Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.18576/amis/13S107

A Multi-Perspective Knowledge Discovery Approach for Word Sense Disambiguation

Rajini. S[1,*] *and Vasuki. A*[2]

[1] Department of Computer Science and Engineering, Kumaraguru College of Technology, Coimbatore, India
[2] Department of Mechatronics Engineering, Kumaraguru College of Technology, Coimbatore, India

Received: 2 May 2019, Revised: 1 Jul. 2019, Accepted: 23 Jul. 2019
Published online: 1 Aug. 2019

Abstract: In this paper, a multi-perspective knowledge discovery approach for word sense disambiguation is proposed. Initially a two-step pre-processing is carried out which includes stop word removal and stemming. From the context of an ambiguous word, the features are extracted such as word embedding, continuous bag-of-words and skip gram models. The unigrams and bigrams are extracted from the text and the bigrams are integrated with the unigrams. Then distributional similarity and semantic similarity scores are evaluated based on the local mutual information, point-wise local mutual information and the feature values. For the context classification, convolutional neural network model is utilized. In order to get strong baseline result, the distributional similarity and the semantic similarity matching algorithm is applied for the text features, particularly the unigram representation process. SemEval-2010 Word Sense Induction and Disambiguation dataset is used in this work. The experimental analysis is carried out by implementing various classifiers such as KNN, Naïve Bayes and Random Forest methods. The proposed approach provides good outcomes in terms of accuracy, F-measure, precision and recall.

Keywords: Word Sense Disambiguation, Word Embedding, Convolutional Neural Network, Distributional Similarity, Semantic Similarity

1 Introduction

Word Sense Disambiguation (WSD) is one of the techniques in natural language processing that can be used for information retrieval, web search, indexing and other purposes. It resolves the problem of identifying the correct sense of ambiguous words based on the context [1]. The objective of the techniques is to solve the WSD problem is to assign a proper meaning to an ambiguous word in a context by selecting the correct meaning from an inventory of word meanings [2]. Identifying the correct meanings of words is a difficult work for machines. The sense or meaning inventories are those that hold words, their senses and additional information about them [3]. The two major types of inventories used in the models intended to carry out WSD are structured inventories such as ontologies and unstructured inventories such as corpora [4]. WordNet (WN) is a commonly-used inventory. It is comprised of words that are sorted into groups of meanings called as *synsets* [5]. Each *synset* contains a textual definition and in these inventories, there are details pertinent to lexical and semantic relations between pairs of *synsets* [6]. In available literature, there are papers dedicated to generic word sense disambiguation. There is a lot of ambiguity in human language because many words possess multiple meanings based on the context in which they appear. For example, consider the following two sentences:

1. The board meeting of Oracle Corporation took place day before yesterday.
2. The teacher wrote on the black board.

The meaning of *board* is different in the two sentences. In the first sentence, *board* means an *organized body of administrators* and in the second sentence, *board* refers to *a large vertically- positioned flat surface used for writing*. In order to make machines understand the underlying meaning, it is necessary for them to process unstructured textual information and transform them into data structures for analysis.

WSD helps to identify the right sense or meaning of a word in a sentence, when the word has several meanings.

[*] Corresponding author e-mail: rajinisphd@gmail.com

WSD tasks can be performed on one or more texts. Text refers to a list of words in sequence and WSD refers to the task of assigning an appropriate sense(s) to all or a number of words in the text. WSD can also be considered as a classification task in which word senses are the classes and an automatic classification method can be used to assign each word occurrence to one or more classes. This is performed by making use of context information and external knowledge sources. There are two different types of WSD tasks namely, *lexical sample* WSD and *all-words* WSD. In the former method, a set of target words alone are disambiguated whereas in the latter method, all the open-class words in the text are disambiguated. WSD approaches can be characterized into knowledge-based methods and machine learning-based methods [7]. Knowledge-based methods mainly focus on the use of knowledge lexical resources and the machine learning-based methods use the evidence separated from annotated and unannotated text and other techniques include domain-driven disambiguation, such as meta-heuristics algorithm [8].

Swarm Intelligence (SI) algorithms have been developed based on inspiration from the behaviour of swarm of insects. Insect swarms can engage effectively in a multitude of exercises; however, the swarms have no supervising or controlling entity [9]. Every individual member, in spite of its restricted capacity, helps to solve various difficult issues through simple collaboration with other members of the swarm. SI algorithms use these features of self-organization and decentralized control [10]. Ant Colony Optimization (ACO) approaches are SI algorithms inspired by pheromone-based ant foraging strategies. Several variants of ACO methods including Ant System (AS), Ant Colony System (ACS) and Max-Min Ant System (MMAS) are proposed [11]. These systems have been effectively utilized for many combinatorial optimization issues, and they give optimal solutions [12]. There is a large body of literature related to the use of these techniques for solving such problems such as the travelling salesperson problem (TSP) [21], the quadratic assignment issue and the vehicle routing problem (VRP) [13]. These techniques have also been used in different Natural Language Processing (NLP) tasks and applications, including WSD [14]. The promising outcomes acquired by Nguyen and Ock with their ACO-TSP model for performing WSD show the effectiveness of this approach and support further investigations of other ACO variants for this task [17]. Hybrid genetic-ant colony optimization algorithm is one of the methods for solving the WSD problem. Two well-known ACO algorithms that have achieved competitive results for a variety of problems are the ACS and the MMAS algorithms [18]. Cuckoo Search, Firefly and Bee Colony Optimization algorithms are the other optimization methods which are used for WSD problem solving [[1], [25]].

Application of graph connectivity [20], N-gram feature method [22] and semantic approach for text clustering [24] help perform unsupervised WSD. In Natural Language Processing (NLP), word sense disambiguation is difficult because this task requires selection of the most precise sense for a word from a set of predefined synonyms. The WSD problem can be solved by using either unsupervised learning or supervised learning method. Supervised learning methods involve use of text corpus or machine-readable dictionaries. Unsupervised learning methods involve knowledge based systems to find correct senses. For precise word sense disambiguation, knowledge discovery is a significant part which involves lexical ambiguity and semantic ambiguity. Further, it concentrates on developing an intelligent deep learning model to resolve the issues in the NLP areas. It is evident from the present works in NLP that most researchers have focused on resolving knowledge discovery challenges such as word sense disambiguation in an un-unified way. The bio-inspired algorithm-based approaches such as Cuckoo, Firefly and Particle Swarm Optimization (PSO) degrade the general efficacy of the knowledge discovery process which leads to certain conceptual misinterpretation. Hence, there is a need to develop a deep learning- aided multi-perspective knowledge discovery process to resolve WSD problem in NLP context. In order to achieve better efficiency, in this work, CNN-based context classification is performed.

Our contribution in this paper are as follows: Deep learning based Convolutional neural Network is utilized for context classification. The classification process is carried out by utilizing the distributional and semantic similarity score calculations. This enhances the work by achieving better accuracy.

The rest of this paper is organized as follows: Section 2 provides the research ideas related to this work, Section 3 describes the proposed methodology, Section 4 shows the experimental results, Section 5 provides the conclusion of the work and Section 6 contains the references.

2 Related Work

Ahmad et al. [10] perceived the WSD problem from a different viewpoint. They proposed a system,that consisted of two main parts. The first part included a data mining algorithm, which ran offline and extracted some useful knowledge about the co-occurrences of the words. In this algorithm, each sentence was imagined as a transaction in Market Basket Data Analysis problem, and the words included in a sentence played the role of purchased items. The second part of the system was an expert system whose knowledge base consisted of the set of association rules generated by the first part. Moreover, in order to deduce the correct senses of the words, they introduced an efficient algorithm based on forward chaining in order to be used in the inference engine of the proposed expert system.

Abualhaijaa and Zimmermann [1] made use of Bee colony optimization for performing word sense disambiguation and named it as D-Bees. Here, several artificial bee agents worked in unison to resolve the issue. D-Bees algorithm has been evaluated on a standard *SemEval* 2007 task 7 coarse-grained English all-words corpus and was compared to the genetic and simulated annealing algorithms as well as ant colony algorithm. It was shown that the bee and ant colony optimization approaches achieved better results than the genetic and simulated annealing algorithms on the given dataset.

Hung et al. [14] suggested that word sense disambiguation can be carried out before assigning a proper sentiment score for a word in SentiWordNet. They proposed three WSD methods for building a domain-oriented sentiment lexicon for sentiment classification. They also combined two tokenization approaches with sentiment vector space modelling. The experiments proved that their word sense disambiguation-based SentiWordNet had the capability to enhance the performance of sentiment classification.

Bartosz and Bridget [4] presented a novel machine learning approach for WSD. They identified two major difficulties for this task: multi-class imbalanced instance distribution and potential presence of class label noise due to the usage of semi-automatic labelling approaches. To address this, they developed a decomposition framework based on a divide-and-conquer solution. Each class from the dataset was treated independently and a dedicated ensemble of local classifiers was trained on it. In order to handle the high-dimensional feature space and ensure diversity among base learners, they used a random subspaces solution. Then, each subspace was subjected to a kernel whitening procedure that rescaled supplied instances in order to form a more compact class representation. Finally, on each transformed subspace, they trained a weighted one-class support vector machine. Therefore, the original multi-class problem was decomposed by assigning a one-class classifier ensemble to each class. They presented a two-step classifier combination scheme, where firstly they combined classifiers within each class and then used their aggregated outputs to reconstruct the original multi-class problem.

Wang et al. [23] focused on kernel methods for automatic WSD. Within this framework, the main difficulty was to design an appropriate kernel function to represent the sense distinction knowledge. Semantic diffusion kernel, which modelled semantic similarity by means of a diffusion process on a graph defined by lexicon and co-occurrence information to smooth the typical representation, had been successfully applied to WSD. However, the diffusion was an unsupervised process, which failed to exploit the class information in a supervised classification scenario. To address the limitation, they presented a sprinkled semantic diffusion kernel to make use of the class knowledge of training documents in addition to the co-occurrence knowledge. The basic idea was to construct an augmented term document matrix by encoding class information as additional terms and appending them to the training documents. Diffusion was then performed on the augmented term-document matrix. In that way, the words belonging to the same class were indirectly drawn closer to each other and it was inferred that the class-specific word correlations got strengthened.

2.1 Background

Convolution Neural Networks (CNNs) are a special type of neural network which focus on data that has grid-like topology [15]. CNNs have been extremely successful in practical applications and are adapted in many architectures such as: text classification, image recognition, object detection and segmentation. By combining multiple blocks and using small filter sizes, CNNs can learn an in-depth representation of input, an affordance that would allow it to surpass all other traditional methods in image-related tasks. CNNs take advantage of the fact that the input consists of images and they constrain the architecture in a more sensible way. In particular, unlike a regular neural network, the layers of a ConvNet have neurons arranged in 3 dimensions: width, height, depth. Depth refers to the third dimension of an activation volume, not to the depth of a full neural network, which can refer to the total number of layers in a network.

3 Proposed Multi Perspective Knowledge Discovery Approach For Word Sense Disambiguation

Word sense disambiguation helps in discovering the precise sense of ambiguous words in text. Due to the vastness of English language, ambiguity problem occurs frequently and extremely intelligent disambiguation approaches are required. In this paper, a deep learning-aided multi perspective knowledge discovery approach has been developed to resolve the word sense disambiguation problem.

Initially, from the context of an ambiguous word, unigrams and bigrams are extracted from the text and for the purpose of experimentation, bigrams are integrated with unigrams. Further, the context of the ambiguous word is sorted out utilizing bag-of-concepts approach. Moreover, we have to select two sets of features such as Part-of-Speech (POS) tagging and local collocations features. In order to carry out word embedding, *word2vec* model has been used and this utilizes Continuous Bag-of-Words (CBOW) and skip-gram to generate the word embedding. For the purpose of determining the aggregation of word embeddings by calculating sum and average, the vectors of the words are generated by considering the context of the ambiguous word. Finally, to get strong baseline result, semantic similarity matching algorithm has been employed for text features.

Figure-1 represents the flow of the proposed method. Initially, pre-processing phase comprises of two steps namely stop word removal and stemming. Feature extraction process is carried out for the training corpus by using the word embeddings, CBOW and skip gram model. Based on the related feature extraction output, the semantic change measurement is evaluated in terms of distributional similarity and semantic similarity. The distributional similarity and the semantic similarity of words are evaluated and given as input to the CNN to get the targeted output. The following sub-sections present a detailed overview of the proposed approach.

3.1 Preprocessing

The pre-processing step is used to reduce the size of the specified dataset and improve the classification results. Real world data are generally incomplete (lacking attribute values), noisy (containing errors or outliers) and inconsistent. Data must be pre-processed in order to perform any data mining functionality. Pre-processing has been performed on the specified dataset and includes:

1.Stop word removal
2.Stemming

3.1.1 Stop word removal

Stop word removal eliminates the words which provide less or no information to carry out text analysis. Words like articles, prepositions, conjunctions, common verbs (e.g. ‘know ’, ‘see ’, ‘do ’, ‘be ’), auxiliary verbs, adjectives (e.g. ‘big ’, ‘late ’, ‘high ’), and pronouns are removed, leaving the content words which are likely to have some meaning. For this reason, a stop-list is usually built with words that should be filtered in the document representation process. They have no distinguishing potential between categories. Words that are to be included in the stop-list are language- and task-dependent. However, there exists a set of general words that can be considered stop-words for almost all tasks such as “and” and “or” words that appear in very few examples (documents) and which are also filtered, because they will very unlikely represent a category.

3.1.2 Stemming

Another commonly-used method is stemming, where the word stem is derived from the occurrence of a word by removing case and inflection information. For example, “computes”, “computing” and “computer” are all mapped to the same stem “compute”. Stemming does not alter significantly the information included in document representation, but it does circumvent feature expansion. The words are passed through a stemmer, which reduces multiple instances of a single word to the root word. For instance,“flying”and “flew”are reduced to fly.

Stemming is the process where the word suffixes are removed. After pre-processing, the incomplete, noisy and inconsistent data are removed. In the next step the relevant features are selected from the pre-processed dataset by using the feature extraction process. A consonant is denoted by c, a vowel by v. A list ccc... of length greater than 0 and it will be denoted by C, and a list vvv... of length greater than 0 is denoted by V.

3.2 Feature Extraction for Word Sense Disambiguation

From the processed dataset, the feature selection process is done. Feature selection is the selection of a subset of features from a larger pool of available features. The goal is to improve the prediction performance of the predictors. This is a crucial step in the design of any classification system, as a poor-feature choice leads to poor system performance.

3.2.1 Word Embedding

Word embedding is a method that helps to convert the bag-of-words representation to a continuous space form. Dimensionality reduction occurs when continuous space forms are used, and thereby making it easy to identify word meanings. Word embedding techniques seek to embed representations of words. Use of a cosine similarity measure on this abstract vector space of embedded words can be used to identify a list of words that are used in similar contexts with respect to a given word. These semantically-related words may be used for various natural language processing tasks. The general idea is to train moving windows with vector embeddings for the words and classify the individual windows. This finds application in tasks such as POS tagging, semantic role labelling, named-entity recognition and others.

The state-of-the-art word embedding approaches involve training deep neural networks with the help of negative sampling. It has been reported that word2vec produces reliable word embeddings in a very efficient manner. Using *word2vec* method, it is possible to generate the word embeddings using CBOW or skip-gram [25]. Word representations can be learnt using a simple neural network model that helps to predict a word’s neighbours. Due to its simplicity, the skip-gram and CBOW models can be trained on a large amount of text data; this parallelized implementation can learn a model from billions of words in hours.

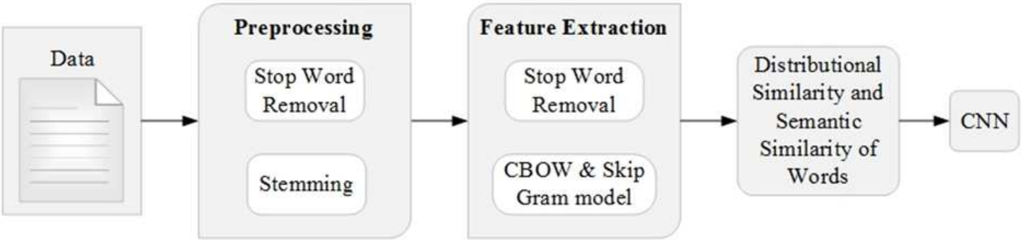

Fig. 1: Flow diagram of proposed approach which involves pre-processing, feature extraction, semantic change measurement and CNN usage

3.2.2 Continuous Bag-of-Words and Skip-Gram models

It was recently shown that the distributed representations of words captured surprisingly much linguistic regularity, and that there are many types of similarities among words that can be expressed as linear translations. For example, vector operations “king”$-$ “man”$+$ “woman”results in a vector that is close to “queen”. Two particular models for learning word representations that can be efficiently trained on large amounts of text data are skip-gram and CBOW [25]. Figure-2 represents the architecture of CBOW method. In CBOW, the training objective is to combine the representations of surrounding words to predict the word in the middle. Due to their low computational complexity, the skip-gram and CBOW models can be trained on a large corpus in a short time, that is, billions of words in hours.

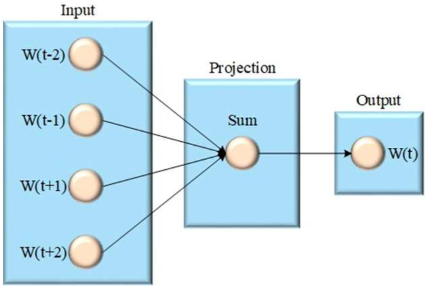

Fig. 2: Continuous Bag-of-Words (CBOW) model used for learning word representations to predict the middle word’s context

Figure-3 represents the architecture of skip-gram method. In the skip-gram model [26], the training objective is to make the system learn how to use the current word to predict its neighbours and thereby judge its context. The skip-gram gives better word representations when the monolingual data is small. CBOW is faster and more suitable for large datasets. They also tend to learn very similar representations for languages.

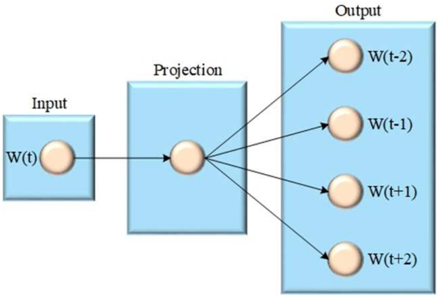

Fig. 3: Skip-Gram model for predicting the neighbouring words based on the current word to judge the context

For a list of training words $w_1, w_2, w_3, w_4, ..., w_T$, the objective of the skip-gram model is to maximize the average log probability ρ.

$$\rho = \frac{1}{T}\sum_{t=1}^{T}\left[\sum_{j=-k}^{k}\log p\left(w_{t+j}|w_t\right)\right] \tag{1}$$

where T denotes the total number of words in the training corpus and k is the size of the training window.
The inner summation goes from $-k$ to k to compute the log probability of correctly predicting the word w_{t+j} given the word in the middle w_t. The outer summation goes over all words in the training corpus.

In the skip-gram model, every word w is associated with two learnable parameter vectors, u_w and v_w. They are the input and output vectors of w respectively. The probability of correctly predicting the word w_i given the word w_j is defined as

$$p\left(w_i|w_j\right) = \frac{\exp({u_{wi}}^{\tau} v_{wj})}{\sum_{t=1}^{V} \exp\left({u_l}^{\tau} v_{wj}\right)} \tag{2}$$

Where, V is the number of words in the vocabulary. This formulation is expensive because the cost of computing $\nabla \log p\left(w_i|w_j\right)$ is proportional to the number of words in the vocabulary V. An efficient alternative to the full softmax is the hierarchical softmax, which greatly reduces the complexity of computing $\log p\left(w_i|w_j\right)$.

3.3 Determining distributional and semantic similarity

3.3.1 Distributional similarity

In the distributional semantics approach, the similarity between words can be quantified by how frequently they appear within the same context in Brown Corpus. These distributional properties of the words are described by a vector- space model where each word is associated with its context vector. The way a context is defined can vary in different applications. The one we use here is the most common approach which considers contexts of a word as a set of all other words with which it co-occurs. In 2-grams, only words that occur right next to the given word are considered as part of its context. The words and their context vectors are used to form a co-occurrence matrix, where row elements are target words and column elements are context terms.

The scores of the constructed co-occurrence matrix are given by Local Mutual Information (LMI) scores computed on the frequency counts of corresponding 2-grams. If words w_1 and w_2 have occurred $C(w\_1, w\_2)$ times together and the counts of number of times words w_1 and w_2 occur individually in the entire corpus is denoted by $C(w\_1)$ and $C(w\_2)$, then local mutual information score is defined as follows:

$$LMI = C(w\_1, w\_2) \,.\, \log_2 \frac{C(w\_1, w\_2)}{C(w\_1)\,C(w\_2)} \tag{3}$$

where N is the overall number of 2-grams in the corpus.

Given the words w_1 and w_2 their distributional similarity is then measured as the cosine product of their context vectors v_2, v_2. It can be represented as,

$$sim(w\_1, w\_2) = cos(v\_2, v\_2) \tag{4}$$

.

This model is applied to measure similarity of word occurrences in two *corpora* of different time periods in the following way. The set of context elements is fixed and remains the same for both *corpora*. Using each *corpus*, a context vector for a word is extracted independently. In this way, each word has a 60s vector and a 90s vector, with the same dimensions, but different co-occurrence counts. The vectors can be compared by computing the cosine of their angle. Since the context vectors are computed in the same vector space, the procedure is same as calculating similarity between two different words in the same *corpora*; the context vectors can be considered as belonging to one co-occurrence matrix and corresponding to two different row elements having word 60s and word 90s.

The procedure explained above is used to measure the semantic change of a word in two corpora that are considered, and therefore between two-time periods. High similarity value (close to 1) would suggest that a word has not undergone semantic change, while obtaining low similarity (close to 0) should indicate a noticeable change in the meaning and the use of the word. Distributional similarity means that words in similar contexts tend to have the same or related meanings.

Considering the rows of the generated matrix of frequencies, each instance $a_x i_y$ is represented by an instance related feature vector denoted by $B'_{a_x i_y} = \left\{fv_{y1}, fv_{y2}, fv_{y3}, \ldots, fv_{yp}\right\}$, where fv_{yh} represents the co-occurrence value when an instance $a_x i_y$ of an ambiguous word a_x appears together with rf_h, where $1 \leq h \leq p$, p is the total number of distinct related features for instance $a_x i_y$. The Instance related feature vector of instance is denoted by

$$B' = \left\{fv_{11}, fv_{12}, fv_{13}, \ldots, fv_{1p}\right\}. \tag{5}$$

From instance related feature vectors, each co-occurrence value fv_{yh} is used to calculate the degree of association between the instance $a_x i_y$ and the related feature rf_h. This association is computed by applying the Pointwise Mutual Information (PMI) function [20],

$$PMI(a_x i_y, rf_h) = \log_2 \frac{P(a_x i_y, rf_h)}{P(a_x i_y, *)P(*, rf_h)} \tag{6}$$

where $P(a_x i_y, *)$ represents the sum of occurrences of $a_x i_y$ over all the features rf_h, where $1 \leq h \leq p$. $P(*, rf_h)$ represents the sum of occurrences of rf_h over all instances $a_x i_y$ of the ambiguous word a_x, where $1 \leq y \leq m$, m is the total number of instances of the ambiguous word a_x. Co-occurrence values are obtained from the matrix of frequencies, where $fv_{a_x i_y, rf_h}$ denotes the co-occurrence frequency of $a_x i_y$ and rf_h, thus $P(a_x i_y, rf_h) = fv_{a_x i_y, rf_h}$.

The PMI function compares the number of occurrences between $a_x i_y$ and rf_h with the number of occurrences that $a_x i_y$ and rf_h have independently. Thus, a matrix of weights (Q) is generated from the matrix of frequencies of related features.

The proposed approach also computes the semantic similarity between such words and the instance of ambiguous word. If the words of a *synset* correspond to related or tested feature of the instance then the score of such *synset* is increased. After obtaining related and tested features from the auxiliary and test *corpus* respectively, the matrices of weights are generated. The matrix of weights Q is used to measure the semantic similarity between the instance of the ambiguous word and each word from *synsets*. The matrix of weights Z (from test features) is used to seek the association degree between an instance of the ambiguous word and each word from synsets. If both words share a large number of features, the semantic similarity between both words will be large. Among the two methods, the second one is much better than the other.

3.3.2 Semantic similarity

Semantic similarity value reflects the semantic relation between words. The proposed approach uses a supervised corpus-based technique relying on statistical associations to calculate semantic similarity between words. Such information is extracted from the matrix of weights Q of related features because the auxiliary corpus has more contexts for ambiguous words than the test corpus. This is achieved by means of the cosine of pointwise mutual information similarity function given in Equation (6), where a_xi_1 and a_xi_2 are instances of ambiguous words (rows) from the matrix of weights Q, but can be any pair of words; values closer to one indicate more similarity whereas values close to zero represent less similarity. The semantic similarity is now computed by using Equation (7). The degree of association between the ambiguous word a_x and the test feature t_j is added to the semantic similarity of a_x and t_j to obtain the final score of synset s_k

$$sim_{\cos PMI}(a_xi_1, a_xi_2) = \frac{A}{\sqrt{\sum_{rf_j \in Q[a_xi_1] \cap Q[a_xi_2]} PMI(a_xi_1, rf_j)^2 C}} \tag{7}$$

$$A = \sum_{rf_j \in Q[a_xi_1] \cap Q[a_xi_2]} PMI(a_xi_1, rf_j) PMI(a_xi_2, rf_j) \tag{8}$$

$$C = \sqrt{\sum_{rf_j \in Q[a_xi_1] \cap Q[a_xi_2]} PMI(a_xi_2, rf_j)^2} \tag{9}$$

The distributional similarity and the semantic similarity scores evaluated for the words are given as the inputs in the CNN for the context classification.

3.4 CNN-based context classification

Based on the similarity scores evaluated from the ambiguous words, the context classification is performed by the CNN. Every node in the hidden layer of CNN is joined with a region of a fixed window size in the input layer. Weight sharing is used for all the regions of the input layer starting from the fixed region to the sub network of the hidden layer. Equation- (10) represents the formula from the input layer to the hidden layer. In order to utilize semantic vector representation method, CNN adopts pooling technology to compress the hidden layers of uncertain lengths into hidden layers of fixed lengths. Max-pooling [27] is used here and the formula for max pooling is expressed in Equation- (12).

$$x_i = [e(w_{i-[win/2]})....e(w_i)....e(w_{i+[win/2]})] \tag{10}$$

$$h_i^{(1)} = \tanh(W_{xi} + b) \tag{11}$$

$$h^{(2)} = \max_{i=1}^{n} h_i^{(1)} \tag{12}$$

CNN can model the local information of each part in the text through its convolution kernel [15]. This network can integrate the full-context word sense from different local information sources through its pooling layer. The length of a sentence helps for determining the vector length. The convolutional layer of CNN is expressed with the activation function in Equation- 11. A window parameter h exists, and then consecutive h words in each group constitute a convolutional layer feature. One feature matrix is finally obtained by combining all the features. Max-pooling, is adopted in the next layer as expressed in Equation- 12. The architecture of enhanced CNN used for performing semantic classification and word sense disambiguation is shown in Figure- 4.

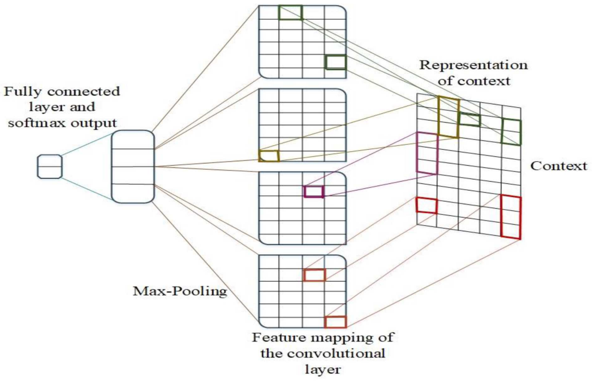

Fig. 4: Context classification using CNN Model with convolution, pooling and fully-connected Layers.

Max pooling selects the maximum Eigen value. This pooling operation basically solves the problem of non-uniform sentence lengths, and then the output lengths could be normalized. In the enhanced CNN model, the number of model filters is decided by the selected window length. The context classification results are then obtained by using 3 hidden layers. This model helps to realize a multi-label text classification system based on the full-text features. The output of this context classification is the retrieval of correct sensible sentence with respect to the target sentence. The most sensible class is estimated by processing the similarity scores. The algorithmic steps of the proposed method is given in Algorithm-1.

4 Results and Discussion

The testing data contains several target sentences for every noun. When a target sentence is given as an input, the CNN classifies the data based on the semantic similarity score evaluated from the extracted features. The training data contains various sentences with the same noun in different meanings. The CNN classifies the data and retrieves the sentence related to the target sentence.

4.1 Dataset Description

SemEval-2010 dataset is used in this work. The primary aim of SemEval-2010 WSI task is to allow comparison of unsupervised word sense induction and disambiguation systems. The target word dataset consists of 100 words, 50 nouns and 50 verbs. 80 percent of the data are used for the training process and 20 percent are used for the testing process. For each target word, participants are provided with a training set in order to learn the senses of that word. The evaluation framework of SemEval-2010 WSI task considers two types of evaluation. The first one is unsupervised evaluation and the second one is supervised evaluation.

4.2 Performance Analysis

In this section, a discussion on the experimental results obtained by using the proposed method and a few older methods on SemEval-2010 dataset is presented. A comparison of traditional prediction model such as K-Nearest Neighbour (KNN) with the proposed method has been carried out and the results are given in Table-1. The table represents the experimental results of various methods for some nouns. It has been proved that the best results are obtained by using the proposed approach.

In the graphs the sample nouns are denoted as follows. The word access is denoted as word1, address is denoted as word2, air is denoted as word3, road is denoted as word4, shape is denoted as word5 and tour is denoted as word6. Figure 5 shows the comparison graph of accuracy of various methods. Using our proposed method the accuracies for the words are 96% for the first word, 95% for the second word, 97% for the third and fourth words, 96% for the fifth word and 95% for the sixth word. While using the KNN method, the accuracies for the words are less. Figure- 6 shows the comparison

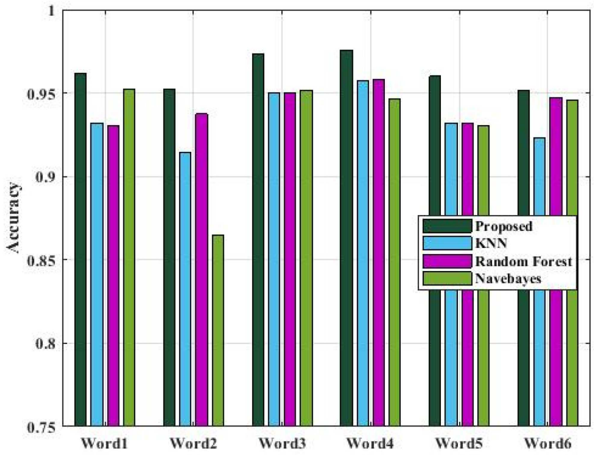

Fig. 5: Comparison graph of Accuracy.

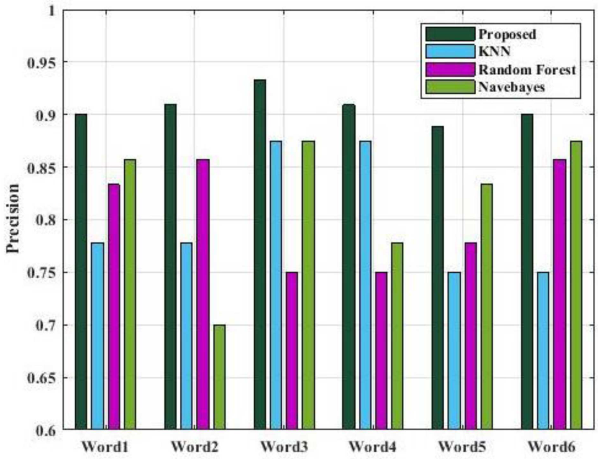

Fig. 6: Comparison graph of Precision.

Algorithm 1: Algorithmic steps for the proposed methodology

Step 1:	Pre-processing is performed by using stop word removal and stemming methods.
Step 2:	Features are extracted from the words by using word embedding along with continuous bag-of-words and skip gram model.
Step 3:	Distributional similarity score is evaluated by calculating the local mutual information value and point wise mutual information value.
Step 4:	Semantic similarity score is evaluated by considering the point wise mutual information value.
Step 5:	Both the distributional similarity and the semantic similarity scores are given as the inputs for the context classification.
Step 6:	Enhanced model is utilized for the classification of context.
Step 7:	The output of the CNN classifier is the correct sensible sentence with respect to the target sentence.

Table 1: Comparison of results got by using various methods

F-measure	Recall	Precision	Accuracy	Evaluation Parameters	Nouns
0.85	0.90	0.86	0.95	Naive Bayes	access
0.76	0.93	0.92	0.93	Random Forest	
0.82	0.95	0.96	0.93	KNN	
0.90	0.97	0.98	0.96	Proposed method	
0.73	0.85	0.83	0.86	Naive Bayes	address
0.85	0.88	0.90	0.93	Random Forest	
0.82	0.92	0.94	0.91	KNN	
0.90	0.95	0.96	0.95	Proposed method	
0.82	0.83	0.81	0.95	Naive Bayes	air
0.80	0.86	0.88	0.95	Random Forest	
0.82	0.91	0.92	0.95	KNN	
0.93	0.94	0.93	0.97	Proposed method	
0.77	0.80	0.79	0.94	Naive Bayes	road
0.80	0.83	0.85	0.95	Random Forest	
0.82	0.89	0.90	0.95	KNN	
0.90	0.92	0.91	0.97	Proposed method	
0.76	0.74	0.75	0.93	Naive Bayes	shape
0.82	0.82	0.84	0.93	Random Forest	
0.80	0.86	0.88	0.93	KNN	
0.88	0.90	0.89	0.96	Proposed method	
0.71	0.80	0.79	0.94	Naive Bayes	tour
0.85	0.81	0.84	0.94	Random Forest	
0.80	0.83	0.84	0.92	KNN	
0.90	0.86	0.87	0.95	Proposed method	

graph of precision of various methods. It shows that the proposed method has attained the highest precision value. Figure- 7 represents the comparison graph of recall. It shows that the recall value is high for the proposed method when compared to the other methods. Figure- 8 represents the comparison graph of f-measure. It shows that the f-measure value is high for the proposed method when compared to the other methods.

From the above results it is inferred that our proposed method is found to be performing better than KNN method.

5 Conclusion

The proposed multi-perspective knowledge discovery method for word sense disambiguation utilizing deep learning method helps in discovering the precise sense of ambiguous words in text. Using this proposed method the context classification is achieved with the best accuracy. SemEval-2010 dataset is used in this work. The results produced by applying the proposed approach tested on the standard dataset have shown that the disambiguation performance of the knowledge discovery approach is

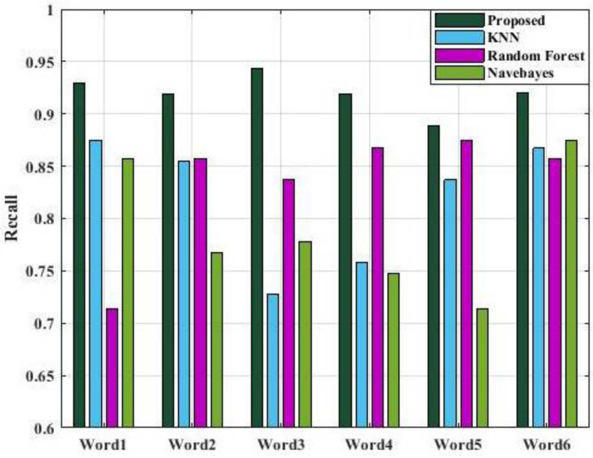

Fig. 7: Comparison graph of Recall.

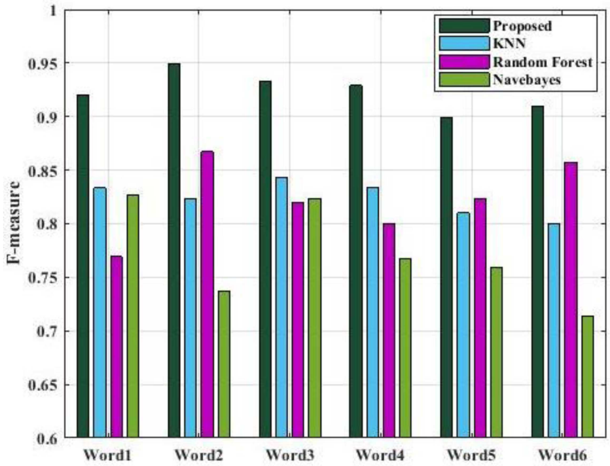

Fig. 8: Comparison graph of F-measure.

improved when compared to earlier methods such as KNN, Random Forest and Naïve Bayes in terms of Precision, Recall, accuracy and f-measure. The experimental results have shown that our proposed method attains the highest accuracy of 97%. In future, modified CNN models and Recurrent Neural Network could be used for resolving the WSD problem.

References

[1] S. Abualhaija and K. H. Zimmermann, D-Bees: A novel method inspired by bee colony optimization for solving word sense disambiguation, Swarm and Evolutionary Computation, 27, 188-195 (2016).

[2] Eneko Agirre, Lluis Marquez and Richard Wicentowski, Computational semantic analysis of language: SemEval-2007 and beyond, Language Resources and Evaluation, 43,2, 97-104 (2009).

[3] Wojdan Alsaeedan, Mohamed El Bachir Menai and Saad Al-Ahmadi, A hybrid genetic-ant colony optimization algorithm for the word sense disambiguation problem, Information Sciences, 417, 20-38 (2017).

[4] Bartosz Krawczyk and Bridget T. McInnes, Local Ensemble Learning from Imbalanced and Noisy Data for Word Sense Disambiguation, Pattern Recognition, 1-31 (2017).

[5] Tolga Bektas, P. Repoussis Panagiotis, and Christos D. Tarantilis, Dynamic vehicle routing problems, Vehicle Routing: Problems, Methods, and Applications, 18, 299 (2014).

[6] Leonora Bianchi, Marco Dorigo, Luca Maria Gambardella and Walter J. Gutjahr, A survey on metaheuristics for stochastic combinatorial optimization, Natural Computing, 8, 239-287 (2009).

[7] Davide Buscaldi and Paulo Rosso, A conceptual density-based approach for the disambiguation of toponyms, International Journal of Geographical Information Science, 22, 301-313 (2008).

[8] Ping Chen, Chris Bowes, Wei Ding and Max Choly, Word sense disambiguation with automatically acquired knowledge, IEEE Intelligent Systems, 27, 46-55 (2012).

[9] Issmail Ellabib, Paul Calamai and Otman Basir, Exchange strategies for multiple ant colony system, Information Sciences, 177,5, 1248-1264 (2007).

[10] S.M. Fakhr Ahmad, M.H. Sadreddini and M. Zolghadri Jahromi, A proposed expert system for word sense disambiguation: deductive ambiguity resolution based on data mining and forward chaining, Expert Systems, Apr 1; 32,2, 178-191 (2015).

[11] Amira Gherboudj, Abdesslem Layeb and Salim Chikhi, Solving 0-1 knapsack problems by a discrete binary version of cuckoo search algorithm, International Journal of Bio-Inspired Computation, 4,4, 229-236 (2012).

[12] Qinglin Guo and Ming Zhang, Question answering based on pervasive agent ontology and Semantic Web, Knowledge-Based Systems, 22, 6, 443-448 (2009).

[13] ChukFong Ho, Masrah Azrifah Azmi Murad, Shyamala Doraisamy and Rabiah Abdul Kadir, Extracting lexical and phrasal paraphrases: a review of the literature, Artificial Intelligence Review, 42,4, 851-894 (2014).

[14] Chihli Hung and Shiuan-Jeng Chen, Word sense disambiguation based sentiment lexicons for sentiment classification, Knowledge-Based Systems, 110, 224-232 (2016).

[15] Ren Kai and Wang Shi-Wen, Applying Convolutional Neural Network Model and Auto-expanded Corpus to Biomedical Abbreviation Disambiguation, Journal of Engineering Science & Technology Review, 9,6, 178-184 (2016).

[16] Angli Liu and Katrin Kirchhoff, Context Models for OOV Word Translation in Low-Resource Languages, in Proc. AMTA 2018, vol. 1: MT Research Track (2018).

[17] Francesco Mondada, Luca Maria Gambardella, Dario Floreano, Stefano Nolfi, J-L. Deneuborg, and Marco Dorigo, The cooperation of swarm-bots: Physical interactions in collective robotics, IEEE Robotics & Automation Magazine, 12,2, 21-28 (2005).

[18] Arturo Montejo-Raez, Eugenio Martinez-C amara, M. Teresa Martin-Valdivia and L. Alfonso Urena-Lopez, Ranked wordnet graph for sentiment polarity classification in twitter, Computer Speech & Language, 28, 1, 93-107(2014).

[19] Roberto Navigli, Word sense disambiguation: A survey, ACM Computing Surveys (CSUR), 41, 2, 1-69 (2009).

[20] Roberto Navigli, and Mirella Lapata, An experimental study of graph connectivity for unsupervised word sense disambiguation, IEEE transactions on pattern analysis and machine intelligence, 32,4, 678-692 (2010).

[21] Kiem-Hieu Nguyen and Cheol-Young Ock, Word sense disambiguation as a traveling salesman problem Artificial Intelligence Review, 1-23 (2013).

[22] Daniel Preotiuc-Pietro and Florentina Hristea, Unsupervised word sense disambiguation with N-gram features, Artificial Intelligence Review, 41,2, 241-260 (2014).

[23] Tinghua Wang, Wei Li, Fulai Liu, and Jialin Hua, Sprinkled semantic diffusion kernel for word sense disambiguation, Engineering Applications of Artificial Intelligence, 64, 43-51 (2017).

[24] Tingting Wei, Yonghe Lu, Huiyou Chang, Qiang Zhou, and Xianyu Bao, A semantic approach for text clustering using WordNet and lexical chains, Expert Systems with Applications, 42, 4, 2264-2275 (2015).

[25] Thomas Weise, Hendrik Skubch, Michael Zapf and Kurt Geihs, Global optimization algorithms and their application to distributed systems, 81, 1-69 (2008).

[26] T. Mikolov, K. Chen, G. Corrado G and J. Dean, Efficient Estimation of Word Representations in Vector Space, in Proc. of the Workshop at International Conference on Learning Representations, Scottsdale, AZ, USA, 1-12, (2013).

[27] J. L. Elman, Finding structure in time, Cognitive science, 14,2, 179-211 (1990).

S. Rajini has completed MS (By Res.) from Anna University-Chennai and is working as Associate Professor in Computer Science and Engineering Department of Kumaraguru College of Technology, Coimbatore, India. She has over two decades of teaching experience. She has published many research articles in journals and conferences. Her areas of interest are Distributed Computing and Natural Language Processing.

A. Vasuki has more than 26 years of experience in teaching, research and administration. She is currently a Professor of Mechatronics Engineering at Kumaraguru College of Technology, Coimbatore. She has published 3 book chapters, 28 papers in International journals and over 60 papers in National and International conferences. She has completed three funded research projects under RPS, AICTE and RESPOND, ISRO. She is currently guiding several research scholars for the doctoral programme. Her areas of interest are Signal / Image Processing and Communications.