Sādhanā (2018) 43:27
https://doi.org/10.1007/s12046-018-0789-y

Enhancing multi-document summarization using concepts

PATTABHI R K RAO* and S LALITHA DEVI*

AU-KBC Research Centre, MIT Campus of Anna University, Chennai 600044, India
e-mail: t.pattabhi@gmail.com; sobha@au-kbc.org

MS received 21 November 2016; revised 12 May 2017; accepted 19 June 2017; published online 10 March 2018

Abstract. In this paper we propose a methodology to mine concepts from documents and use these concepts to generate an objective summary of all relevant documents. We use the conceptual graph (CG) formalism as proposed by Sowa to represent the concepts and their relationships in the documents. In the present work we have modified and extended the definition of the concept given by Sowa. The modified and extended definition is discussed in detail in section 2 of this paper. A CG of a set of relevant documents can be considered as a semantic network. The semantic network is generated by automatically extracting CG for each document and merging them into one. We discuss (i) generation of semantic network using CGs and (ii) generation of multi-document summary. Here we use restricted Boltzmann machines, a deep learning technique, for automatically extracting CGs. We have tested our methodology using MultiLing 2015 corpus. We have obtained encouraging results, which are comparable to those from the state of the art systems.

Keywords. Concept mining; text mining; multi-document summarization; machine learning; restricted Boltzmann machines; MultiLing 2015 dataset.

1. Introduction

Today, with the advancement of technology, there is an explosion of data available on web. Earlier the content had to be generated by the publishing houses. However, now with easy access to internet, the users themselves are able to generate content using blogs, micro-blogs such as Facebook and Twitter. Thus, there is a great need for mining the web and extracting relevant information. To automatically identify the relevant information, semantically driven mining methods are necessary. One such method where semantic relation can be utilized in mining is the concept-based mining where conceptual graph (CG) formalism is used to determine the concept and its relations. Here the underlying concepts have to be identified and they in turn are used for mining the relevant data. The concept thus obtained could be used for various applications such as Information retrieval, extraction, database creation, to generate summaries, etc.

Concept mining is the task of extracting the concepts embedded in the text document and concept is a representation of an idea or entity. A concept can be either a word or phrase and is totally dependent on the semantics of the sentence. Thus, identification of concepts from text documents involves aspects of artificial intelligence such as natural language processing (NLP) and machine learning. Concept mining is a non-trivial task and what constitutes a concept is also very important. Identification of concepts provides us a proper understanding of texts, helps in understanding relationships and gives a semantic representation of the text. We observe in the literature that the traditional methods to identify concepts have been through the use of thesaurus such as WordNet, dictionaries or lexicons.

There are various methods for summarizing a text and we find that concept-based summarization will be more semantically driven and gives cohesion to the summary automatically generated. “A summarizer is a system whose goal is to produce a condensed representation of the content of its input for human consumption” [1]. In most of the methods used for automated summary generation, the end result is a collection of sentences that do not have connectivity of topic, or we can say the cohesion of the text is not present. We are trying to bring in this cohesion to the summary through the CG-based summarization.

Automated summarization is an important area of research in NLP, which uses data mining technology. One of the popularly known earliest works on text summarization is by Luhn [2]. He proposed that frequency of a word in articles provides a useful measure of its significance. Significance factor was derived at sentence level and top ranking sentences were selected to form the auto-abstract.

A variety of automated summarization schemes have been proposed in the last decade. NeATS [3] is an

*For correspondence

approach based on sentence position, term frequency, topic signature and term clustering, and MEAD [4] is a centroid-based approach. Iterative graph-based ranking algorithms, such as Kleinberg' s HITS algorithm [5] and Google's Page- Rank [6], have been traditionally and successfully used in web-link analysis, social networks and more recently in text processing applications. Erkan and Radav [7], Mihalcea [8], Mihalcea and Tarau [9] and Mihalcea *et al.* [10] have been proposed for single-document summary generation.

Multi-document summarization is the process of filtering important information from a set of documents to produce a condensed version for particular users and applications. It can be viewed as an extension of single-document summarization. Issues like redundancy, novelty, coverage, temporal relatedness, compression ratio, etc., are more prominent in multi-document summarization [4]. MEAD is a multi-document summarization system. MEAD is a large scale extractive system that works in a general domain. SUMMONS [11] is an abstractive system that works in a strict domain, and relies on template-driven Information Extraction (IE) technology and Natural Language Generation (NLG) tools. Virendra and Tanveer [12] propose a multi-document summarization system that uses sentence clustering. It initially identifies summaries on single documents and then combines single-document summaries using sentence clustering.

Earlier works demonstrate that in the multi-document summarization, methods are used to combine single-document summaries to form multi-document summary. However, a more intuitive methodology would be to process the multiple documents as one set collectively and develop a coherent and semantic summary. In this work we propose such a methodology, where multiple documents are collectively considered for generating a summary. And for this we use the CG formalism.

We propose an algorithm that forms a semantic network of all the documents in the set. From this semantic network, we form a summary for the set of documents. The semantic network is generated using CG.

The major contributions of this work are as follows:

i) We have used CGs, which is semantic knowledge representation formalism.
ii) We have modified and extended the definition given by Sowa [13]. We discuss in detail what constitutes a concept and how concepts are formed in section 1.1 of this paper.
iii) We mine the concepts and their relationships and develop a CG completely by automated means. All the earlier works in literature have used partial automation for the development of CGs.
iv) The formalism of CGs helps in generating an abstractive summary. Most of the earlier works are extractive summaries. Those that have generated abstractive summary are not scalable, as they use a rule-based approach.
v. CGs are scalable and can be adopted for any language. Though here in this work we have demonstrated using English, they can be used for any language from any language family.
vi. This is one of the first works to fully, automatically extract CGs using one of the deep learning algorithms.

This paper is organized as follows. In the next sub-section, we describe briefly the background of CGs. Section 2 describes our modification and extension of concept definitions used to facilitate our work. The methodology and our approach are described in detail in section 3. In section 4, we describe the experiments and their results. Section 5 concludes the paper.

1.1 *Background of CG*

A CG is a graph representation of logic based on the semantic networks of artificial intelligence and existential graphs of Charles Sanders Peirce. John Sowa states the purpose of CGs as follws: "to express meaning in a form that is logically precise, human readable and computationally tractable" [13]. Mathematically, a CG is a bipartite, directed, finite graph; each node in the graph is either a concept node or relation node. Concept node represents entities, attributes, states and events, and relation node shows how the concepts are interconnected. A node (concept or relation) has two associated values: a type and a referent or marker; a referent can be either a single generic referent or an individual referent. Thus a CG consists of a set of concept types and a set of relation types.

A CG is represented mainly in two forms, viz., (i) display form and ii) linear form. The display form uses the traditional graph form, where concept nodes are represented by rectangular boxes and relation nodes are represented by ovals. In the linear form, concepts are represented by square brackets and relation nodes are represented using parenthesis. To represent these graphs internally in the computer system we use a list data structure consisting of triplet value (c1, c2, r), where c1 is concept one, c2 is concept two and r is the relationship between the concepts c1 and c2. This triplet structure can be again represented using traditional matrix representation, which is currently followed by information systems. The following example gives more insight into CGs.

Example 1: English sentence: "Marie hit the piggy bank with a hammer."

Figure 1 shows the CG for the example 1 sentence. The concepts are "Marie", "Hit", "Hammer" and "the piggy bank"; these concepts are connected by the relationships "agent", "instrument" and "patient", respectively. From the graph we can infer the following: the subject "Marie" "hit" the object "piggy bank" using the instrument "hammer".

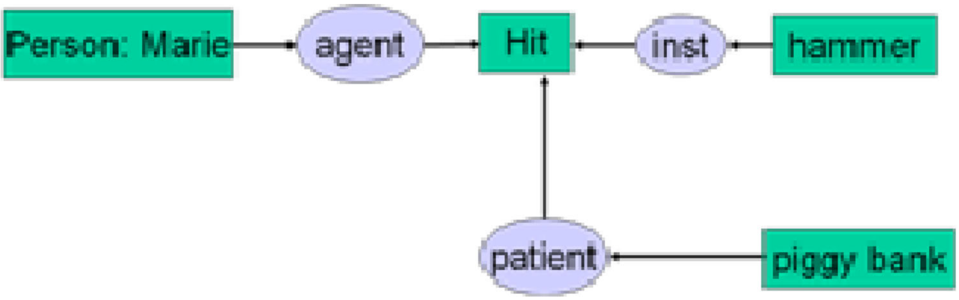

Figure 1. A conceptual graph – example sentence 1.

2. Definition of concepts and concept formation

One of the most important parts of this work is the identification of concepts in a document. Thus, it is necessary for us to understand what a concept is. In general a concept is defined as a representation or expression of a thought or idea conceived in human mind. This can be a perception of an object, a natural phenomenon or a feeling experienced by us. Edward and Douglas and [14], in their study of concepts, have summarized three views or approaches on definition of concepts, viz.

a) classical approach
b) probabilistic approach
c) prototype approach

Classical approach is one of the most popularly used in the formal treatments in mathematics and logic. Sowa, in his [13, 15] works on CG, has followed the classical approach. He defines concepts in terms of percepts. Percepts are the units of perception. He states that "the process of perception generates a structure 'u' called a CG in response to some external entity or scene 'e'; for every percept 'p' there is a concept 'c', called the interpretation of 'p'." Though he describes about abstraction, the emphasis is on the objects and perception. He describes words for the concepts. Sowa considers percepts as the basic unit for the concept formation [13]. His work elaborates on the structural part of the graph formalism. He does not specifically describe what constitutes a concept and how the different concepts are formed. Here we explore on what constitutes a concept from the semantic and computational perspective.

In our analysis of the documents we observe that the words in isolation have a particular meaning and have a different meaning when they are in collocation with immediate words. Also the meaning of individual words in a phrase varies with the meaning of the phrase. Thus we arrive at the conclusion that the phrases and words in collocation are to be considered as a single unit. We term these phrases or collocation words, which have unique meaning in a particular context, as 'a semantic unit'. We consider the semantic unit as the basis for our concept definition. Thus we define, for every semantic unit 'SU', a concept 'c' that is directly related to the semantics or meaning of 'SU'. In the document we look for phrases and collocations of words having unique meaning. Modifying the basic unit for concept, we have substantially modified and extended the definition of concept given by Sowa to facilitate our work.

What constitutes a semantic unit is discussed here. We consider the syntactic and semantic tags for defining the semantic units. The grammatical categories that form semantic units are described below:

i) **Multiword expressions**
Multiword expressions (MWEs) are expressions that are made up of at least two words that can be syntactically and/or semantically idiosyncratic in nature. Moreover, they act as a single unit. MWEs are idiosyncratic interpretations that cross word boundaries [16].
Examples: 'kick the bucket', 'in short', 'by and large', 'take off' (frozen forms).

ii) **Endocentric phrases**
An endocentric phrase consists of two words, in which one is the head and other is a modifier and both together would specify or narrow down the meaning of the head.
Examples: 'house boat', 'diesel motor'.

iii) **Exocentric phrases**
An exocentric phrase consists of two words whose meaning is different from those of the constituent words.
Examples: 'pale face', 'white collar', 'pick pocket'.

iv) **Possessive noun phrases**
Possessive noun phrases show the ownership or possession of an object or person. These phrases consist of two entities. The first entity owns or possesses the second entity.
Examples: 'cattle's pasture', 'John's book'

v) **Noun phrases**
They are a set of words that together form a noun, have one meaning and would refer to a single entity.
Examples: 'smart phone', 'running water'.

vi) **Verb phrases**
They are a set of words that together form a verb and have one meaning and would refer to a single action, activity.
Examples: 'mild boiling', 'fast bowling'.

Here we discussed about how two or more words could form a single concept. Further we give in detail how they are formed with examples.

Concept formation Here we describe how two or more words would combine to form a new concept.

A new concept 'c3' would be formed by the combination of concepts c1 and c2:

- if concept c1 modifies c2, i.e., c1 is modifier of c2;
- if c2 is specified by the specifier c1.

There are different types of combination of words that are formed by the grammatical features associated with the words in concept such as specifier, modifier and MWE. The explanation given here shows how such combinations can happen.

The new concept c3 is a kind or type of c1 or c2. In general the type of c3 is similar to the type of c2 since c2 forms the head of the combination.

Example: *[c3] – thematic [c1] + connection [c2].*
Example: *[c3] – mobile [c1] + phone [c2].*

The new concept c3 is a specialization of c2 and has a different meaning not obtained from c1 and c2.

Example *[c3] – love [c1] + life [c2],*
[c3] – deep [c1] + fry [c2],
[c3] – continuous [c1] + production [c2].

Types of concepts

Thus we now classify concepts into three types based on the cognition.

a) **Abstract concepts**
They are concepts for which there is no external physical image associated. They express the concepts of cognition, emotions, phenomenon and communication.

i) Cognition – express thoughts, e.g., think, like, hate, dream, love.
ii) Emotional state – express emotional state of the mind – e.g., anger, happy, excited, dejected.
iii) Phenomenon – natural things not seen by eyes, but experienced, – e.g., electricity, magnetism, gravitation.

b) **Semi-abstract concepts**
They are concepts that express actions of cognition; they are classified as semi-abstract because there is a physical image that we can associate with them, but actually they do not have, e.g., eat, drink, sleep, run, talk, say, red, green.

c) **Concrete concepts**
They are concepts that describe physical objects such as tree, plant, chair, book, etc.

Here we observe that for concepts the most likely part-of-speech (POS) categories of lexical words involved in the formation are noun–noun, adjective–noun, adverb–noun and noun–verbal noun. The prepositions or postpositions generally do not form concepts. They indicate the relationship between the concepts.

3. Our methodology

Our approach involves two primary components. The first component is the extraction of CGs for all the documents. A semantic network is generated. The second component involves generation of maximal link chains of the semantic network to generate a summary of the documents. The overall architecture of the system is shown in figure 2.

The input documents are collected from web. The aim of our present work is to generate multi-document summary for a given set of documents. Summary for multiple documents that are not related to each other or not similar to each other will create problems because the content in each of these documents will be totally different. Even if we generate a summary for these multiple documents, we will not be getting a proper representative, coherent summary. Hence for this purpose it is essential that we first cluster the input documents so that we get different clusters of related documents. Thus, as a first step, we perform soft clustering of the input documents to group the documents into several clusters and for each cluster of related documents we generate a summary.

After the documents are clustered, each document is processed to obtain syntactic and semantic information using NLP tools. The sentence splitting and tokenizing are done using grammar and heuristic rules. We make use of Brill's POS tagger [17] and fnTBL [18] for POS tagging and chunking, respectively. We have used a named entity recognizer that was developed in house. This uses Conditional Random Fields (CRFs), a machine learning technique [19]. After the NLP processing of the documents, we identify the concepts and their relationships. The next subsections describe in detail the extraction of concepts and their relationships and formation of CG.

3.1 *Extraction of CGs*

3.1a *Concept identification*: There are two sub-modules in this component; the first one is the concept identification module and second is the relation detection module. In the concept identification module, the concepts as defined in section 2 are automatically identified using deep learning. Deep learning is a branch of machine learning based on a

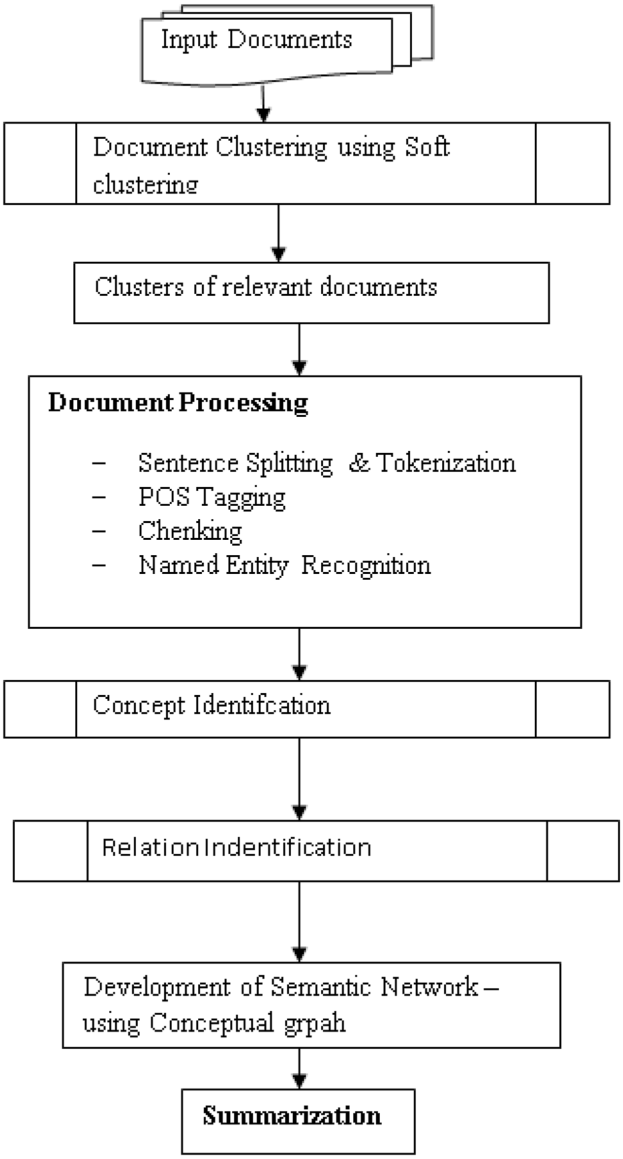
Input Documents
Document Clustering using Soft clustering
Clusters of relevant documents
Document Processing
– Sentence Splitting & Tokenization
– POS Tagging
– Chenking
– Named Entity Recognition
Concept Identifcation
Relation Indentification
Development of Semantic Network – using Conceptual grpah
Summarization

Figure 2. Overall system architecture.

set of algorithms that attempt to model high-level abstractions in data using a deep graph with multiple processing layers, composed of multiple linear and non-linear transformations [20, 21]. Deep learning is defined as a class of machine learning algorithms that use a cascade of many layers of nonlinear processing units for feature extraction and transformation and learn multiple levels of representations that correspond to different levels of abstraction; the levels form a hierarchy of concepts. In this work, a restricted Boltzmann machine (RBM), which is one of the methods in deep learning, is considered. In an earlier work by Pattabhi and Sobha [22], they described identification of concepts and their relationships using RBMs. The same implementation is used for identifying the concepts in this work.

A RBM is a probabilistic model. It models a distribution by splitting the input space in many different ways. RBM is a type of Boltzmann machine (BM). BMs are a particular form of log-linear Markov Random Field (MRF), for which the energy function is linear in its free parameters to make them powerful enough to represent complicated distributions that go from the limited parametric setting to a non-parametric one. We consider that some of the variables are never observed (they are called hidden). By having more hidden variables (also called hidden units) we can increase the modelling capacity of the BM. RBMs further restrict BMs to those without visible-visible and hidden-hidden connections. Unlike other unsupervised learning algorithms such as clustering, RBMs discover a rich representation of the input. RBMs are shallow, two-layer neural nets. The first layer of the RBM is called the visible, or input, layer, and the second is the hidden layer. A graphical depiction of a RBM is shown in figure 3.

Each circle in this graph represents a neuron-like unit called a node, and nodes are simply where calculations take place. The nodes are connected to each other across layers, but no two nodes of the same layer are linked. That is, there is no intra-layer communication — this is the restriction in an RBM. Each node is a locus of computation that processes input, and begins by making stochastic decisions about whether to transmit that input or not. Let x be the value of the visible node (or input value) and w1is the weight at node 1; then the result obtained is given by the following equation:

$$activation\ f((weight\ w * input\ x) + bias\ b) = output\ a. \tag{1}$$

Hence, Eq. (1) when expanded becomes

$$activation\ f((xw1 + xw2 + xw3 + xw4) + b) = a(output). \tag{2}$$

Because inputs from all visible nodes are being passed to all hidden nodes, an RBM can be defined as a *symmetrical bipartite graph*.

In this work we provide three levels of data as input in the visible layer. The first level is the words or tokens. The second level is the POS information and the third level is the named entity information. A modified graphical depiction of the RBM is shown in figure 4.

In our case we give input as word, POS and NE.

Thus, $x = <y1, y2, y3>$ where $y1$ =word, $y2$ =POS and $y3$ =NE.

Thus, in our case, Eq. (1) will be as follows:

$$\begin{aligned} f((<y1,y2,y3> * w1 + <y1,y2,y3> * w2 \\ + <y1,y2,y3> * w3 + <y1,y2,y3> * w4) + b) = a \end{aligned} \tag{3}$$

and when expanded, Eq. (3) becomes

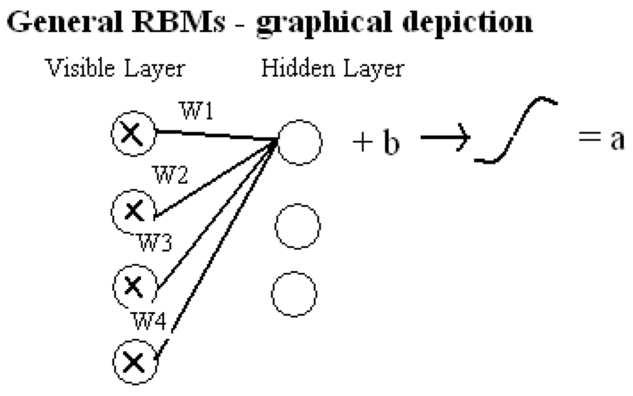

Figure 3. General RBM – graphical depiction.

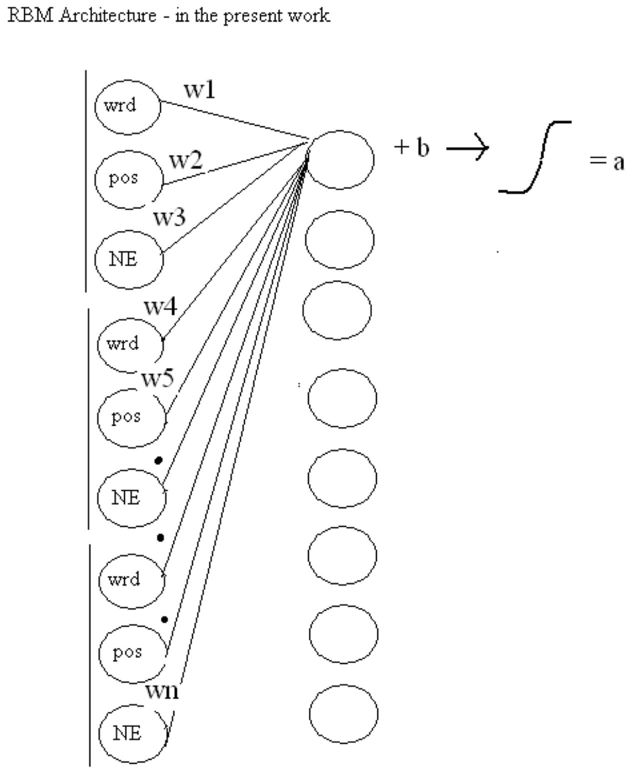

Figure 4. RBM architecture implemented in the present work.

$$f(((y1w1 * y2w1 * y3w1) + (y1w2 * y2w2 * y3w2) + (y1w3 * y2w3 * y3w3) + (y1w4 * y2w4 * y3w4)) + b) = a. \tag{4}$$

The motivation behind using the word, POS and NE tags for RBMs is that the unsupervised RBMs can detect the structures in the input and automatically obtain better feature vectors for classification. Most of the earlier NLP works have used only words as input for training the RBMs. The aim of the present work is to identify concepts from the word representations. The POS tag and NE tag help in adding sense and semantic information to the learning. The NE tag will help in identifying whether they are attributes of objects, phenomena, events, etc. This gives indications on the kind of concepts while learning and thus helps in concept identification. We have modelled RBMs as pairs of 3-ary observations. The 3-ary consists of word, POS and NE tag.

An RBM is a generative stochastic neural network that can learn probability distribution over its set of inputs. RBMs are trained to maximize the product of probabilities assigned to training set V (a matrix, each row of which is treated as a visible vector v):

argmax P(v)
w

or equivalently, to maximize the expected log probability of a training sample selected randomly from V:

argmax E[log P(v)]
w.

These three levels of data in the visible layer (or input layer) are converted to *n*-dimensional vectors and passed to the hidden layer of the RBM. The word vectors, POS vectors and NE vectors are the vector representations. They are obtained from the word2vec, and are also called as word embedding. Word embedding, in computational linguistics, is referred to as distributional semantic model, since the underlying semantic theory is called distributional semantics [20]. A real-valued *n*-dimensional vector for each level is formed using the word2vec algorithm. Word2vec creates or extracts features without human intervention and it includes the context of individual words/units provided in the projection layer. Word2vec is a computationally efficient predictive model for learning word embeddings from the text. The context comes in the form of multiword windows. Given enough data, usage and context, Word2vec can make highly accurate word associations [23]. Word2vec expects a string of sentences as its input. Each sentence — that is, each array of words — is vectored and compared to other vectored lists of words in an *n*-dimensional vector space. Related words and/or groups of words appear next to each other in that space. The output of the Word2vec neural net is a vocabulary with a vector attached to it, which can be fed into the next layer of the deep-learning net for classification. We make use of the DL4J Word2vec API for this purpose.

We have obtained optimal hyper-parameters for good performance by performing several trials. The main hyper-parameters that we need to tune include choice of activation function, number of hidden units, learning rate, dropout value and dimensionality of input units. We used 20% of training data for tuning these parameters. The optimal parameters include 200 hidden units, rectilinear activation function, 200 batch size, 0.025 learning rate, 0.5 dropout and 25 training iterations. We obtained the best development set accuracy for 80-dimensional word vector and 5-dimensional POS and NE tag vectors. Thus, for each word, we have 3-arys word vector, POS vector and NE vector, consisting of 90 dimensions. The output layer uses softmax function for probabilistic multi-class classification. We use our corpus as data for learning the Word2vec embeddings to convert the data to 90-dimensionsal 3-arys for input to the RBMs. We train the RBM and using the RBMs we identify the concepts given in the document. Once the concepts are extracted we need to identify the relationships between them and thus form a semantic network.

3.1b *Relation identification*: Concepts are always interconnected and do not exist in isolation. Concepts are connected with each by various relationships. We need to identify the various relationships that exist between the concepts to form a CG, which is a semantic network. Figure 5 shows the process flow diagram in the relation identification module.

3.2 *Summary generation*

Relationship identification module uses a hybrid approach. Here we have two sub-modules, rule-based engine and support vector machine (SVM) engine. The outputs of both engines are merged.

The linguistic rules are used initially to identify well-defined relations. The linguistic rules use syntactic structure of the sentence. Some of the linguistic rules are described here.

(i) If concept c1 is a verb/verbal phrase, concept c2 is a noun/noun phrase and there are subordinators such as "after", "later" before the c2 then they are markers of temporal relations. Using these temporal relationships one can infer senior–junior relationships, if this exists between two person concepts. For example, from the sentence "John joined ABC corp after Marie", John is junior to Marie.

(ii) If concepts c1 and c2 are connected by be verbs such as "is", then there exists"is a" or "sub-type" relationship.

We have developed a preposition relation mapping table, which defines different relations for each type of prepositions associated between verb–noun, noun–noun concepts. We also make use of an SVM classifier to identify relations independent of the rule-based engine. The output of SVM classifier and the output of the rule-based engine are merged to get the set of all relations. In the SVM engine output we consider only those relations that get high confidence score of more than 0.75 as valid relations. The features used for training SVM engine are the words and POS feature

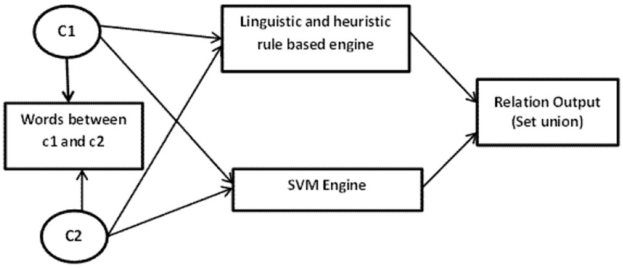
C1
Linguistic and heuristic rule based engine
Words between c1 and c2
Relation Output (Set union)
SVM Engine
C2

Figure 5. Relation detection module – process flow diagram.

3.3 *Summary generation*

The<concept-relation-concept> tuple obtained is actually a bipartite graph consisting of two classes of nodes "concepts" and "relations" and forms a CG. For a sentence, many such tuples are obtained depending on the number of clauses. They are merged into sub-graphs of the sentence to form a CG. Sub-graphs are merged by computing clique-sum. In this method, two graphs are merged by merging them along the shared clique. A clique in a graph is a subset of vertices in which every two vertices are connected by an edge. Each tuple can be considered as a clique. We identify the shared cliques and merge them to form a unified network of the CG for all the documents in a set. This complete CG is the semantic network of the set of documents. This is a kind of inheritance network, where the lower nodes correspond to more specific regularities and the upper nodes to more general ones. This hierarchy allows multiple inheritances. Thus we form a multi-document semantic network.

From this semantic network a multi-document summary is generated. The multi-document summary generation has the following two steps:

(i) identify clusters of the longest chain of nodes in the graph;
(ii) select the sentences that contain the nodes that are in the longest chain as summaries.

3.3a *Algorithm: Identification of cluster of the longest chain of nodes*: This is similar to identification of the longest path problem in a directed acyclic graph. The semantic network obtained from the earlier steps is a directed acyclic graph. The longest path problem for a general graph is not as easy as the shortest path problem because the longest path problem does not have optimal substructure property. In fact, the longest path problem is NP-Hard for a general graph. However, the longest path problem has a linear time solution for directed acyclic graphs. The idea is similar to linear time solution for the shortest path in a directed acyclic graph. Here, in our approach since we deal with bipartite graphs and for the purpose of summary generation we need to identify the most significant nodes, we adopt the Hopcroft–Karp algorithm to identify maximal matches of the graph. The Hopcroft–Karp algorithm [24, 25], which we have implemented, is described here.

Let U and V be the two sets in the bipartition of G, and let the matching from U to V at any time be represented as the set M. The algorithm is run in phases. Each phase consists of the following steps.

a) *A breadth-first search partitions the vertices of the graph into layers.*
b) *The free vertices in U are used as the starting vertices of this search and form the first layer of the partitioning.*
c) *At the first level of the search, there are only unmatched edges, since the free vertices in U are by definition not adjacent to any matched edges.*
d) *At subsequent levels of the search, the traversed edges are required to alternate between matched and unmatched. That is, when searching for successors from a vertex in U, only unmatched edges may be traversed, while from a vertex in V only matched edges may be traversed.*
e) *The search terminates at the first layer k where one or more free vertices in V are reached.*
f) *All free vertices in V at layer k are collected into a set F. That is, a vertex v is put into F if and only if it ends the shortest augmenting path.*
g) *The algorithm finds a maximal set of vertex disjoint augmenting paths of length k. This set may be computed by depth first search from F to the free vertices in U, using the breadth first layering to guide the search.*
h) *The depth first search is allowed only to follow edges that lead to an unused vertex in the previous layer, and paths in the depth first search tree must alternate between matched and unmatched edges.*
i) *Once an augmenting path is found that involves one of the vertices in F, the depth first search is continued from the next starting vertex.*
j) *Each one of the paths found in this way is used to enlarge M.*

The algorithm terminates when no more augmenting paths are found in the breadth first search part of one of the phases. Now the sentences are selected from the documents that contain the nodes or vertices of the maximal match. We put a thrush hold to the number of sentences to be considered for summary. The number of such selected sentences is restricted to 10% of the total sentences in the whole set of documents.

4. Experiments and results

The evaluation of the concept-relation identifier and the multi-document summarization is discussed in this section. One of the first tasks for performing experiments using machine learning is to have a manually annotated corpus. We have performed our experiments using two sets of data. Thus we first describe the manual annotation work.

4.1 *Data annotation*

In this work we have used two sets of data, one to develop concept identifier module (CG extraction) and other for the multi-document multi-lingual summarization module. As described earlier we have applied the CGs for generating an automatic summarizer. The automatic summarizer has been tested using the benchmark data provided during the MultiLing 2015 MMS track shared task [26].

For the CG extraction, we have prepared the data. We collected documents from online news portals such as WSJ, NYT and The Times of India. The data consist of 1000 news articles. The news articles were taken from different domains such as political, business, sports, accidents, disasters, science and entertainment. The corpus was divided into two sets: training and testing (80–20 ratio). In the training phase the documents are pre-processed for POS tagging [17] and NP–VP chunking [18]. After pre-processing, the words are tagged with concept classes, i.e., we mark up chunk of words depending on whether they form a concept or not. We have used HTML style mark-up format for annotating the concepts and the relationships. The concepts are marked using the tag<concept> and relationships are marked using<relation> tag.

The<concept> tag has the attributes “ID”, “Type”. The attribute “ID” takes a numeric value, which is a unique number assigned for each concept. This attribute is obligatory. The other attribute “Type” is optional, which describes the class of concept, whether it is abstract, concrete or semi-abstract.

The<relation> has the attributes “ID”, “CSREF” and “Type”. The first two are obligatory and the last attribute is optional. The attribute “ID” takes a numeric value, which is a unique number assigned for each relation. The attribute “CSREF” takes the IDs of the concepts that are connected by this relation.

For example, let us consider the sentence described in Example 1:

<concept ID=‘1’ Type=‘Concrete’>Marie</concept><relation ID=‘1’CSREF=‘1:2’>hit</relation><concept ID=‘2’>the piggy bank</concept><relation =‘2’ CSREF=‘2:3’>with</relation><concept ID=‘3’>a hammer</concept>

The same information is stored in a triplet form as a list data structure in the machine for machine learning. The triplet structure for this example would be as follows:

a. (“Marie”, “the piggy bank”, “hit”), b. (“the piggy bank”, “a hammer”, “with”).

4.2 *Experiments and results for CG extraction*

The concepts are represented as vectors of 100 dimensions using the Word2Vec algorithm. These vectors are then presented in the format as required by the DBMs and trained. The evaluation metrics used are the precision, recall and *F*-measure as used in other NLP works. Table 1 shows the results for<concept-relation-concept> tuple.

Table 1. Our system results in comparison with earlier works reported in literature for extraction of conceptual graphs.

Sl. no.	Method	Precision (%)	Recall (%)	*F*-measure (%)
1	[27]	78.75	70.20	74.22
2	[28]	73.30	68.30	70.71
3	Present approach	79.34	72.54	75.79

We have taken 200 documents from various domains of news papers for evaluation. In table 1, all the available systems results are given. The Shih-Yao system [27] is the first system developed; it uses rule-based approach and the rules are developed for chemical domain documents. Though it cannot be compared to the domain we have taken, being the only system available, we have taken it as one of the base systems. The other system by [28] is on the same domain but uses a different machine learning approach, CRFs. From the results it can be seen that our present approach gives encouraging results compared to the other two approaches.

The analysis of the results shows that incorrect identification of concepts gave the maximum error. In the total number of errors, 25% of the errors are false positives. Most of the errors are due to either combining more than one concept as single concept or splitting a single concept into two different concepts. In the output we observe either two concepts are combined into one or only partially identified. The main advantages of our approach are the following: (i) it is scalable and (ii) robust as this could be used for any domain and any set of documents. The other advantage is that the feature extraction for the softmax as well as for the SVM is completely automated with the use of deep learning. We observe that we have obtained improved results.

4.3 *Experiments and results for summary generation*

We have first used the concept-relation identification system discussed in the previous section for identifying the concept-relation and then from the concept-relation identified, generated the summary. The corpus used for this purpose is the MMS track corpus from task dataset of the MultiLing 2015 [26], which was categorized into 15 news clusters, where each cluster contained a set of 10 news articles related to a topic. We obtained a precision of 78.34%, recall of 72.54% and *F*-measure of 75.32%. In the second step we used the summary generation. Here there is no training phase as we directly use the CGs produced in the first step. As explained in section 3.3, after the CGs are obtained we form the semantic network of the CGs obtained for all documents in a set. From the semantic network, the summary is generated. We tested the proposed summary generator. For summary evaluation, we used the commonly used automatic evaluation tool called the ROUGE package, which was developed by [29]. ROUGE is based on the *n*-gram overlap between a system-generated summary and a set of reference summaries. It measures a summary quality by counting overlapping units, such as the word *n*-gram, word sequences and word pairs between the candidate summary and the reference summaries. The ROUGE-*N* recall score is computed using the formula shown in figure 6, where ROUGE-*N* is an *n*-gram recall between a system-generated summary and a set of reference summaries; ‘*n*’ stands for the length of the *n*-gram, gram and Countmatch (gram) are the maximum number of *n*-grams co-occurring in a system-generated summary and a set of reference summaries. The older versions of the ROUGE package, such as Versions 1.1, 1.2, 1.2.1 and 1.4.2, used only a recall-based score for summary evaluation. However, the newer version of the ROUGE package – ROUGE 1.5.5 – evaluates summaries based on three metrics such as ROUGE-*N* precision, ROUGE-*N* recall and the ROUGE-*N* *F*-score, where *N* can be 1, 2, 3, 4, etc. Thus, the ROUGE toolkit reports separate scores for 1, 2, 3 and 4-grams, and also for the skip bigram. We have used ROUGE Version 1.5.5 for our system evaluation. Among the various ROUGE scores, the unigram- and bigram-based ROUGE score (ROUGE-1 & 2) have been shown to agree most with human judgment [30]. The ROUGE-2 metric is found to have high correlation with human judgments at a 95% confidence level and hence used for evaluation.

Table 2 shows the ROUGE score obtained using our proposed approach and also shows results of other reported systems in MultiLing 2015 evaluation exercise. The results obtained are comparable to those from the state of the art.

$$ROUGE-N=\frac{\sum_{S\in(\text{Reference Summaries })}\sum_{gram_n\in S}\text{Count}_{\text{match}}(gram_n)}{\sum_{S\in(\text{Reference Summaries})}\sum_{gram_n\in S}\text{Count}(gram_n)}$$

Figure 6. ROUGE-*N* recall score formula.

Table 2. Our system results in comparison with earlier works reported in literature for multi-document summary generation.

System/algorithm reference	Average *F*-measure
MultiLing2015 Baseline	0.1800
MMS 2	0.2220
MMS 8	0.2185
MMS 15	0.2004
Our approach	0.2198

5. Conclusion

We have presented an algorithm for text mining to enhance multi-document summarization. Here we have generated multi-document summaries using CGs. One of the main advantages is that the summary is coherent and also easily scalable. This approach can be adopted for any language, with a very minimal customization, since this uses CG principles of knowledge representations. We have tested our approach using MultiLing 2015 corpus, which is a very popularly used benchmark dataset. We obtain average *F*-measure of 0.2198 ROUGE score which is comparable to that from the state of the art. As we can see from table 2, this method has outperformed most of the other methods. The main objective of our work was to ascertain how capturing of structure of a sentence and thereby of the document would help in generating multi-document summaries. We found that it was very useful and got good results. The use of CGs helped in the capture of the structure and the semantics and helped in generating abstractive summary.

References

[1] Mani I 2001 Summarization evaluation: an overview. In: *Proceedings of NTCIR*

[2] Luhn H P 1958 The automatic creation of literature abstracts. *IBM J. Res. Dev.* 2(2): 159–165

[3] Lin C Y and Hovy E H 2002 From single to multi-document summarization: a prototype system and its evaluation. In: *Proceedings of ACL-2002*, pp. 457–464

[4] Radev D, Jing H, Stys M and Tam D 2004 Centroid-based summarization of multiple documents. *Inf. Process. Manage.* 40: 919–938

[5] Kleinberg 1999 Authoritative sources in a hyperlinked environment. *J. ACM* 46(5): 604–632

[6] Brin S and Page L 1998 The anatomy of a large scale hypertextual Web search engine. *Comput. Netw. ISDN Syst.* 30: 1–7

[7] Erkan G and Radev D 2004 Lexpagerank: prestige in multi-document text summarization. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Barcelona, Spain, July

[8] Mihalcea R Graph-based ranking algorithms for sentence extraction, applied to text summarization. In: *Proceedings of ACL 2004 on Interactive Poster and Demonstration Sessions (ACLdemo 2004)*, Barcelona, Spain

[9] Mihalcea R and Tarau P 2004 TextRank – bringing order into texts. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*, Barcelona, Spain

[10] Mihalcea R, Tarau P and Figa E 2004 PageRank on semantic networks, with application to word sense disambiguation. In: *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*, Geneva, Switzerland

[11] McKeownand K and Radev D 1995 Generating summaries of multiple news articles. In: *Proceedings of the 18th Annual International ACM*, Seattle, WA, pp. 74–82

[12] Virendra G and Tanveer J S 2012 Multi-document summarization using sentence clustering. In: *IEEE Proceedings of the 4th International Conference on Intelligent Human–Computer Interaction*, Kharagpur, India, pp. 314–318

[13] Sowa J F 1984 *Conceptual structures, information processing in mind and machine.* Addison Wesley, Boston, MA, USA

[14] Edward E S and Douglas L M 1981 *Categories and concepts* Cambridge, Massachusetts–London, England: Harvard University Press

[15] Sowa J F 1976 Conceptual graphs for a data base interface. *IBM J. Res. Dev.* 20(4): 336–357

[16] Ivan A S, Baldwin T, Bond F, Copestake A and Flickinger D 2002 Multiword expressions: a pain in the neck for NLP. In: *Proceedings of the 3rd International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2002)*, Mexico City, Mexico, pp. 1–15

[17] Brill E 1994 Some advances in transformation based part of speech tagging. In: *Proceedings of the Twelfth International Conference on Artificial Intelligence (AAAI-94)*, Seattle, WA, pp. 722–727

[18] Ngai G and Florian R Transformation-based learning in the fast lane. In: *Proceedings of NAACL'2001*, Pittsburgh, PA, pp. 40–47

[19] Lafferty J, McCallum A and Pereira F 2001 Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: *Proceedings of the 18th International Conference on Machine Learning (ICML-2001)*, pp. 282–289

[20] Hinton G and Salakhutdinov R 2006 Reducing the dimensionality of data with neural networks. *Science* 313(5786): 504–507

[21] Srivastava N, Salakhutdinov R R and Hinton G E 2013 Modeling documents with a deep Boltzmann machine. In: *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*

[22] Rao P R K and Lalitha Devi S 2015 Automatic identification of conceptual structures using deep Boltzmann machines. In: *Proceedings of the Forum for Information Reterival and Evaluation, ACM DL*, Gandhinagar, India, pp. 16–80

[23] Mikolov T, Chen K, Corrado G and Dean J 2013 Efficient estimation of word representations in vector space. In: *Proceedings of the Workshop at ICLR*

[24] Blum N 2001 A simplified realization of the Hopcroft–Karp approach to maximum matching in general graphs. *Tech. Rep. 895549-CS*, Computer Science Department, University of Bonn

[25] Hopcroft J E and Karp R M 1973 An n5/2 algorithm for maximum matchings in bipartite graphs. *SIAM J. Comput.* 2(4): 225–231, https://doi.org/10.1137/0202019

[26] Giannakopoulos G, Kubina J, John M C, Steinberger J, Favre B, Kabadjov M, Kruschwitz U and Poesio M 2015 Multiling 2015: multilingual summarization of single and multi-documents, on-line fora, and call-center conversations. In: *Proceedings of SIGDIAL*, Prague, pp. 270–274

[27] Yang S Y and Soo V W 2012 Extract conceptual graphs from plain texts in patent claims. *J. Eng. Appl. Artif. Intell.* 25(4): 874–887

[28] Rao P R K, Lalitha Devi S and Rosso P 2013 Automatic identification of concepts and conceptual relations from patents using machine learning methods. In: *Proceedings of*

the 10th International Conference on Natural Language Processing (ICON 2013), Noida, India

[29] Lin C Y 2004 ROUGE: a package for automatic evaluation of summaries. In: *Proceedings of the Workshop on Text Summarization Branches Out*, Barcelona, Spain

[30] Lin C Y and Hovy E 2003 Automatic evaluation of summaries using n-gram co-occurrence. In: *Proceedings of the 2003 Language Technology Conference (HLT-NAACL 2003)*, Edmonton, Canada, pp. 71–78