Future Generation Computer Systems 96 (2019) 552–562

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Performance analysis of machine learning classifiers on improved concept vector space models

Zenun Kastrati [*], Ali Shariq Imran

Norwegian University of Science and Technology, Norway

ARTICLE INFO

Article history:
Received 24 July 2018
Received in revised form 29 December 2018
Accepted 5 February 2019
Available online 18 February 2019

Keywords:
Document representation
CVS
iCVS
Document classification
Deep learning
Ontology

ABSTRACT

This paper provides a comprehensive performance analysis of parametric and non-parametric machine learning classifiers including a deep feed-forward multi-layer perceptron (MLP) network on two variants of improved Concept Vector Space (iCVS) model. In the first variant, a weighting scheme enhanced with the notion of concept importance is used to assess weight of ontology concepts. Concept importance shows how important a concept is in an ontology and it is automatically computed by converting the ontology into a graph and then applying one of the Markov based algorithms. In the second variant of iCVS, concepts provided by the ontology and their semantically related terms are used to construct concept vectors in order to represent the document into a semantic vector space.

We conducted various experiments using a variety of machine learning classifiers for three different models of document representation. The first model is a baseline concept vector space (CVS) model that relies on an exact/partial match technique to represent a document into a vector space. The second and third model is an iCVS model that employs an enhanced concept weighting scheme for assessing weights of concepts (variant 1), and the acquisition of terms that are semantically related to concepts of the ontology for semantic document representation (variant 2), respectively. Additionally, a comparison between seven different classifiers is performed for all three models using precision, recall, and F1 score. Results for multiple configurations of deep learning architecture are obtained by varying the number of hidden layers and nodes in each layer, and are compared to those obtained with conventional classifiers. The obtained results show that the classification performance is highly dependent upon the choice of a classifier, and that the Random Forest, Gradient Boosting, and Multilayer Perceptron are among the classifiers that performed rather well for all three models.

1. Introduction

The global Internet population has reached 3.8 billion in 2017 from 3.4 billion the year before, which is 47% of the world's population [1]. According to IBM [2], in 2013 the amount of data produced was 2.5 quintillion when the Internet users were around 2.7 billion only. The number is expected to grow in coming years which means that the amount of data produced will be tremendous. By 2020, it is estimated that around 1.7 MB of data will be created every second for every person on earth.

The penetration of Internet of Things (IoT) and smart gadgets to households and a huge amount of data produced every minute as a result has created a need for better organization and structuring of the data, which according to [3] is mostly unstructured. Despite the computational resources available nowadays, organizing and structuring tremendous amount of data is not a trivial task and without it, finding and extracting useful information from massive Internet resources is a challenge [4]. Nearly 3.87 million Google searches are conducted every minute of the day by the users [1]. Finding relevant information for every query from plethora of resources is a challenging task. For text-based documents, ontology can play a vital role in this regard [5].

An ontology is a data representation techniques that not only help better organize data but also help categorize and classify data objects for easy search and retrieval. Many text document classification approaches widely employ ontologies to classify and organize text-based documents. A text document is generally represented by a vector space model [6]. A vector space model is a feature vector representation constructed by terms/words occurring in a document and their corresponding weights. Each term denotes a dimension in the vector space and it is independent to other terms in the same document. This representation technique is based on string literals and fail to consider order of words and semantic relationships between them i.e. taxonomic and non-taxonomic relations. In order to overcome these issues, a conceptual space document representation emerged as a means that takes advantages of using wide coverage of concepts and

[*] Corresponding author.
E-mail address: zenun.kastrati@ntnu.no (Z. Kastrati).

https://doi.org/10.1016/j.future.2019.02.006

relations provided by ontologies. In a conceptual space representation, a document is represented as a vector comprised of concepts (rather than words) and their weights. Concepts are identified and located in a document through a matching technique which links the terms appearing in that document with the concepts in the ontology. In fact, the link between a term t and a concept c is a mapping denoted by $\langle t, c \rangle$ in which textual description defined in label of t is replaced with textual description defined in label of c. The weights of concepts are defined by counting the occurrences of the concepts within a document i.e. concept relevance. Researchers in [7–12] have widely used concept vector space model for document classification. Even though this approach has proven useful for document classification of many domains, it however has some limitations. Two major limitations of this approach are: (1) it relies on the exact technique in which a document is represented into vector space using concept vectors built by mapping terms occurring in a document with concepts appearing in a ontology, and (2) weighting technique that treats all concepts equally important regardless of where the concepts are depicted in the hierarchy of an ontology [13]. The importance is not equal for all concepts and it depends on relations of concepts with other concepts in the ontology hierarchy. Concepts which have more relations with other concepts are more important than the concepts which have less relations [14].

These limitations are addressed in this paper by proposing an improved concept vector space model in which

1. a weighting technique enhanced with the new concept importance parameter is used to assess weight of ontology concepts. The concept importance in our case is computed automatically by first converting the ontology into an ontology graph and then implementing one of the Markov based algorithms called PageRank. The obtained importance is then aggregated with the concept relevance in order to achieve the final weight of that particular concept.
2. concept vectors used to represent the document into a semantic vector space are constructed by using concepts provided by the ontology through exact technique and by acquiring terms that are related and can be attached to concepts of that ontology.

The rest of the paper is structured as follows. Section 2 describes related work. Section 3 gives an overview of the proposed architecture and presents a detailed description of our proposed concept vector space model. Section 4 describes the concept importance calculation procedure and presents the performance of conventional and deep machine learning classifiers on the INFUSE dataset for classifying funding documents in to five distinct categories. Lastly, Section 5 concludes the paper and gives an insight into the future work.

2. Related work

The field of document classification has attracted a lot of attention in recent years, thereby resulting in a wide variety of approaches. Depending on the vector space document representation model employed there are two main categories of these approaches relevant to the classification task: (1) Keyword based vector space approach, and (2) Concept (ontology) based vector space approach.

The first approach relies on a set of terms (words) extracted from the documents in the dataset. This approach has some limitations as it does not consider the dependency between the terms and it also ignores the order and the syntactic structure of the terms in the documents. To overcome these limitations, concept based vector space approach comes into effect. This approach relies on a set of concepts taken from an ontology to derive the semantic representation of documents. There is some research work in which concepts exploited by ontologies are used for semantic document representation. One example is presented in [15], in which the authors introduced a classification approach that relies on a document representation model constructed using concepts gathered by a domain ontology. In particular, a domain ontology for Health, Safety, and Environment for oil and gas application contexts is used for classifying documents dealing with accidents from the oil and gas industry. An extended version of classification approach given in [15] is presented later in [9]. This extended work proposed a classification approach that employs a semantic document representation model that, besides concepts derived by the ontology, uses a list of semantically related terms. Although the approach presented in this paper is similar to our work, we differ in the way of how we acquire semantically related terms. An extraction technique that relies on semantic and contextual information of terms is used in our approach to find and extract the most semantically related terms instead of n-gram extraction technique used in [9].

Concept vector space approach employs a weighting technique for assessing weight of concepts that relies on the concept relevance as a discriminatory feature for document classification. A drawback of this weighting technique is that it considers all concepts equally regardless of where in the hierarchy the concepts occur. There have been some efforts to find concepts importance depending on the position of concepts where they are depicted in the hierarchy. For instance, researchers in [16] used three different weights for concepts depending on the position where they occur in the ontology hierarchy. The first weight was assigned to concepts which are occurring as classes, second weight for concepts occurring as subclasses and the third weight for concepts occurring as instances. The value of these weights is set empirically through trial and error by conducting experiments. The value of 0.2 is set for concepts which occur as classes, 0.5 for concepts occurring as subclasses and 0.8 when concepts occur as instances.

A slightly different approach of computing weights is implemented in [17,18] where layers of ontology tree are used to represent the position of concepts in the ontology. The weight of each concept is then computed by counting the length of path from the root node to the given concept. The same approach of using layers for calculating weight values of concepts is used in [19]. Path length is also used to compute the weight of concepts but rather than considering all ontology concepts, only the leaf concepts are used. The idea behind this approach is that more general concepts, such as superclasses, are implicitly taken into account through the use of leaf concepts by distributing their weights to all of their subclasses down to the leaf concepts in equal proportion.

The drawback of above presented approaches is that they compute concepts' weight either empirically through trial and error by conducting experiments thus keeping these weights fixed or using the path length. Furthermore, the approach presented in [19] uses only the top-level ontology for computing weights. Our approach uses a Markov based PageRank algorithm to compute the concept importance. The algorithm uses all concepts of ontology and the importance of a concept is computed relative to all other concepts in the ontology.

From classification perspective, studies presented above have not established well the representation of documents which is one of the main aspects that influences the performance of ontology based classification models. Documents are represented as vectors containing relevance of the concepts that are gathered by an ontology by searching only the presence of their lexicalizations

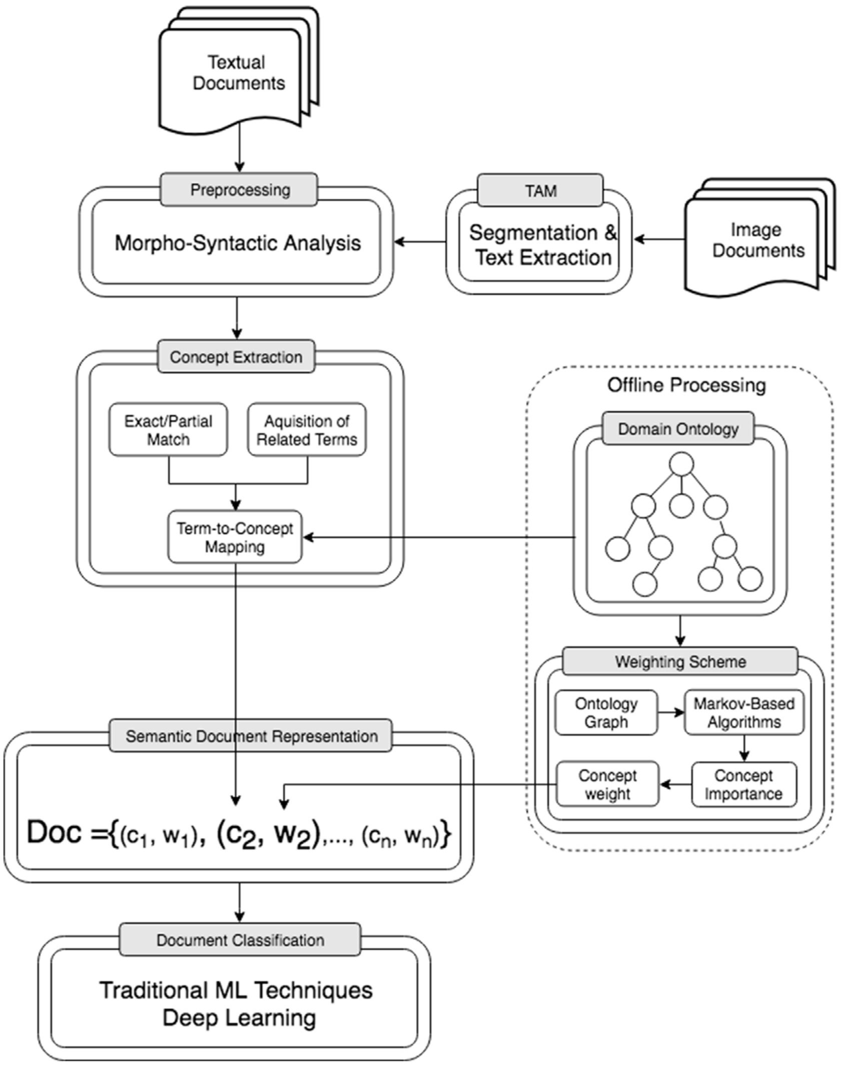


Fig. 1. Architecture of the proposed classification model.

(concept label) in the documents. As a result of this, classification models are limited to capture the whole conceptualization involved in documents.

Another strand of research covers the work related to the use of machine learning approaches for document classification. For instance, the authors in [8] proposed a machine learning based classification approach for understanding sentiment through differentiating good news from bad news. This is achieved using a vector space document representations learned by deep learning and convolutional neural networks with a test accuracy of 85%. Another example of using convolutional recurrent deep learning model for classification is proposed in [7]. This approach is similar to our work but our focus is on classification of documents instead of sentences and we use feature vectors constructed by concepts derived by an ontology.

3. Architecture of the proposed model

The main goal of the proposed model shown in Fig. 1 is classification of image and textual documents using an improved concept vector space which relies on semantically rich document representations and an enhanced concept weighting scheme. An image document in our case is a movie frame containing handwritten lecture notes on the chalkboard extracted from a lecture video employing image processing techniques while the textual documents are financial documents that are stored in pdf format. The model consists of seven main modules that are described in the following subsections.

3.1. Text analysis module - TAM

The input of proposed classification model is a collection of documents that can be stored either as unstructured textual data or image. If the input is a document image, it initially goes through a text analysis module called TAM to extract texts from that image.

TAM module itself consists of three steps and preprocessing is the very first one which ensures that the image has a readable text. The readable quality of a text in a document image is mostly affected by blocking and blurring artefacts as a result of compression and denoising. These readable text issues are avoided by using a metric designed for evaluating text quality called a reference free perceptual quality metric (RF-PQM) [20]. The image is then converted into binary format using Otsu technique [21] and text regions are localized using a 4-connected component based labelling approach as illustrated in Fig. 2.

The next step of TAM module is segmentation and extraction of text lines from the connected components obtained as blobs after localization followed by extraction of words using vertical 1-D projection histogram. We assume that the text documents

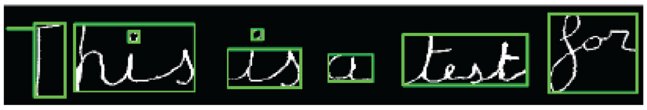


Fig. 2. Labelling approach using 4 connected-components.

obtained at this stage are correct since the evaluation of TAM itself is beyond the scope of this paper. Readers are therefore advised to refer to [22] and [23] for full details on the TAM module.

3.2. *Preprocessing*

This module takes as input text documents extracted from the image documents in the TAM module and/or a collection of documents stored in unstructured textual formats, e.g. Word, PDF, Powerpoint slides, etc. These text documents undergo preprocessing steps including morpho-syntactic analysis. The first preprocessing step involved is tokenization in which the text is split in small pieces known as tokens. Next, stop words and duplicate words are removed, and finally a stemming is performed to normalize the retrieved words.

The output of this module is a collection of documents composed of plain text with no semantics associated to them and it is linked directly to the concept extraction module in order to embed semantics into those text documents.

3.3. *Concept extraction*

Concept extraction module concerns with construction of feature vectors. A feature vector is an n-dimensional vector comprised of concepts provided by domain ontologies so as to make a move from the keyword-based vector representation towards the semantic-based vector representation. To achieve this step toward semantic representation, we primarily need to associate terms extracted from documents with concepts of the ontology. Terms are located and extracted from documents using a Lucene inverted indexing technique which generates a list of all unique terms that occur in any document and a set of documents in which these terms occur. The extracted terms are stemmed using a stemming method. Further, noisy terms, i.e terms with single character, are removed from the list of extracted terms. The extracted terms are associated with the concepts of the ontology using: (1) the matching method in which terms appearing in a document are mapped with the relevant concepts from the domain ontology, and (2) acquisition of relevant terminology that is semantically related and can be attached to concepts of that domain ontology.

The matching method [12] follows the idea of searching for concepts in the domain ontology that have labels matching either partially or exactly/fully with a term occurring in a document. To put it simply, each term identified and located in a document is searched in the domain ontology, and if an instance term matches its concept label than term is replaced with the concept. Concept labels are considered all lexical entries and lexical variations contained in a concept. The obtained concepts are used to construct concept vectors. An exact match is the case where a concept label is identical with a instance term occurring in the document. A partial match is the case when concept label contains a term occurring in the document. The exact and partial match is formally defined as following.

Definition 1. Let *Ont* be the domain ontology and let *D* be the dataset composed of documents of this given domain. Let $Doc \in D$ be a document defined by a finite set of terms $Doc = \{t_1, t_2, \dots, t_i\}$. Mapping of term $t_i \in Doc$ into concept $c_j \in Ont$ is defined as:

$$EM(t_i, c_j) = \begin{cases} 1, & \textit{if label } (c_j) = t_i \\ 0, & \textit{if label } (c_j) \neq t_i \end{cases}$$

$$PM(t_i, c_j) = \begin{cases} 1, & \textit{if label } (c_j) \textit{contains } t_i \\ 0, & \textit{if label } (c_j) \textit{does not contain } t_i \end{cases}$$

where, *EM* and *PM* denote exact match and partial match, respectively.

If $EM(t_i, c_j) = 1$, it means that term t_i and concept label c_j are identical, then term t_i is replaced with concept c_j. For example, for a concept in the ontology such as *Organization* or *Call* as shown in Fig. 3, there exists an identical term that appears in the document.

If $PM(t_i, c_j) = 1$, it means that term t_i is part of concept label c_j, then term t_i is replaced with concept c_j. For example, the *ProjectFunding* compound ontology concept shown in Fig. 3, contains terms that appears in the document such as *Project* and/or *Funding*.

Extraction of concepts through acquisition of relevant terminology that is related and can be attached to ontology concepts is a more complex task which is achieved through exploitation of both contextual and semantic information of terms occurring in a document.

Contextual information of a term is defined by its surrounding words and it is computed using Eq. (1).

$$Context(t_i, t_j) = \frac{t_i \cdot t_j}{\|t_i\| \|t_j\|} \tag{1}$$

The vectors, t_i and t_j, are composed of values derived by three statistical features, namely, term frequency, term font types, and term font sizes, respectively. Different font types, i.e. *bold*, *italic*, *underline*, and font sizes, i.e. *title*, *level 1*, *level 2*, are introduced to derive the context. In our case, values of these statistical features are extracted from input pdf documents using Apache PDFBox library, that is, an open source Java library which allows creation of new pdf documents, manipulation of existing documents and the extraction of content from documents.

Semantic information of a term is calculated using a semantic similarity measure based on the English lexical database WordNet. Wu&Palmer similarity measure [24] is employed to compute a semantic score (Eq. (2)) for all possible pairs of terms t_i and t_j occurring in a document.

$$Semantic(t_i, t_j) = \frac{2 * depth(lcs)}{depth(t_i) + depth(t_j)} \tag{2}$$

Parameter, *depth(lcs)* shows the least common subsumer of terms t_i and t_j, and parameters $depth(t_i)$ and $depth(t_j)$ show the path's depth of terms t_i and t_j, in the WordNet.

Combination of contextual and semantic information gives an aggregated score as shown in Eq. (3).

$$AggregatedScore(t_i, t_j) = \lambda * Context(t_i, t_j) + (1 - \lambda) * Semantic(t_i, t_j) \tag{3}$$

where, λ is set to 0.5 showing an equal contribution of context and semantic components on the aggregated score.

Aggregated score through a rank cut-off method is used to acquire terms that are related to concepts of the ontology. More concretely, terms that are above the specified threshold (top-N) are considered to be the relevant terms.

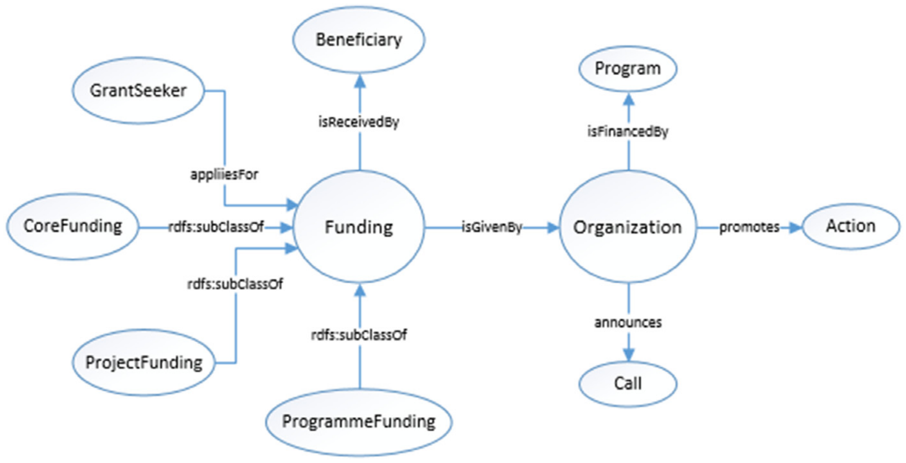


Fig. 3. A part of INFUSE ontology graph.

3.4. Domain ontology

This module covers domain ontology which interfaces Term-to-Concept mapper component in the concept extraction module and the weighting scheme module. A domain ontology is a data model which represents concepts and relations between them in a given domain. An ontology structure is formally represented by a 5-tuple [25], as shown in Eq. (4).

$$Ont := (C, R, H_C, rel, A) \tag{4}$$

where,

- C is a set of concepts, e.g. *Funding*, *Call*;
- R is a set of relations, e.g. *announces*, *promotes*;
- H_C is a hierarchy or taxonomy of concepts with multiple inheritance, e.g. *ProgrammeFunding is a Funding* and *FinanceProgramme is a ProgrammeFunding*;
- rel is a set of non-taxonomic relations which are described by their domain and range restrictions, e.g. *isReceivedBy, appliesFor*;
- A is a set of ontology axioms, expressed in an appropriate logical language, which describe additional constraints;

The ontology definition shown in Eq. (4) can be domain specific by defining a lexicon which is a 3-tuple $Lex := (\mathcal{L}, \mathcal{F}, \mathcal{G})$ consisting of a set of lexical entries $\mathcal{L}$ for concepts and relations, and two sets $\mathcal{F}$ and $\mathcal{G}$ that link concepts and relations with their lexical entries.

3.5. Weighting scheme

The weight of a concept is a numeric value which is assigned to each concept in order to assess its power in distinguishing a particular document from others. A technique used to compute the weight of concepts is known as concept weighting scheme. There exist various weighting schemes that typically rely on the relevance of concepts reflected by frequency of occurrences of concept's lexicalizations within a document. In this module we present an enhanced concept weighting scheme which besides concept relevance, introduces a new parameter called concept importance that reflects the contribution of a concept in the ontology. Concept importance is processed offline and it involves the following steps: (1) mapping the domain ontology into an ontology graph, (2) applying Markov based algorithms, and (3) calculation of concept importance and aggregation with concept relevance.

The first and the foremost step of this module is to convert the domain ontology described in Section 3.4 into an ontology graph for calculating concept importance. To achieve this, we adopt a model where the ontology is represented as a directed acyclic graph. The modelling is an equivalent mapping which means that an ontology concept is mapped into a graph vertex and an ontology relation into a graph edge which connects two vertices. The formal definition of this graph, known as ontology graph, is given as follows.

Definition 2. Given a domain ontology Ont, the ontology graph $G = \{V, E, f\}$ of Ont is a directed acyclic graph, where V is a finite set of vertices mapped from concepts in Ont, E is a finite set of edge labels mapped from relations in Ont, and f is a function from E to $V \times V$.

In Fig. 3, we present part of the INFUSE ontology graph which consists of a subset of concepts and relations from the funding domain. The details of the INFUSE ontology are given in Section 4.

In the semantic web, a formal syntax for defining ontologies is Web Ontology Language (OWL) and Resource Description Framework (RDF) Schema. These languages represent the ontology as a set of Subject–Predicate–Object (*SPO*) expressions known as RDF triples. The set of RDF triples is known as RDF graph where subject is the source vertex and object is the destination vertex, and predicate is a directed edge label which links those two vertices. The formal definition of RDF graph is given as following.

Definition 3. Given a set of RDF triples T, the RDF graph $G = \{V, E, f\}$ of T is a directed acyclic graph, where V is a finite set of vertices (subjects and objects) in G defined as $V = \{v_u : u \in (S(T) \cup P(T))\}$, E is a finite set of edge labels (predicates) in G defined as $E = \{e_{SPO} : SPO \in T\}$, f is a function linking subject S to an object O by an edge E defined as $f = \{f_P : f_P = V_S \to V_O, V_S, V_O \in T\}$

The ontology graph and RDF graph are not the same for a given ontology. The difference is that a relation in an ontology graph is defined as a vertex in the RDF graph. For example, relation *isReceived* in ontology graph shown in Fig. 3 is represented as a vertex in RDF graph, as shown in Fig. 4. In other words, a relation in RDF graph is a link between a subject denoted by *rdfs:domain* property and an object denoted by *rdfs:range* property as given in Definition 3.

The next step is computation of the importance of vertices of the graph using an adoption of the Markov based algorithms. The graph can be either ontology graph or RDF graph as defined

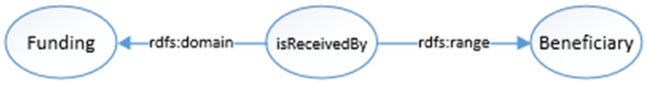


Fig. 4. An example RDF graph representation.

above. The idea behind Markov based algorithms is representing the graph as a stochastic process, more concretely as a first-order Markov chain where the importance for a given vertex is defined as the fraction of time spent traversing that vertex for an infinitely long time in a random walk over the vertices. The probability of transitioning from a vertex i to a vertex j is only dependent on the vertex i and not on the path to arrive at vertex j. This property, known as the Markov property, enables the transition probabilities to be represented as a stochastic matrix with non-negative entries and the maximum probability of 1.

In this paper, we use PageRank [26] algorithm as one of the most well known and successful example of Markov based algorithms [27].

A simplified principle of work of PageRank algorithm is as follows. It initially defines the importance of a vertex i as given in Eq. (5).

$$PR(i) = \sum_{j \in V_i} \frac{PR(j)}{Outdegree(j)} \tag{5}$$

where, $PR(j)$ is the importance of vertex j, V_i is the set of vertices that links to vertex i, and $Outdegree(j)$ is the number of vertices that have outlinks from vertex j.

As we can see from Eq. (5), the PageRank is an iterative algorithm. It assigns an initial importance to a vertex i as shown in Eq. (6).

$$PR^{(0)}(i) = \frac{1}{N} \tag{6}$$

where, N is the total number of vertices in the graph. Then PageRank iterates as per Eq. (7) and continues to iterate until a convergence criterion is satisfied.

$$PR^{(k+1)}(i) = \sum_{j \in V_i} \frac{PR^{(k)}(j)}{Outdegree(j)} \tag{7}$$

The process can also be defined using the matrix notation. Let M be the square, stochastic transition probabilities matrix corresponding to the directed graph G, and $Imp(k)$ is the Importance vector at the k^{th} iteration. Then the computation of one iteration corresponds to the matrix–vector multiplication as shown in Eq. (8).

$$PR^{(k+1)} = M * PR^{(k)} \tag{8}$$

The entry of transition probability matrix M, for a vertex j which links to vertex i, is defined using Eq. (9).

$$p_{i,j} = \begin{cases} \frac{1}{Outdegree(j)}, & \textit{if there is a link from } j \textit{ to } i \\ 0, & \textit{otherwise} \end{cases} \tag{9}$$

There are two properties that are necessary to be satisfied in order for a Markov based algorithm to converge. It should be aperiodic and irreducible [28]. The transition probability matrix M is a stochastic matrix with probability 1 and this makes the PageRank algorithm aperiodic. The PageRank algorithm is not irreducible due to the definition given in Eq. (9), where some of the transition probabilities in matrix M may be 0. This does not meet the criteria of irreducibility property which requires the transition probabilities to be greater than 0.

To make the PageRank algorithm irreducible in order to converge, a damp factor $1-\alpha$ is introduced. As a result of this, a new transition probability matrix M^* is defined where a complete set of outgoing edges with probability α/N are added to all vertices in graph. The definition of matrix M^* is given in Eq. (10).

Table 1
An example of building concept vector space.

Doc	GeographicalArea			Applicant		
	Imp	Rel	w	Imp	Rel	w
d1	0.130	0.797	0.104	0.020	0.797	0.016
d2	0.130	0.624	0.081	0.020	0.624	0.012
d3	0.130	0.000	0.000	0.020	0.860	0.017

$$M^* = (1-\alpha)M + \alpha \left[\frac{1}{N}\right]_{N \times N} \tag{10}$$

The damp factor besides enabling the PageRank algorithm to converge also overcomes the problem of rank sinks [28].

Replacing M^* with M in Eq. (8), the PageRank algorithm is defined as given in Eq. (11).

$$PR^{(k+1)} = (1-\alpha)M \times Pr^{(k)} + \alpha \left[\frac{1}{N}\right]_{N \times N} \tag{11}$$

Finally, concept importance is defined as given in Eq. (12).

$$Imp(c_i) = PR^{(k+1)} \tag{12}$$

The final step of this module is aggregation of concept importance and concept relevance to compute weight of concepts. The value of a concept weight is in the range of [0,1] because both concept importance and concept relevance are normalized.

$$w(c_i) = Imp(c_i) \times Rel(c_i) \tag{13}$$

Concept importance Imp is computed using Eq. (12) described above, while concept relevance Rel is computed using Eq. (14).

$$Rel(c_i) = \sum_{i=1}^{m} Freq(c_i) \tag{14}$$

where, $Freq(c_i)$ is the frequency of occurrences of lexicalizations of concept c_i in the document to be classified.

3.6. Document representation

The output of both modules, concept extraction and weighting scheme, will serve as an input to semantic document representation module for representing a document. More concretely, concepts obtained from concept extraction module and their weights computed through weighting scheme module are used to represent a document in a vector space as defined in Eq. (15).

$$Doc = \{(c_1, w_1), (c_2, w_2), (c_3, w_3), \ldots, (c_i, w_i)\} \tag{15}$$

where c_i is the i^{th} concept obtained from concept extraction module and w_n is its weight computed from weighting scheme module.

Table 1 illustrates an example of semantic document representation through a vector space that is constructed by using concepts (*GeographicalArea* and *Applicant*) and their weights composed of two components, Importance (Imp) and Relevance (Rel), as described in Section 3.5.

3.7. Document classification

The last module of proposed model deals with classification of documents into appropriate categories using conventional machine learning classifiers and deep learning. In essence, a document represented via concept vector space is fed into the classifier to build a prediction model that can be used to classify a new unseen document.

Table 2
Concept importance for the top ten concepts of the INFUSE ontology.

No	Concept	Concept importance
1	Coverage	0.20
2	GeographicalArea	0.13
3	Topic	0.11
4	County	0.07
5	Participant	0.06
6	Programme	0.05
7	Organization	0.05
8	Funding	0.05
9	Applicant	0.04
10	Candidate	0.04

4. Results and analysis

This section describes the calculation of concept importance of a real-world ontology. It also gives a description of the dataset used to perform the experiments for demonstrating the applicability of our proposed document representation models. Finally, it provides a thorough comparison of document classification results achieved using both conventional machine learning techniques and deep networks.

4.1. Concept importance calculation

A real-world domain ontology called INFUSE ontology is used for computing concept importance. This ontology is developed as part of the INFUSE[1] project and it comes from the funding domain. It is composed of 85 concepts, e.g. *Funding*, *GrantSeeker* and 18 object properties, e.g. *isGivenBy*, *appliesFor*, that connect these concepts. A part of INFUSE domain ontology represented as an ontology graph is shown in Fig. 3.

To convert the ontology into an ontology graph and compute the concept importance, we have used the RDF rank algorithm. This algorithm is part of the extensions module of GraphDB [29] and it computes the importance for every vertex in the entire RDF graph. Table 2 shows the concept importance values of the top ten concepts of the INFUSE ontology. The concept importance is a floating point number with values varying between 0 and 1.

Fig. 5 shows the concept importance values in ranking order after having computed them for all the concepts of the INFUSE ontology. As can be seen from the chart diagram, the concept importance is different for different concepts, varying from 0.2–0.02 for almost half of the concepts set, while for the rest of the concepts it is 0.01. These findings confirm the idea that the contribution of ontology concepts in terms of concepts' discriminating power is different and thus some concepts are more important than the others with respect to document classification.

4.2. Performance evaluation of baseline CVS and iCVS

In order to demonstrate the general applicability of our proposed classification model and to validate its effectiveness, extensive experiments using various classifiers are conducted on the INFUSE dataset.

The INFUSE dataset consists of 467 grant documents that had been collected and classified into 5 categories by field experts as part of the INFUSE project. The dataset is split randomly, in which 70% of the documents are used to build the classifier and the remaining 30% to test the performance of the model. The number of documents in each category varied widely, ranging from the Society category which contains 165 documents to the Music category which contains only 14 documents. Table 3 shows five categories along with the number of training and testing documents in each category.

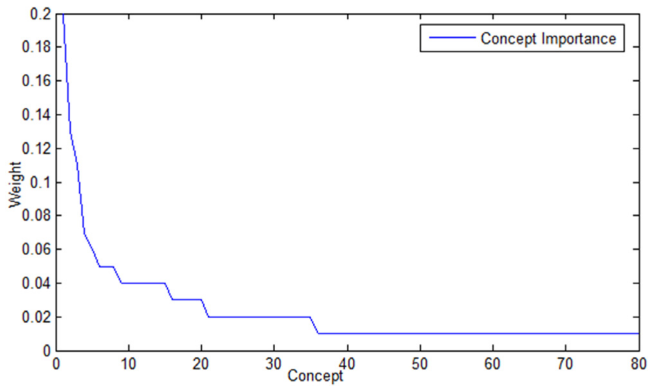


Fig. 5. Concept importance for all concepts of the INFUSE ontology.

Table 3
Dataset size.

No	Category	# Train	# Test	Total
1	Culture	102	44	146
2	Health	73	32	105
3	Music	10	4	14
4	Society	115	50	165
5	Sportssociety	26	11	37
6	Total	326	141	467

Parametric and nonparametric machine learning techniques are used for experimenting. A parametric machine learning technique assumes that the data can be parameterized by a fixed number of parameters. In essence, the statistical model of parametric techniques is specified by a simplified function through two types of distributions, namely, the class prior probability, and the class conditional probability density function (posterior) for each dimension. On the contrary, a nonparametric machine learning technique assumes no prior parameterized knowledge about the underlying probability density function and the classification uses the information provided by training samples alone.

Naive Bayes is a parametric machine learning technique applied for classification in this paper, while nonparametric techniques applied in this paper include Decision Tree and Random Forest. We also have chosen to use Support Vector Machine (SVM) for classification that can be either parametric or non-parametric technique. Linear Support Vector Machine contains a fixed size of parameters represented by the weight coefficient and thus it belongs to the parametric techniques. On the other side, Non-linear Support Vector Machine is a non-parametric technique and Radial Basis Function Kernel Support Vector Machine, known as RBF Kernel SVM, is a typical example of this family. In addition, we have applied two boosting techniques, namely Gradient Boosting and Ada Boosting, which grant power of ensemble classifiers that generate multiple predictions and majority voting among the individual classifiers.

Additionally, a Multilayer Perceptron (MLP) is used in this study. An MLP is a feed-forward Artificial Neural Network (ANN). The artificial neurons in the network compute a weighted sum of its inputs x_i, adds a bias b, and applies an activation function. A simple ANN is represented as: $y = f(wx_i + b)$, where w is the weigh and f is the activation function. Most commonly used activation functions are *sigmoid*, which is $\sigma(z) = 1/(1+e^{-z})$ and *rectified linear units* which is $ReLU(z) = max(0, z)$. The weight

[1] https://www.eurostars-eureka.eu/project/id/7141.

Table 4
Performance of conventional ML techniques using baseline CVS.

Technique	Precision (%)	Recall (%)	F1 (%)
Naive Bayes	67.24	60.99	61.90
Decision Tree	66.10	66.40	65.50
Random Forest	77.69	77.30	77.25
SVM	81.73	77.30	78.85
Gradient Boosting	82.99	82.26	82.58
Ada Boosting	58.61	53.90	54.69

and bias terms are estimated by training the network on the observable data to minimize the loss using cross-entropy or mean square error. In an MLP, the neurons are structured into layers. These layers are fully-connected which implies that every neuron in one layer is connected to every neuron in the adjacent layer. The input and the output layers are the visible layers in the network while a network may contain multiple hidden layers. Normally, a network containing more than one hidden layer is known as a deep neural network.

The standard information retrieval measures such as precision, recall and F1 measure, are used to evaluate the performance of the document classification. Precision is the number of documents which are classified correctly with respect to all classified documents. It is given as: $tp/(tp+fp)$. Recall is the number of classified documents with respect to the total number of documents in the dataset. Recall is defined as: $tp/(tp+fn)$, where tp, tn, and fn are true positive, true negative, and false negative samples. F1 measure is the harmonic mean of precision and recall and it is defined as: $2((precision * recall)/(precision + recall))$.

Best results are obtained on the conventional machine learning techniques for following configurations. For the Bayesian classifier, a Gaussian NB is used whereas for SVM, a radial basis function (RBF) kernel SVM is used. A value of 0.001 is used for *gamma* which describes how much influence a single training sample has, and a maximum value is set for the regularization parameter c. The depth of the tree for RF classifier is set to 10 which gave best results. For all other parameters of the classifiers, default configurations are used. For deep learning based MLP architecture, multiple simulations consisting of $L \times N$ are carried out by varying the number of hidden layers L and the number of neurons N in each layer, where $L = \{3, 5, 7\}$, and $N = \{64, 128, 256, 512, 1024\}$. Fig. 6 shows the total number of trainable parameters for a 5-hidden layer MLP containing 1024 neurons in each layer. The input to the network shown is 323 size concept vector for iCVS variant 2. *Relu* is applied as the activation function, *adam* is used as the optimizer, while the learning rate α is set to $1e^{-3}$. A *softmax* function is applied at the last layer to convert the likelihood of a test sample belonging to one of the 5 classes.

Three different models of vector space document representation are used to test the classifiers. In the first model called baseline CVS, we conducted a document classification experiment on the INFUSE dataset in which an exact/partial match technique is employed to match term occurring in a document with relevant concepts of the ontology to build concept vectors for representing documents into vector space. Precision, recall, and F1 results obtained from six conventional Machine Learning techniques and a deep MLP with different number of hidden layers and neurons are shown in Tables 4 and 5, respectively. As can be seen from the results, Gradient Boosting classifier shows the best performance compared to other conventional classifier achieving a 82.58% of weighted F1 score. On the other hand, MLP with 3 hidden layers and 1024 neurons in each layer outperforms other deep network achieving an F1 score of 80.02%.

In the second experiment, we performed document classification using the same classifiers on the same corpus of documents from the INFUSE dataset, but employing the second model of document representation. The second model called iCVS variant 1 is an enhanced concept weighting scheme that is used for assessing weight of concepts of the ontology. Six different conventional Machine Learning techniques, and a Multilayer Perceptron with different number of hidden layers and different number of neurons per layer, are used for classification and the obtained results are shown in Tables 6 and 7, respectively. As with baseline CVS model, the obtained results using iCVS variant 1 show that Gradient Boosting classifier achieved the highest improvement compared to other conventional machine learning and deep learning techniques. In the context of deep networks, the best performance is achieved by an MLP architecture with 7 hidden layers and 256 neurons per layer with an F1 score of 76.64%,.

iCVS variant 2 model is also evaluated in a similar fashion. In this model, concept vectors for representing documents into vector space are build through acquisition of new terms that are semantically related and can be attached to concepts of the ontology. In our case, for each concept of the INFUSE ontology we used only the top-5 terms found as relevant in terms of

Table 5
Performance of MLP using baseline CVS.

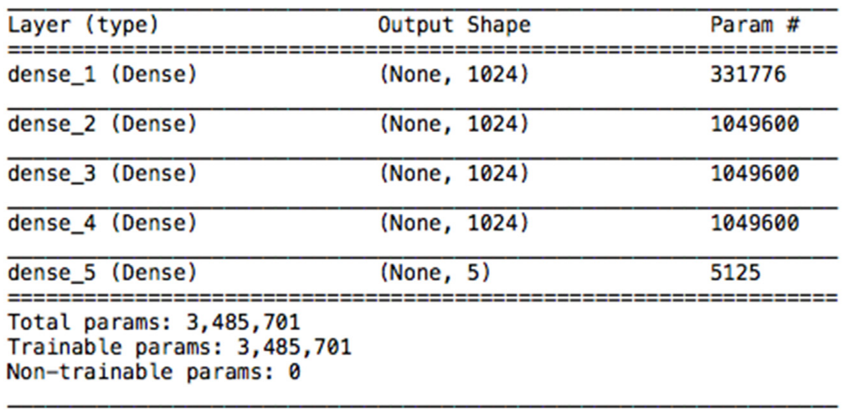
# of hidden layers	# of neurons	Precision (%)	Recall (%)	F1 (%)
3	64	79.32	78.72	78.47
	128	77.80	78.01	77.89
	256	77.05	77.30	77.08
	512	79.75	79.43	79.07
	1024	80.13	80.14	80.02
5	64	78.11	78.01	77.50
	128	78.29	78.01	77.96
	256	75.21	74.46	74.36
	512	77.21	76.59	76.64
	1024	77.87	77.30	77.24
7	64	77.99	78.01	77.77
	128	77.93	77.30	77.40
	256	76.53	75.58	75.89
	512	75.00	73.75	73.92
	1024	78.73	76.59	76.90

Table 6
Performance of conventional ML techniques using iCVS variant 1.

Technique	Precision (%)	Recall (%)	F1 (%)
Naive Bayes	66.63	53.90	57.73
Decision Tree	69.10	70.00	68.80
Random Forest	84.54	80.85	82.07
SVM	66.65	53.19	56.64
Gradient Boosting	83.06	81.56	82.14
Ada Boosting	61.72	60.28	60.33

Table 7
Performance of MLP using iCVS variant 1.

# of hidden layers	# of neurons	Precision (%)	Recall (%)	F1 (%)
3	64	72.84	73.04	72.77
	128	67.40	69.50	68.22
	256	71.86	70.92	71.29
	512	73.69	73.75	73.55
	1024	73.35	73.04	72.81
5	64	70.33	69.50	69.53
	128	72.65	73.04	72.77
	256	72.16	72.34	72.16
	512	68.30	68.79	68.23
	1024	73.14	73.04	72.82
7	64	66.55	68.08	66.46
	128	67.79	69.50	68.30
	256	76.82	76.59	76.64
	512	77.10	75.17	75.87
	1024	73.48	73.75	73.47



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 1024)              331776
_________________________________________________________________
dense_2 (Dense)              (None, 1024)              1049600
_________________________________________________________________
dense_3 (Dense)              (None, 1024)              1049600
_________________________________________________________________
dense_4 (Dense)              (None, 1024)              1049600
_________________________________________________________________
dense_5 (Dense)              (None, 5)                 5125
=================================================================
Total params: 3,485,701
Trainable params: 3,485,701
Non-trainable params: 0
_________________________________________________________________
```

Fig. 6. Model summary for a 5-hidden layer MLP architecture for 323 concept input vector size with 1024 neurons.

Table 8
Performance of conventional ML techniques using iCVS variant 2.

Technique	Precision (%)	Recall (%)	F1 (%)
Naive Bayes	67.02	65.95	65.28
Decision Tree	79.20	77.90	76.70
Random Forest	77.04	74.46	75.06
SVM	85.66	83.68	84.11
Gradient Boosting	84.35	83.68	83.96
Ada Boosting	69.79	60.99	62.56

Table 9
Performance of MLP using iCVS variant 2.

# of hidden layers	# of neurons	Precision (%)	Recall (%)	F1 (%)
3	64	85.05	85.10	84.98
	128	80.12	80.14	79.55
	256	79.04	79.43	78.79
	512	81.47	81.56	81.29
	1024	81.68	82.26	81.80
5	64	80.11	80.85	80.11
	128	78.07	79.43	78.06
	256	80.76	80.85	80.34
	512	78.79	78.72	77.82
	1024	78.42	79.43	78.58
7	64	77.68	78.01	77.21
	128	80.76	80.85	80.47
	256	77.50	77.30	77.17
	512	82.99	83.68	83.07
	1024	81.69	82.26	81.57

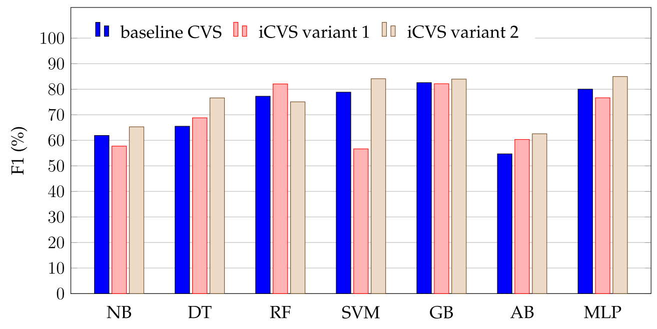


Fig. 7. F1 measure of different classifiers using exact/partial match (baseline CVS), enhanced weighting scheme (iCVS variant 1), and acquisition of related terms (iCVS variant 2).

relatedness. For example, terms *fund*, *amount*, *part*, *subsistence*, and *grant*, are the top-5 terms that are found to be the most semantically related terms with ontology concept *funding*. The performance of document classification, in terms of precision, recall and F1 measure, achieved by six conventional Machine Learning techniques and a Multilayer Perceptron with different number of hidden layers and neurons, is given in Tables 8 and 9, respectively. As can be seen from the results shown in Tables 8 and 9, the best performing classifier is an MLP having three hidden layers and 64 neurons in each layer with an F1 score of 84.98% which is slightly better than SVM with an F1 score of 84.11%.

A side by side comparison is illustrated in Fig. 7 for three models. The figure presents a complete picture of the performance of conventional machine learning and deep learning techniques on the INFUSE dataset for the proposed models. The bar chart shows the weighted F1 score obtained by conventional machine learning, namely Naive Bayes (NB), Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM), Gradient Boosting (GB), and Ada Boosting (AD), and a Multilayer Perceptron (MLP) with 3 hidden layers and 64 neurons per layer, tested on three different models of document representation.

As can be seen from the results shown in Fig. 7, a higher weighted classification F1 score is achieved by all classifiers using iCVS variant 2. An exception is Random Forest that gives slightly worse classification performance than other classifiers. Random Forest is an ensemble method that employs the same decision tree classifier on different training sets generated using the bootstrap sampling method. In a bootstrap sampling, a new training set is created by taking data from the original training set, thus some data may be used several times to construct the forest and others not at all. This may be one of the reasons that this classifier performs worse.

It is also interesting to note from Fig. 7 that in general MLP classifier outperforms all conventional machine learning classifiers achieving a classification F1 score of 84.98%. On the other hand, the worst performance is shown by Naive Bayes classifier which may have happened due to the imbalanced classes of the INFUSE dataset. Imbalanced classes may result in biasing of the classifier towards the majority of the class and thus the performance of Naive Bayes classifier can quickly turn poor.

An interesting fact that also can be observed from the bar chart shown in Fig. 7 is that iCVS variant 1 model has different impact on the performance of classifiers. While nonparametric and boosting machine learning techniques demonstrate a positive impact on document classification using an iCVS variant 1, parametric and MLP show a negative impact on classification performance giving worse accuracy.

5. Conclusion and future work

In this paper, we have investigated and analysed the document classification performance using a concept vector space model improved with new concept weighting scheme, and semantic document representation. Concept weighting scheme is enhanced with new parameter that takes into account the importance of ontology concepts. Concept importance is computed automatically and this is achieved by converting the ontology into a graph and then employing the PageRank algorithm on it. Importance of an ontology concept is then aggregated with concept relevance which is computed using the frequency of appearances of a concept in the document. A semantic representation of document is achieved using concepts derived from ontology through matching technique and acquisition of new terms that can be semantically related with ontology concepts.

We conducted various document classification experiments on three models of document representation i.e. baseline CVS model and iCVS model with two variants. Additionally, a comparison between seven different classifiers is performed for all three models using precision, recall, and F1 score. For all three models, Random Forest, Gradient Boosting, and Multilayer Perceptron, performed rather well. Furthermore, a thorough investigation is carried out to evaluate the performance of MLP by varying the number of hidden layers and the number of neurons in each layer. A three hidden layer MLP with 64 neurons achieves higher classification performance compared to other architecture configurations.

Generally, iCVS variant 1 employing an enhanced weighting scheme used for assessing weights of concepts did not add much to the overall performance except for Random Forest which gave better results employing baseline CVS and iCVS variant 2 with an F1 score of just over 81%. Our findings showed that adding more concepts to ontology improves the classification performance by 4.78 percentage point on average in all cases, however, it is computationally expensive due to a large number of feature vectors. The classification performance is also highly dependent upon the choice of a classifier and we can achieve the same performance on the iCVS model (variant 1 and variant 2) with Random Forest and Gradient Boosting classifier.

Investigation and analysis of classification performance is done on real-world ontology and dataset consisting a small number of documents, so in future work we plan to conduct a performance analysis in a large-scale dataset. We also plan to implement and test other Markov based algorithms for computing concept importance as fundamental part of concept weighting scheme and compare those techniques with the PageRank algorithm.

Furthermore, the primary focus of our study was addressing two major concept vectors limitations namely exact matching and weighting scheme by proposing an improved concept vector space model. However, our proposed approach does not handle another concept vectors limitation which is ontological relationships. Future studies on the current topic are therefore suggested in order to establish representation of documents in which concept vectors can be redefined to consider the various relationships that exist in an ontology.

Acknowledgment

The authors would like to thank Cristina Marco from the INFUSE project for providing the domain ontology and the dataset used in this paper.

References

[1] DOMO, Data never sleeps 6.0: how much data is generated every minute? 2018, https://www.domo.com/learn/data-never-sleeps-6. (Accessed 18 June 2018).

[2] R. Jacobson, 2.5 quintillion bytes of data created every day. how does cpg & retail manage it? 2018, https://www.ibm.com/blogs/insights-on-business/consumer-products/2-5-quintillion-bytes-of-data-created-every-day-how-does-cpg-retail-manage-it/. (Accessed 2018-07-20).

[3] P. Raghavan, Extracting and exploiting structure in text search, in: SIGMOD Conference, ACM, 2003, p. 635.

[4] A.-A.R. Al-Azmi, Data, text, and web mining for business intelligence: a survey, Intl. J. Data Mining Knowl. Manag. Process 3 (2) (2013) 1–26.

[5] S. Khan, M. Safyan, Semantic matching in hierarchical ontologies, J. King Saud Univ. 26 (3) (2014) 247–257.

[6] M. Keikha, A. Khonsari, F. Oroumchian, Rich document rrepresentation and classification: an analysis, Knowl.-Based Syst. 22 (1) (2009) 67–71.

[7] A. Hassan, A. Mahmood, Convolutional recurrent deep learning model for sentence classification, IEEE Access 6 (2018) 13949–13957.

[8] U. Reshma, B. Ganesh, M. Kale, P. Mankame, G. Kulkarni, Deep learning for digital text analytics: sentiment analysis, CoRR abs/1804.03673 (2018).

[9] N. Sanchez-Pi, L. Marti, A.C.B. Garcia, Improving ontology-based text classification: an occupational health and security application, J. Appl. Log. 17 (2016) 48–58.

[10] C. Bratsas, V. Koutkias, E. Kaimakamis, P. Bamidis, N. Maglaveras, Ontology based vector space model and fuzzy query expansion to retrieve knowledge on medical computational problem solutions, in: Proceedings of the 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, IEEE, 2007, pp. 3794–3797.

[11] P. Castells, M. Fernandez, D. Vallet, An adaptation of the vector space model for ontology based information retrieval, IEEE Trans. Knowl. Data Eng. 19 (2) (2007) 261–272.

[12] S. Deng, H. Peng, Document classification based on support vector machine using a concept vector model, in: Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence, IEEE, 2006, pp. 473–476.

[13] Z. Kastrati, A.S. Imran, S.Y. Yayilgan, An improved concept vector space model for ontology based classification, in: 2015 11th International Conference on Signal-Image Technology & Internet-Based Systems, SITIS, IEEE, 2015, pp. 240–245.

[14] G. Wu, J. Li, L. Feng, K. Wang, Identifying potentially important concepts and relations in an ontology, in: Proceedings of the 7th International Conference on The Semantic Web, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 33–49.

[15] N. Sanchez-Pi, L. Marti, A.C.B. Garcia, Text classification techniques in oil industry applications, in: Proceedings of the International Joint Conference SOCO'13-CISIS'13-ICEUTE'13, Springer International Publishing, 2014, pp. 211–220.

[16] X. quan Yang, N. Sun, Y. Zhang, D. run Kong, General framework for text classification based on domain ontology, in: Proceedings of the 3rd International Workshop on Semantic Media Adaptation and Personalization, IEEE, 2008, pp. 147–152.

[17] J. Fang, L. Guo, X. Wang, N. Yang, Ontology-based automatic classification and ranking for web documents, in: Proceedings of the 4th International Conference on Fuzzy Systems and Knowlede Discovery, IEEE, 2007, pp. 627–631.

[18] H. Gu, Z. Kuanjiu, Text classification based on domain ontology, J. Commun. Comput. 3 (5) (2006) 261–272.

[19] C.d.C. Pereira, A.G.B. Tettamanzi, An evolutionary approach to ontology-based user model acquisition, in: Proceedings of the 5th International Workshop on Fuzzy Logic and Applications, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006, pp. 25–32.

[20] A.S. Imran, F.A. Cheikh, Blind image quality metric for blackboard lecture images, in: Proceedings of the 18th European Signal Processing Conference, IEEE, 2010, pp. 333–337.

[21] L. Jianzhuang, L. Wenqing, T. Yupeng, Automatic thresholding of gray-level pictures using two-dimension Otsu method, in: Proceedings of the International Conference on Circuits and Systems, vol. 1, IEEE, 1991, pp. 325–327.

[22] Z. Kastrati, A.S. Imran, Document image classification using SEMCON, in: Proceedings of the 20th Symposium on Signal Processing, Images and Computer Vision, STSIVA, 2015, pp. 1–6.

[23] A.S. Imran, S. Chanda, F.A. Cheikh, K. Franke, U. Pal, Cursive handwritten segmentation and recognition for instructional videos, in: 2012 Eighth International Conference on Signal Image Technology and Internet Based Systems, 2012, pp. 155–160.
[24] Z. Wu, M. Palmer, Verbs semantics and lexical selection, in: Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics, 1994, pp. 133–138.
[25] A. Maedche, Ontology Learning for the Semantic Web, Springer US, 2002.
[26] S. Brin, L. Page, The anatomy of a large-scale hypertextual web search engine, in: Proceedings of the 7th International Conference on World Wide Web 7, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, 1998, pp. 107–117.
[27] S. White, P. Smyth, Algorithms for estimating relative importance in networks, in: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, 2003, pp. 266–275.
[28] L. Page, S. Brin, R. Motwani, T. Winograd, The pagerank citation ranking: bringing order to the web, Technical Report, Stanford InfoLab, 1998.
[29] Ontotext, GraphDB workbench users guide 2014, http://owlim.ontotext.com/display/GraphDB6/GraphDBWorkbench. (Accessed 2015-09-20).

Dr. Zenun Kastrati obtained his Ph.D. from Norwegian University of Science and Technology - NTNU, Norway, in Computer Science in 2018 and a Master degree in Computer Science through the EU TEMPUS Programme developed and implemented jointly by University of Pristina, Kosovo, Université de La Rochelle, France, and Institute of Technology Carlow, Republic of Ireland. His research interests focus on the field of Semantic Web, with particular interest in text document classification using ontologies and machine learning. He has published several papers on international peer-reviewed journals and conferences related to the semantic web. Since 2009, he is associated with the Faculty of Electrical and Computer Engineering at the University of Prishtina, Kosovo as a teaching and research assistant.

Dr. Ali Shariq Imran obtained his Ph.D. from University of Oslo (UiO), Norway in Computer Science in 2013 and a Masters in Software Engineering and Computing from National University of Science & Technology (NUST), Pakistan. He specializes in applied research with a focus on deep learning technology and its application to signal processing, natural language processing, and semantic web. He has over 50 peer-reviewed journals and conference publications. He is a guest editor and a board member of IEEE Access Journals and HCI International Conference 2016/2017. He is currently associated with the department of electronics at National University of Science and Technology (NTNU), Norway as a postdoc researcher.