Hypernymy Detection for Vietnamese Using Dynamic Weighting Neural Network

Bui Van Tan[1], Nguyen Phuong Thai[2] and Pham Van Lam[3]

[1] University of Economic and Technical Industries, Hanoi, Vietnam
bvtan@uneti.edu.vn
[2] VNU University of Engineering and Technology, Hanoi, Vietnam
thainp@vnu.edu.vn
[3] Institute of Linguistics, Vietnam Academy of Social Sciences, Hanoi, Vietnam
phamvanlam1999@gmail.com

Abstract. The hypernymy detection problem aims to identify the "is-a" relation between words. The problem has recently been receiving attention from researchers in the field of natural language processing. So far, fairly-effective methods for hypernymy detection in English have been reported. Studies of hypernymy detection in Vietnamese have not been reported yet. In this study, we applied a number of hypernymy detection methods based on word embeddings and supervised learning for Vietnamese. We propose an improvement on the method given by Luu Tuan Anh et al. (2016) by weighting context words proportionally to the semantic similarity between them and the hypernym. Based on Vietnamese WordNet, three datasets for hypernymy detection were built. Experimental results showed that our proposal can increase the efficiency from 8% to 10% in terms of accuracy compared to the original method.

Keywords: hypernymy detection, taxonomic relation, lexical entailment.

1 Introduction

Hypernymy is the relationship between a generic word (hypernym) and its specific instance (hyponym), For example, *vehicle* is a hypernym of *car* while *fruit* is a hypernym of *mango*. This relationship has recently been studied extensively from different perspectives in order to develop the mental lexicon [1]. In addition, hypernymy are rarely also referred to as the taxonomic [2], is-a [3] or inclusion relations [1]. Hypernymy is the most basic relation in many structured knowledge such as WordNet [4], BabelNet [5].

In natural form, nouns in Vietnamese usually have information in type, although this type of classification may be direct, indirect, and not multi-level. At the highest level, they are: *cây<tree>*, *con<child>* and so on, nouns play the role of determining the type. In Vietnamese nouns, the leading elements are typed elements, and these are the elements of the higher order (hypernym); for example:

xe<vehicle> - *xe_đạp<bicycle>*;

xe_đạp<bicycle>– xe_đạp_điện<electric bicycle>;
xe<vehicle> - xe_đạp_điện<electric bicycle>.

The classification of subordinate compounds is very clear in Vietnamese. When noun is coordinated compound, many cases of classification values are also expressed, for example:

cây_cỏ<plants> = thực_vật<plants>;
cây_con<creature> = thực_thể_sinh_học<biological entity>;
trâu_bò<buffalo_cow> = động_vật_kéo<cattle>.

In contrast, this method is normally not applied for ordinary words in English, if used, the words grafted are usually only descriptive value for the original word, but rarely turn the ordinary word to the hypernym. Compound method in English can be used a bit more in scientific terminology structure.

From a computational point of view, automatic hypernymy detection is useful for NLP tasks such as taxonomy creation [6],[7], recognizing textual entailment [8], and text generation [9], among many others. A good example is presented in [10], to recognize entailment between sentences, firstly, it must recognize the hypernymy between words; for example: *George was bitten by a dog* → *George was attacked by an animal, bitten* is hyponym of *attacked,* and *dog* hyponym of *animal*.

According to Peter Turney [10], the solution for this issue is usually based on three approaches such as: i) the methods based on context inclusion hypothesis [11], [12]; ii) the methods based on the context combination hypothesis [13]; iii) the method based on similarity differences hypothesis [14]. Another classification, the previous methods for this problem can be generally divided into two categories such as: statistical and linguistic approaches and both of them relying on word vector representation [2].

Word embeddings such as GloVe and Word2Vec have shown promise in a variety of NLP tasks. These word's representations are constructed to minimize the distance between words with similar contexts. According to the distributional similarity hypothesis [11], it was reported that similar words should have similar representations. However, they made no guarantees about more fine-grained semantic properties [15]. Recently, word embeddings has been exploited in conjunction with supervised learning to detect relations between word pairs. Yu et al. [16] propose a simple yet effective supervision framework to identify hypernymy relations using distributed term representations. First, they designed a distance-margin neural network to learn term embeddings based on some pre-extracted hypernymy data. Then, they applied such embedding as term features to identify positive hypernymy pairs through a supervision method. However, the term embedding learning method proposed [16] only learns through the pairwise relations of words without considering the contextual information between them. The recent studies [17],[18],[19] showed that contextual information between hypernym and hyponym is an important indicator to detect hypernymy relations. Tuan et al., (2016) proposed a dynamic weighting neural network to learn term embedding based on not only the hypernym and hyponym terms, but also the contextual information between them [2]. It should be noted that the context words are weighted equally in this model.

In this study, we propose an improvement of the word embedding model which was reported in [2] by weighting context words. We then apply the identified embedding as features to hypernymy detection using the supervised method support vector machine. Currently, there are neither studies on hypernymy detection nor datasets published for Vietnamese. Therefore, three datasets for hypernymy detection were built and published. Experimental results demonstrated that our proposal can increase the performance compared to the original method.

2 Related Work

Hypernymy detection problem is set out for a pair of word (u, v), determine whether word u is a hypernym of v or not. Previous studies on this problem can be categorized into two main approaches: statistical learning and linguistic pattern matching [2]. Some recent case studies have been published based on distributional representation [21],[22]. Linguistic approaches rely on lexical-syntactic patterns [23], [24].

Recently, Omer Levy et al. [18] pointed out that using linear SVMs, as foregoing work has done, reduces the classification task to that of predicting whether in a pair of words, the second one has some general properties associated with being a hypernym [18]. Some studies on hypernymy relation detection using word embeddings (i.e. Word2Vec and GloVe) as the input attributes for SVM [25], [26]. Several studies have proposed new neural network models, Yu et al. (2015) proposed a dynamic margin model to learn term embeddings based on pre-extracted taxonomic relation data [16]. However, Yu's model only use pairs of hypernymy separated pairs without considering the contextual information between them. In order to improve Yu's model, Luu Tuan Anh proposed a dynamic weighting neural network that uses contextual information for training, training data is a set of triples (hypernym, hyponym, context words) [2]. Another notable publication is the hierarchical embedding model for hypernymy detection and directionality [27].

The approach that is closest to our work is proposed by Luu Tuan Anh et al. (2016) [2]. However, in this model, context words are weighted equally. We assume that the role of context words is uneven; words that have sematic similarity are large with higher hypernym, the weight assigned to them must be greater.

3 The Proposed Approach

According to Tuan Anh Luu's approach [2] (DWN model), the role of context words is the same in a training sample, each word is assigned a coefficient $\frac{1}{k}$, whereas hyponym has the coefficient k to reduce the bias problem of high number of contextual words. Observation the triples extracted from the Vietnamese corpus, can see that some of them have high number of contextual words; the semantic similarity between each contextual word and the hypernym is different (Table 1). We assume that the role of contextual words is uneven, the word which has high semantic similarity with hypernym should be assigned greater weighting. Therefore, we estimate that the weight for contextual words is proportional to the semantic similarity between them and hypernym. Through this weighting, it is possible to reduce the bias of many contextual words that they themselves are less important.

Table 1. Some triples.

Sentence	Hypernym –Hyponym	Context words
Một trong những loài ***hoa*** *có gai nhọn, có nhiều màu_sắc và hương_thơm quyến_rũ là* ***hoa_hồng*** <One of the flowers that have sharp thorns, many colors and seductive fragrances is rose>	*hoa*<flower>-*hoa_hồng*<rose>	<*có gai nhọn, nhiều màu_sắc và hương_thơm quyến_rũ là*>
voi *là loài ăn thực_vật nên chúng thường sống ở khu_vực rừng nhiệt_đới có nhiều cỏ, chúng là loài* ***động_vật*** sống trên cạn to lớn nhất còn tồn_tại cho đến ngày_nay <elephants are herbivores so they live in tropical forests where there is a lot of grass, they are the largest terrestrial animals that have been alive until now>	*động_vật*<animal > -*voi*<elephant>	<*là loài ăn thực_vật nên chúng thường sống ở khu_vực rừng nhiệt_đới có nhiều cỏ, chúng là loài*>

In section 3.1, we present an improvement on DWN model, section 3.2 presentation using the support vector machine for hypernymy detection based on the word embeddings.

3.1 Learning word embeddings

In recent years, word embeddings have shown promise in a variety of NLP tasks. The most typical of these techniques is Word2Vec [20], with two models Skip-gram and Continuous bag of words (CBOW). The CBOW model is roughly the mirror image of the Skip-gram model, it is based on a predictive model, this model predicting the current word w_t from the context window of $2n$ words around it (Equation 1).

$$O=\frac{1}{T}\sum_{t=1}^{T}\log_{p}(w_t \mid w_{t-n},...,w_{t-1},w_{t+1},...,w_{t+n}) \quad (1)$$

The same as DWM model, our model consists of three steps: fisrt, extracting hypernymy pairs from Vietnamese Wordnet; second, extracting training triples from corpus; finally, training the neural network, in this step, for each of the triplets in the training set, we complement semantic similarity coefficient between contextual words with hypernym.

Vietnamese WordNet. WordNet is a lexical database for the English language [4]. Currently, Vietnamese WordNet (see Fig.1) has been constructed and applied quite effectively in studies on Vietnamese natural language processing [28]. Vietnamese WordNet contains 32,413 synsets, 66,892 words [1].

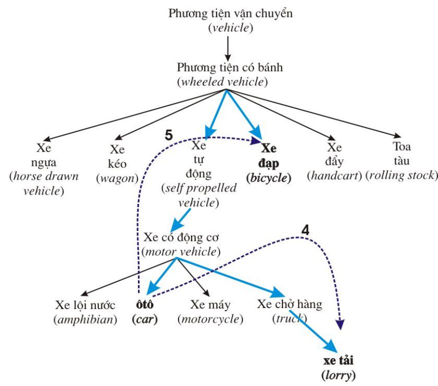


Fig.1. A fragment of the Vietnamese WordNet hypernym hierarchy

Semantic Similarity Measurement. To evaluate the semantic similarity level between contextual words and hypernym, we use the Lesk algorithm [29], a study [28] has shown that this algorithm gives the best results for the semantic similarity problem in Vietnamese. This algorithm proposed by Michael E. Lesk for word sense disambiguation problem can measure the similarity based on the gloss of words, with the hypothesis *two words are similar if the definition shares common words*. The similarity of a pair of word is defined as a function that overlaps the corresponding definitions (glosses) provided by a dictionary (Equation 2).

$$Sim_{Lesk}(w_1, w_2) = overlap(gloss(w_1), gloss(w_2)) \qquad (2)$$

In Vietnamese WordNet, *vợ*<*wife*>, *chồng*<husband> are defined as follows:

vợ: "*người phụ_nữ đã kết_hôn, trong quan hệ với người đàn_ông kết_hôn với mình*"<a married woman; a man's partner in marriage>

chồng: "*người đàn_ông đã kết_hôn, hôn phu của người phụ_nữ trong hôn nhân*"<a married man; a woman's partner in marriage>

Extracting Data. The purpose of this step is to extract a set of hypernymy pairs for training, a list of hypernymy pairs has been extracted from Vietnamese WordNet. As a result, the total number of hypernymy pairs is 269,781. After that, we extract the triples of hypernym, hyponym and the context words between them. Context words as all words located between the hypernym and hyponym in a sentence. Using the set of hypernymy pairs extracted from the first step as reference, we extract from the corpus all sentences which contain at least two words involved in this list. Corpus used in this study contains about 21 million sentences (about 560 million tokens), which are crawled from the internet and then filtered, standardized, and segmented. In total, we have extracted 2,985,618 training triples from this corpus including 138,062 hypernymy pairs.

In a triple <*hype, hypo, contextual words*>, with each contextual word x_{ct}, we define the coefficient α_t which is proportional to the semantic similarity between x_{ct} and hypernym. The word similarity is evaluated by the Lesk algorithm based on their glosses in Vietnamese WordNet, α_t defined in equation 3.

$$\alpha_t = \frac{Sim_{Lesk}(x_{ct}, hype)}{\sum_{i=1}^{k} Sim_{Lesk}(x_{ci}, hype)} \quad (3)$$

Note that: $\sum_{i=1}^{k} \alpha_i = 1$

Training Model

The word embeddings model proposed in [2] consists of three layers: input layer, hidden layer and output layer. The nodes on adjacent layers are fully connected. The vocabulary size is V, and the hidden layer size is N. The input layer has $k+1$ nodes, where each node is a one-hot *V*-dimensional vector. The weights between the input layer and hidden layer are represented by a $V \times N$ matrix W. Each row of W is a *N*-dimensional vector representation v_t of the associated word t of the input layer (see Fig. 2 [2]).

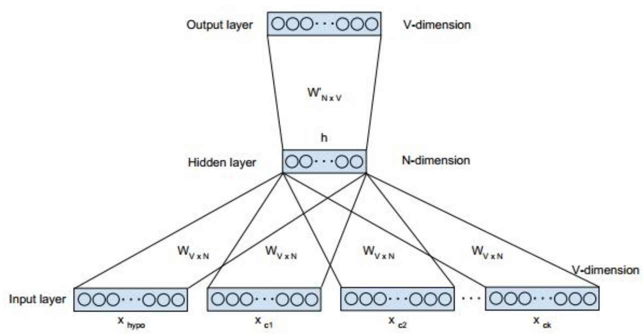


Fig. 2. The architechture of dynamic weighting neural netword model.

The target of the neural network is to predict the hypernym word from the given hyponym word and contextual words. Given a triple $< hype, hypo, c_1, c_2, ..., c_k >$ in the training data, x_{hypo}, $x_{c_1}, x_{c_2}, ..., x_{c_k}$ is one-hot *V*-dimensional vectors respectively. Denote $x_{contexts}$ as the summation vector of the context vectors, for each *k*-context word $x_{contexts}$ is calculated as follows:

$$x_{contexts} = \alpha_1 \times x_{c_1} + \alpha_2 \times x_{c_2} + ... + \alpha_k \times x_{c_k} \quad (5)$$

Let v_t denote the vector representation of the input word t, v_t and $v_{contexts}$ as follows:

$$v_t = x_t^{\mathrm{T}} W \quad (6)$$

$$v_{contexts} = x_{contexts}^{\mathrm{T}} \cdot W \quad (7)$$

The output of hidden layer h is calculated as:

$$h = \frac{v_{hypo} + v_{contexts}}{2} \quad (8)$$

From the hidden layer to the output layer, there is a different weight matrix W', which is a $N \times V$ matrix. Each column of W' is a *n*-dimensional vector $v_t^{'}$ representing the output vector of word t. Using these weights, we can compute a score u_t for each word in the vocabulary (Equation 9):

$$u_t = v_t^{'T}.h \quad (9)$$

Where $v_t^{'}$ is the j-th column of the matrix W' (the output vector of t). Then we use softmax, a log-linear classification model, to obtain the posterior distribution of hypernym word, which is a multinomial distribution (Equation 10).

$$p(hype \mid hypo, c_1, c_2, ..., c_k) = \frac{e^{u_{hype}}}{\sum_{i=1}^{V} e^{u_i}} = \frac{e^{v'^{T}_{hype} . \frac{v_{hypo} + v_{contexts}}{2}}}{\sum_{i=1}^{V} e^{v'^{T}_{i} . \frac{v_{hypo} + v_{contexts}}{2}}} \quad (10)$$

The objective function is then defined as:

$$O = \frac{1}{T} \sum_{t=1}^{T} \log(p(hype_t \mid hypo_t, c_{1t}, c_{2t}, ..., c_{kt})) \quad (11)$$

Herein, $t = < hype_t, hypo_t, c_{1t}, c_{2t}, ..., c_{kt} >$ is a sample in training data set T, $hype_t, hypo_t, c_{1t}, c_{2t}, ..., c_{kt}$ respectively hypernym, hyponym and contextual words. After maximizing the log-likelihood objective function in Equation 11 over the entire training set using stochastic gradient descent, the word embeddings are learned accordingly.

3.2 Supervised Hypernymy Detection

Recently, some studies using support vector machine (SVM) [30] for relation detection especially for hypernymy detection problem [18],[31]. In this work, SVM is also used to identify pair of words represented by embeddings vectors are hypernymy or not. Linear SVM is used because speed and simplicity, we used the Scikit-Learn[1] implementations with default settings. Inspired by the experiments of Julie Weeds et al.[22], some combinations of vectors are also experimental and reported.

4 Contruction of the Hypernymy Datasets for Vietnamese

The datasets play an important role in the field of relation detection problem, and construction of an accurate and valid dataset is a challenge[22],[32]. So far, the standard datasets for this problem in Vietnamese have not been published yet. For the purpose constructing a Vietnamese dataset, we refer some datasets which have been published for English[2].

---

[1] http://scikit-learn.org

[2] http://u.cs.biu.ac.il/~nlp/resources/downloads/lexical-inference-datasets/

Table 2: Some datasets.

Dataset	#Instances	#Positive	#Nagative
BLESS	14,547	1,337	13,210
ENTAILMENT	2,770	1,385	1,385
Turney 2014	1,692	920	772
Levy 2014	12,602	945	11,657

BLESS dataset: BLESS is a collection of examples of hypernyms, co-hyponyms, meronyms and random unrelated words for each of 200 concrete, largely monosemous nouns [32].

ENTAILMENT dataset: It consists of 2,770 pairs of terms, with equal number of positive and negative examples of hypernymy relation. Altogether, there are 1,376 unique hyponyms and 1,016 unique hypernyms [13].

Turney and Mohammad dataset: is based on a crowdsourced dataset of 79 semantic relations. Each semantic relation was linguistically annotated as entailing or not [14].

Levy dataset: is based on manually annotated entailment graphs of subject-verb-object tuples. This dataset is the most realistic dataset, since the original entailment annotations were made in the context of a complete proposition [18].

Analyze the differences between hypernymy in English and Vietnamese, based on the structure of published datasets for English, especially the criteria given by Julie Weeds [22] for a benchmark datasets, the requirements for a Vietnamese dataset are as follows:

The dataset should contain words that belong to different domains.

A dataset needs to be balanced in many respects in order to prevent the supervised classifiers making use of artefacts of the data.

There should be an equal number of positive and negative examples of a semantic relation.

The negative examples need to be pairs of equally similar words, but where the relationship under consideration does not hold.

The number of words in the dataset, should balance in classes (e.g. city, actor, ...) and instances (e.g. Paris, Tom Cruise, ...).

To visualize the structure of the *Vds1*, *Vds2* and *Vds3* datasets[3], they are represented graph structure. The vertex is a word, the edge of graph is a pair of word in dataset (see Fig. 3, 4).

Vds1 dataset: The words of this dataset are selected from Vietnamese WordNet and they belong to different domains: plants, animals, furniture, foods, materials, vehicles and others. Each pair of word (u,v) in the dataset is assigned one of the three semantic relation labels.

Hypernym: u is hypernym of v, (e.g. hoa<flower> - hoa_hồng<rose>).

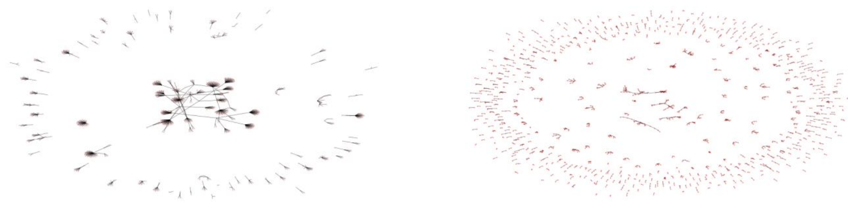
Co-hyponym: u that is a co-hyponym (coordinate) of v, (e.g. hoa_hồng<rose>-hoa_hướng_dương<sunflower>).

Random: u has no hypernym or co-hyponym relation with v, (e.g. hoa<flower> – xe_đạp<bicycle>).

Vds2 dataset: This dataset consists of 1,657 hypernymy pairs which are chosen from 269,781 hypernymy pairs extracted from Vietnamese WordNet (Table 3). Fig. 3a

[3]https://github.com/BuiVanTan2017/Vhypernymy

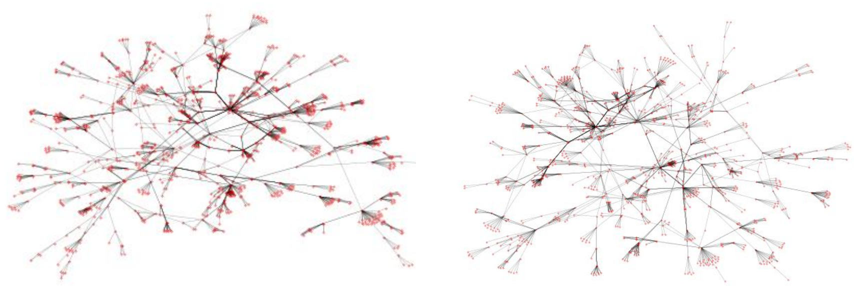
shows that the Vds1 dataset contains hypernymy pairs and they belong to some domains, some words share a hypernym forming tree structure. In contrast, Fig. 3b shows that most of the hypernymy pairs are disjoint pairs, because they are randomly selected from WordNet Vietnamese.



a – Vds1 dataset b – Vds2 dataset

Fig. 3. Visualization of the datasets

Vds3 dataset: We extracted from Vietnamese WordNet two subnets. The first subnet contains of hypernymy pairs extracted from the taxonomy tree, which is a subtree with the root node as động_vật <animal>(Vds3$_{animal}$); The second subnet is a subtree with the root node as thực_vật <plant> (Vds3$_{plant}$). In other words, these subnets are taxonomy trees. The height of tree which corresponds to Vds3$_{animal}$ is 12, and contains 2,284 hypernymy pairs. For Vds3$_{animal}$, the height of tree is 9 and contains 2,267 hypernymy pairs. Fig. 4 visualizes two subnets, Fig. 4a shows Vds3$_{animal}$ and Fig. 4b shows Vds3$_{plant}$. The number of pairs for each relation from the three datasets are summarized in Table 3.



a - động_vật<animal> b - thực_vật<plant>

Fig. 4. Visualization of subnets.

Table 3. Statistics of three datasets.

Dataset		Relation	#Instance	Total
Vds1		hypernymy	976	10285
		co-hyponym	8283	
		random	1026	
Vds2		hypernymy	1657	3314
		random	1657	
Vds3	động_vật<animal>	hypernymy	2284	2284
	thực_vật<plant>	hypernymy	2267	2267

5 Experimental Setup

We conduct experiments to evaluate performance of improved method compared to other methods. It proves that our improvement on Luu Tuan Anh' model enhances performance of hypernymy detection in Vietnamese. Three techniques of word embeddings are implemented: Word2Vec[4] model [20], DWM [2], and our improved DWM model (our). Training the Word2Vec model in Vietnamese, we use a corpus which contains about 21 million sentences (about 560 million words), we exclude from this corpus any word that appears less than 50 times. Data for training DWM and improved DWM model has 2,985,618 triples and 138,062 individual hypernymy pairs which are extracted from the above corpus. To decide whether word u is a hypernym of word v, we build a classifier that uses embedding vectors as features for hypernymy detection. Specifically, we use Support Vector Machine (SVM)[30] for this purpose. Inspired by the experiments of Julie Weeds et al. [22], some combinations of vectors are also experimental and reported.

Table 4. Some combinations of vectors.

svmDIFF	A linear SVM trained on the vector difference $v_{hype} - v_{hypo}$
svmMULT	A linear SVM trained on the pointwise product vector $v_{hype} \oplus v_{hypo}$
svmADD	A linear SVM trained on the vector sum $v_{hype} + v_{hypo}$
svmCAT	A linear SVM trained on the vector concatenation $v_{hype} \oplus v_{hypo}$
svmCATs	A linear SVM trained on the vector concatenation $v_{hype} \oplus v_{hypo} \oplus (v_{hype} - v_{hypo})$

Hereafter, the experiments were conducted on three datasets Vds1, Vds2 and Vds3.

Experiment 1. Experiment on Vds1 dataset, the data includes 976 hypernymy pairs (positive labels), and 1,026 pairs which are not hypernymy (negative labels), these pairs are mixed then selected 70% for training and 30% for testing. To increase the independence between training and testing sets, we exclude from the training set any pair of terms that has one word appearing in the testing set. The results shown in Table 5 are the accuracy of methods when using different combinations of vectors.

[4] *http://code.google.com/p/word2vec/*

Table 5: Performance results for the Vds1 dataset.

Dataset	model	svmDIFF	svmMULT	svmADD	svmCAT	svmCATs
Vds1	Word2Vec	0.82	0.77	0.81	0.80	0.79
	DWM	0.81	0.79	0.82	0.82	0.84
	Our	0.86	0.83	0.84	0.87	0.89

The experimental results in Table 5 show that improved method performs better than Word2Vec and DWM methods in accuracy. svmDIFF gives better results for Word2Vec model, but performance of DWM and improved method is higher than with svmCATs.

Experiment 2. Experiment on Vds2 dataset, the data includes 1,657 hypernymy pairs (positive labels), and 1,657 pairs which are not hypernymy (negative labels), the same as experiment 1, these pairs are mixed then selected 70% for training and 30% for testing. To increase the independence between training and testing sets, we exclude from the training set any pair of terms that has one word appearing in the testing set. The results shown in Table 6 are the performance of methods that are measured in terms of precision, recall and F1.

Table 6. Performance results for the Vds2 dataset.

Dataset	Model	Precision	Recall	F1
Vds2	Word2vec	0.85	0.87	0.86
	DWM	0.88	0.88	0.88
	Our	0.90	0.94	0.92

Experiment 3. This experiment aims to evaluate the capacity of methods to recognize a subnet. Two subnets: $Vds3_{animal}$, $Vds3_{plant}$ respectively are used for training and testing data. in this experiment, svmCATs is used for combinations of vectors. Experimental results are presented in Table 7.

Table 7. Performance results for the Vds3 dataset.

Model	Training	Testing	Precision	Recall	F1
Word2vec	$Vds3_{animal}$	$Vds3_{plant}$	0.50	0.60	0.55
DWM			0.52	0.64	0.57
Our			0.61	0.76	0.68
Word2vec	$Vds3_{plant}$	$Vds3_{animal}$	0.58	0.72	0.64
DWM			0.57	0.73	0.64
Our			0.62	0.78	0.69

In the experimental parts 2 and 3, the precision can be characterized as the measurement of exactness or quality, whereas the recall is the measurement of completeness or quantity. As seen in Table 6 and 7, the improved method produced the better results than the original one, not only in term of the precision but also the recall.

6 Conclusion

In this work, a number of hypernymy detection methods based on word embeddings and supervised learning for Vietnamese, and make the following contributions. First, improved an word embeddings model by weighting contextual words proportionally to the semantic similarity between them and the hypernym. Experimental results demonstrated that our proposal can increase the efficiency from 8% to 10% in terms of accuracy compared to the original method. Second, based on Vietnamese WordNet, three datasets for hypernymy detection have been built and published. Based on the results from this work, we plan to expand WordNet using hypernymy detection method. Further studies how to construct a taxonomy from texts in Vietnamese, as well as recognizing textual entailment will be conducted in the future.

References

1. Phuong-Thai Nguyen, Van-Lam Pham, Hoang-Anh Nguyen, Huy-Hien Vu, Ngoc-Anh Tran, Thi-Thu Ha Truong: A Two-Phase Approach for Building Vietnamese WordNet, the 8th Global Wordnet Conference (2015).
2. Luu Anh Tuan, Yi Tay, Siu Cheung Hui, See Kiong Ng: Learning Term Embeddings for Taxonomic Relation Identification Using Dynamic Weighting Neural Network, Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 403–413, Austin, Texas (2016).
3. Julian Julian Seitner, Christian Bizer, Kai Eckert, Stefano Faralli, Robert Meusel, Heiko Paulheim and Simone Paolo Ponzetto: A Large Database of Hypernymy Relations Extracted from the Web. Proceedings of the 10th edition of the Language Resources and Evaluation Conference. Portorož, Slovenia (2016).
4. Christiane Fellbaum: WordNet: An Electronic Lexical Database. MIT Press (1998).
5. Roberto Navigli, Simone Paolo Ponzetto, BabelNet: Building a Very Large Multilingual Semantic Network, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 216–225, Uppsala, Sweden, 11-16 July (2010).
6. Rion Snow, Daniel Jurafsky, and Andrew Y Ng: Learning syntactic patterns for automatic hypernym discovery. Advances in Neural Information Processing Systems 17 (2004).
7. Roberto Navigli, Paola Velardi, and Stefano Faralli: A Graph-based Algorithm for Inducing Lexical Taxonomies from Scratch. Proceedings of the 20th International Joint Conference on Artificial Intelligence, pages 1872–1877 (2011).
8. Ido Dagan, Dan Roth, Mark Sammons, and Fabio Massimo Zanzotto: Recognizing Textual Entailment: Models and Applications. Synthesis Lectures on Human Language Technologies (2013).
9. Or Biran and Kathleen McKeown: Classifying taxonomic relations between pairs of wikipedia articles. In Proceddings of Sixth International Joint Conference on Natural Language Processing (IJCNLP), pages 788–794, Nagoya, Japan (2013).
10. Turney, P. D., and Mohammad, S. M: Experiments with three approaches to recognizing lexical entailment. Natural Language Engineering 21(3):437–476 (2015).
11. Geffet, M., and Dagan, I.: The distributional inclusion hypotheses and lexical entailment. In Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (ACL 2005), pp. 107–114, Ann Arbor, MI (2005).

12. Kotlerman, L., Dagan, I., Szpektor, I., and Zhitomirsky-Geffet, M.: Directional distributional similarity for lexical inference. Natural Language Engineering, 16(4), 359–389 (2010).
13. Baroni, M., Bernardi, R., Do, N.-Q., and Shan, C.: Entailment above the word level in distributional semantics. In Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2012), pp. 23–32, Avignon, France (2012).
14. Turney, P. D., and Mohammad, S. M. 2015. Experiments with three approaches to recognizing lexical entailment. Natural Language Engineering 21(3):437–476. (2015)
15. N. Nayak: Learning hypernymy over word embeddings, (2015).
16. Zheng Yu, Haixun Wang, Xuemin Lin, and Min Wang: Learning term embeddings for hypernymy identification. Proceedings of the 24th International Joint Conference on Artificial Intelligence, pages 1390–1397 (2015).
17. Paola Velardi, Stefano Faralli, and Roberto Navigli: Ontolearn reloaded: A graph-based algorithm for taxonomy induction. Computational Linguistics, 39(3):665–707 (2013).
18. Omer Levy, Steffen Remus, Chris Biemann, Ido Dagan, and Israel Ramat-Gan: Do supervised distributional methods really learn lexical inference relations. Proceedings of the NAACL conference, pages 1390–1397 (2014).
19. Luu A. Tuan, Jung J. Kim, and See K. Ng.: Incorporating Trustiness and Collective Synonym/Contrastive Evidence into Taxonomy Construction. Proceedings of the EMNLP conference, pages 1013–1022 (2015).
20. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013a).
21. Stephen Roller, Katrin Erk, and Gemma Boleda: Inclusive yet selective: Supervised distributional hypernymy detection. Proceedings of the COLING conference, pages 1025–1036 (2014).
22. Julie Weeds, Daoud Clarke, Jeremy Reffin, David J Weir, and Bill Keller: Learning to distinguish hypernyms and co-hyponyms. Proceedings of the COLING conference, pages 2249–2259 (2014).
23. Marti A. Hearst.: Automatic acquisition of hyponyms from large text corpora. Proeedings of the 14th Conference on Computational Linguistics, pages 539–545 (1992).
24. Rion Snow, Daniel Jurafsky, and Andrew Y. Ng.: Semantic taxonomy induction from heterogenous evidence. In Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics (ACL), pages 801–808, Sydney, Australia (2006).
25. Liling Tan, Rohit Gupta, and Josef van Genabith: Usaar-wlv: Hypernym generation with deep neural nets. Proceedings of the SemEval, pages 932–937 (2015).
26. Ruiji Fu, Jiang Guo, Bing Qin, Wanxiang Che, Haifeng Wang, and Ting Liu: Learning semantic hierarchies via word embeddings. Proceedings of the 52nd Annual Meeting of the ACL, pages 1199–1209 (2014).
27. Kim Anh Nguyen, Maximilian Koper, Sabine Schulte im Walde, Ngoc Thang Vu, Hierarchical: Embeddings for Hypernymy Detection and Directionality, Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 233–243, Copenhagen, Denmark, September 7–11 (2017).
28. Bui Van Tan, Nguyen Phuong Thai, Pham Van Lam: Construction of a Word Similarity Dataset and Evaluation of Word Similarity Techniques for Vietnamese, Knowledge and Systems Engineering (KSE), 2017 9th International Conference on (2017).
29. M. Lesk: Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from a ice cream cone. In Proceedings of SIGDOC ’86, (1986).

30. Corinna Cortes and Vladimir Vapnik: Supportvector networks. Machine learning, 20(3):273–297 (1995).
31. Yogarshi Vyas, Marine Carpuat, Detecting Asymmetric Semantic Relations in Context: A Case Study on Hypernymy Detection, Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017), pages 33–43 (2017).
32. Marco Baroni and Alessandro Lenci: How we blessed distributional semantic evaluation. Proceedings of the GEMS 2011 Workshop on Geometrical Models of Natural Language Semantics, pages 1–10 (2011).
33. Lili Kotlerman, Ido Dagan, Idan Szpektor, and Maayan Zhitomirsky-Geffet: Direc ional Distributional Similarity for Lexical Inference.Natural Language Engineering 16(4):359–389 (2010).