases, the average results of accuracy or MAE/RMSE are calculated from 10 iterations in the cross-validation and shown as final results. The experiments were run on a PC with an Intel Core i7-4770 processor, 3.40 GHz, and 16 GB of RAM.

Case 1. The original attributes ISBN, author, year, publisher, userId, location, age, and rating are used.

Case 1.1. The attributes are set to nominal. The experimental results are shown in Table 4.

Table 4. Case 1.1 results (nominal only)

Method	Accuracy (%)	MAE	RMSE	Runtime (s)
OneR	**71.7177**	**0.2828**	**0.5318**	**0.1**
AdaBoost.M1	67.8021	0.438	0.4663	1.14
NB	70.5036	0.3238	0.4557	**0.04**
BN	70.1756	0.3249	0.4627	**0.23**

Case 1.2. The attributes are nominal and numeric. The experimental results are shown in Table 5.

Table 5. Case 1.2 results (nominal and numeric))

Method	Accuracy (%)	MAE	RMSE	Runtime (s)
OneR	**71.7177**	**0.2828**	**0.5318**	**0.12**
AdaBoost.M1	67.8021	0.438	0.4663	1.7
NB	70.3069	0.327	0.4555	0.12
BN	70.1632	0.325	0.4624	0.47

The two above cases show that OneR dominates other algorithms.

Case 2. The ISBN2VecMean, userId, location, age, rating attributes are used.

ISBN is transformed to ISBN2VecMean as proposed in Section 3. An ISBN2VecMean represents a Book and its attributes. Tables 6 and 8 present experimental results in this case. Besides, we also try to test if the ontology vector of each book affects the classification performance, hence the ontology vector is scaled to 10 dimensions, namely BookVec-10, for validation; the results are shown in Tables 7 and 9. In Tables 6 and 7, all attributes are set to nominal. In Tables 8 and 9, numeric attributes are kept.

Table 6. Case 2.1.a results (nominal only))

Method	Accuracy (%)	MAE	RMSE	Runtime (s)
OneR	**71.7177**	**0.2828**	**0.5318**	**0.03**
AdaBoost.M1	67.8021	0.4381	0.4667	0.65
NB	**71.2119**	0.3309	0.4385	**0.03**
BN	**71.047**	0.3298	0.4425	**0.06**

Compared with Case 1, the accuracy of NB and BN become better when using ISBN2VecMean, and the runtimes in Case 2 are also faster. However, it is not good to use BookVec-10 as shown in Table 7. It will be better if the values of BookVec-10 are set to numeric (Table 9).

Table 7. Case 2.1.b results (nominal only, BookVec-10)

Method	Accuracy (%)	MAE	RMSE	Runtime (s)
OneR	71.7177	0.2828	0.5318	0.06
AdaBoost.M1	67.8021	0.438	0.4663	1.5
NB	63.2942	0.3705	0.5636	0.06
BN	62.4474	0.3783	0.5743	0.27

Table 8. Case 2.2.a results (nominal and numeric)

Method	Accuracy (%)	MAE	RMSE	Runtime (s)
OneR	71.7177	0.2828	0.5318	0.07
AdaBoost.M1	67.8021	0.438	0.467	1.39
NB	**70.8621**	0.3353	0.4389	**0.07**
BN	**71.1342**	0.3313	0.4419	**0.43**

Table 9. Case 2.2.b results (nominal and numeric, BookVec-10)

Method	Accuracy (%)	MAE	RMSE	Runtime (s)
OneR	71.7177	0.2828	0.5318	0.43
AdaBoost.M1	67.8021	0.438	0.467	6.2
NB	**70.9222**	0.335	0.4387	0.29
BN	**71.1459**	0.33	0.4413	1.03

As shown, the NB and BN algorithms are improved after transforming data. Hence, the following cases will concentrate on these two algorithms.

Case 3. The attribute selection strategies mentioned in Section 3 are applied. The nom.num datasets are used to evaluate numeric attributes. Rating is the class attribute, so the selected attributes are non-class attributes.

Case 3.1. Assess the attribute selection strategies for NB-based classification applied to the datasets of 16 attributes without Review_Vec_plus. In this sub-case, WrapperSubsetEval adopts the NB classifier to select attributes. As a result, the accuracy of NB is improved significantly (Table 10). The WrapperSubsetEval strategy gives the best attributes for classification. TitleVecMean and ISBN2VecMean have presented their effectiveness. In comparison with the previous experiments in [3], the attribute selection strategy and classification algorithm are similar to Case 3.2 in [3], in that the tuple of TitleVecSumLen, author, and userId could obtain the highest accuracy. However, that tuple is not the best choice in this case because it results in lower accuracy, that is why the tuple of titleVecMean, author, ISBN2VecMean, userId is selected. This has shown that ISBN2VecMean has a positive contribution to the classifier.

Table 10. Case 3.1 results

Method	Selected attributes	Accuracy	MAE	RMSE	Runtime
OneR	userId	71.6264	0.3499	0.4262	0.01
Symmetrical Uncertainty Ranking Filter	userId	71.6264	0.3499	0.4262	0.01
Gain Ratio	userId	71.6264	0.3499	0.4262	0.01
Information Gain	userId, ISBN	71.6523	0.3401	0.4301	0.02
WrapperSubsetEval (Adopting NB)	**titleVecMean, author, ISBN2VecMean, userId**	**71.9267**	**0.3392**	**0.4274**	**0.14**

Case 3.2. Apply the attribute selection strategies to evaluate attributes of nominal and numeric types in the datasets of 17 attributes. In this sub-case, Review_Vec_plus is involved to examine whether it is better than Review_Vec.

As known, WrapperSubsetEval got some benefits when adopting a classification algorithm to evaluate attributes before training. Therefore, this case studies some classification algorithms which are correspondingly adopted in WrapperSubsetEval.

In sub-case 3.2.a (Table 11), the nom.num datasets are used. When NB is adopted in WrapperSubsetEval, the ISBN2VecMean, age, author, titleVecMean, titleVecSumLen, and userId attributes are selected. Because both titleVecMean and titleVecSumLen represent the title attribute, each of them is examined separately as well. As a result, **titleVecSumLen** associated with ISBN2VecMean, age, author and userid can provide a better result of the NB-classification. The tuple of titleVecSumLen, author, and userId which was suggested in Case 3.2 in [3] could not give higher accuracy, so was not selected for classification. While, **titleVecMean** associated with ISBN2VecMean, Review_Vec, overall_score, userid and year can well support the BN-classification and give the best result. Moreover, **ISBN2VecMean** appearing in these good results points out itself important contribution. The C4.5 and AdaBoost.M1 algorithms provide worse results.

Table 11. Case 3.2.a results (nominal and numeric)

Adopted Method	Selected attributes	Accuracy (%)
NB	**ISBN2VecMean, age, author, titleVecMean, titleVecSumLen, userId**	**71.9367**
	ISBN2VecMean, age, author, titleVecMean, userId	**71.965**
	ISBN2VecMean, age, author, titleVecSumLen, userId	**71.9738**
C4.5	age, latitude, longitude, year	68.5622
BN	**ISBN2VecMean, Review_Vec, overall_score, titleVecMean, userId, year**	**72.1045**
AdaBoost.M1	Location	67.8015

It also shows that ISBN2VecMean, Review_Vec, and titleVecMean can improve the classification performance, especially in the BN-based classification. Attribute selection is not effective in the nom dataset, as shown in Table 12. Besides, this sub-case (3.2.b) did not work when running C4.5.

Table 12. Case 3.2.b results (nominal)

Adopted Method	Selected attributes	Accuracy (%)
NB	Overall_score, user_id	72.0998
C4.5	-	-
BN	Overall_score, user_id, year	**72.0574**
AdaBoost.M1	latitude	67.8015

Case 4. Evaluate the performance of word-embedding methods applied to review text. In this case, the word-embedding methods of PV-DM and DBOW are compared with the Word2Vec-based one. There are three variants of Review_Vec_plus generated:

- From the PV-DM-based model with vector size 100, namely PVDM_100.
- From the PV-DM-based model with vector size 300, namely PVDM_300.
- From the DBOW-based model with vector size 100, namely DBOW_100.

In addition, because this case uses the two types of training and testing data: nom and nom.num datasets, six experimental sub-cases are performed for each classification algorithm, as shown in Tables 13 and 14. For example, the nom.num_PVDM_300 case examines the nom.num dataset in which Review_Vec_plus values are generated from PVDM_300.

Case 4.1. Apply the attribute selection strategy of WrapperSubsetEval adopting NB to the datasets of 17 attributes with differently transformed reviews, and then NB is applied to classify the training data.

As seen in Table 13, based on the attribute selection strategy, **Review_Vec_plus** is selected in Case nom.num_PVDM_300; **ISBN2VecMean** and **titleVecMean** are selected in Case nom.num_DBOW_100, but the classification accuracy is not higher than the other cases in the nom datasets. This shows that the transformed attributes can improve the NB-based classification in the nom.num datasets, with an accuracy higher than the accuracy of OneR. Compared with Case 3.2, it is not better in terms of the NB-based classification performance. In other words, Review_Vec or **Review_Vec_plus** do not contribute much to the NB-based classification, in which, Review_Vec_plus is a little bit more effective, especially with vector size 300, so it was selected in Case nom.num_PVDM_300.

Table 13. Case 4.1 results (NB-based classification)

Case	Selected attributes	Accuracy (%)
nom.num_PVDM_300	age, author, Review_Vec_plus, userId, rating	71.9191
nom.num_PVDM_100	age, author, userId, rating	**71.9532**
nom_PVDM_300	Overall_score, user_id	**72.0998**
nom_PVDM_100	Overall_score, user_id	**72.0998**
nom_DBOW_100	Overall_score, user_id	72.0998
nom.num_DBOW_100	**ISBN2VecMean**, age, author, **titleVecMean**, userid	**71.965**

Case 4.2. Similar to Case 4.1, but NB is replaced with BN.

Table 14. Case 4.2 results (BN-based classification)

Case	Selected attributes	Accuracy (%)
nom.num_PVDM_300	overall_score, **Review_Vec_plus**, **titleVecSumLen**, userid, year	**72.1981**
nom.num_PVDM_100	overall_score, **Review_Vec_plus**, **titleVecSumLen**, userid, year	**72.1139**
nom_PVDM_300	overall_score, userid, year,	72.0574
nom_PVDM_100	overall_score, userid, year	72.0574
nom.num_DBOW_100	**ISBN2VecMean**, **Review_Vec**, overall_score, **titleVecMean**, userid, year	**72.1045**
nom_DBOW_100	Overall_score, user_id, year	72.0574

This sub-case shows that the transformed attributes contribute better to classification in the nom.num data (Table 14). **Review_Vec_plus** dominates Review_Vec when using the **PVDM_300** model. The larger vector size the Doc2Vec model has, the more effective the classification is. Compared with the DBOW_100 model, the PVDM_100 model is better to create more accurate vectors. Moreover, looking back at Table 3, the distribution of **Review_Vec_plus** is the best one. That means attributes with normal distributions should be selected for classification.

5. Conclusions

As seen, the Onto2Vec model is effective to generate **ISBN2VecMean** which can improve the classification performance. Moreover, the Word2Vec and Doc2Vec -based models can enhance the NB and BN –based classifications by transforming text data into numbers. In that, the **PV-DM-based model** is more effective to generate vectors. The digitalization of text data is not only saving memory space, but also speeds up the training time. Furthermore, the Onto2Vec-based data transformation is an innovative solution that can be applied to different kinds of domain data without manually constructing domain ontologies many times.

Besides, titleVecSumLen and titleVecMean are more effective than titleDotPro as selected in the attribute selection strategies. This is reasonable with the visual distributions of these attributes shown in Subsection 4.2, i.e., titleVecSumLen and titleVecMean are more normal than titleDotPro. In comparison with the previous study [3], the WrapperSubsetEval attribute selection strategy in combination with the NB classification algorithm can achieve the best results in Case 3.2 [3] and Case 3 in this study. The contribution of ISBN2VecMean to the tuple titleVecSumLen, author,

and userId has made better results than the previous study. It is noticed that the datasets used in this study are the same as the ones used in the previous study, but with fewer records. Therefore, the accuracy values are not compared in both studies, but the selected attributes are considered to evaluate the performance of the studied models. In the future, we will extend the proposed models to different datasets to evaluate the flexibility and scalability of the models.

Acknowledgement. This research is funded by Vietnam National University HoChiMinh City (VNU-HCM) under grant number C2021-28-06.

References

[1] He, Xiangnan, and Liao, Lizi and Zhang, Hanwang and Nie, Liqiang and Hu, Xia and Chua, Tat-Seng *Neural Collaborative Filtering*,2017, pp.173–182 (https://doi.org/10.1145/3038912.3052569)

[2] Huang, Jin, and Zhao, Wayne Xin and Dou, Hongjian and Wen, Ji-Rong and Chang, Edward Y. *Improving Sequential Recommendation with Knowledge-Enhanced Memory Networks*, 2018, pp.505–514, (https://doi.org/10.1145/3209978.3210017)

[3] Nguyen, Thi Thanh Sang, and Do, Pham Minh Thu *Classification optimization for training a large dataset with Naïve Bayes*. Journal of Combinatorial Optimization, 2020, https://doi.org/10.1007/s10878-020-00578-0.

[4] Mikolov, Tomas, and Sutskever, Ilya and Chen, Kai and Corrado, Greg and Dean, Jeffrey *Distributed Representations of Words and Phrases and their Compositionality*. Proceedings of the 26th International Conference on Neural Information Processing Systems. Neural and Information Processing System (NIPS), 2014,pp.3111-3119

[5] Nguyen, Thi Thanh Sang. *Model-Based Book Recommender Systems using Naive Bayes enhanced with Optimal Feature Selection*.2019, pp.217-222.

[6] Alhejaili, Abdullah and Shaheen, Fatima. Latent Feature Modelling for Recommender Systems. In 2020 IEEE 21st International Conference on Information Reuse and Integration for Data Science (IRI). 2020, pp. 349-356

[7] Rong, Xin *word2vec Parameter Learning Explained*. CoRR,2014, http://arxiv.org/abs/1411.2738

[8] Morin, Frederic, and Bengio, Yoshua *Hierarchical Probabilistic Neural Network Language Model*. 2005, http://www.gatsby.ucl.ac.uk/aistats/fullpapers/208.pdf

[9] Mikolov, Tomas, and Yih, Wen-tau and Zweig, Geoffrey.*Linguistic Regularities in Continuous Space Word Representations*. Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, pp.746-751

[10] Le, Quoc, and Mikolov, Tomas *Distributed representations of sentences and documents*, 2014 pp.II-1188-II-1196

[11] Antoniou, Grigoris and Harmelen, Frank van *A Semantic Web Primer*, MIT Press, 2008.

[12] Antoniou, Grigoris and Harmelen, Frank van, Handbook on Ontologies,Springer-Verlag Berlin Heidelberg, 2009. *Web Ontology Language: OWL.* , pp. 91-110, .

[13] Fayyoumi, Ebaa and Sahar, Idwan. *Semantic Partitioning and Machine Learning in Sentiment Analysis.* Data. **6** (2021)

[14] Smaili, Fatima Zohra and Gao, Xin and Hoehndorf, Robert *Onto2Vec: joint vector-based representation of biological entities and their ontology-based annotations*. Bioinformatics, 2018, pp. i52-i60. https://doi.org/10.1093/bioinformatics/bty259

[15] Frank, Eibe and Hall, Mark A. and Witten, Ian H.,. *The WEKA Workbench. Online Appendix for "Data Mining:Practical Machine Learning Tools and Techniques"*. 4th ed. Morgan Kaufmann, 2016.

[16] Quinlan, J. Ross*C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., 1993, pp.302

[17] Han, Jiawei, and Kamber, Micheline and Pei, Jian. Data Mining. 3rd ed. Morgan Kaufmann, 2012.*Chapter9 - Classification: Advanced Methods.*, pp. 393-442

[18] Freund, Yoav and Schapire, Robert E. (1996) *Experiments with a new boosting algorithm*.

[19] Trojahn, Cassia, Renata, Vieira, Daniela, Schmidt, Adam, Pease and Giancarlo, Guizzardi. *Foundational ontologies meet ontology matching: A survey*. Semantic Web Preprint, 2021 pp.1-20.

[20] Refaeilzadeh, Payam and Tang, Lei and Liu, Huan. Encyclopedia of Database Systems. Springer US,2009. *Cross-Validation.* , pp. 532-538.

[21] Witten, Ian H. and GFrank, Eibe and Hall, Mark A. Data Mining: Practical Machine Learning Tools and Techniques, 3rd ed. Morgan Kaufmann, 2011 *Chapter 5 - Credibility: Evaluating What's Been Learned.* , pp. i147-187.