Enhancing Performance of Lexical Entailment Recognition for Vietnamese based on Exploiting Lexical Structure Features

***Abstract*—The lexical entailment recognition problem aims to identify the *is-a* relation between words. The problem has recently been receiving research attention in the natural language processing field. In this study, we propose a novel method (VLER) for this problem on Vietnamese. For this purpose, we first exploit such lexical structure information of words as a feature, then combine this feature with vectors representation of words such as a unique feature for recognizing the relation. Moreover, we applied a number of methods based on word embedding and supervised learning, experimental results showed that our method achieves the best performance in the hypernymy detection task than other methods in terms of accuracy.**

***Index Terms*—lexical entailment, hypernymy detection, taxonomic relation, lexical entailment recognition**

I. Introduction

Word-level lexical entailment (LE) is an asymmetric semantic relation 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 (Nguyen et al., 2016). In addition, LE is also referred to as the taxonomic (Luu et al., 2016), is-a or hypernymy (Nguyen et al., 2017). LE is one of the most basic relations of many structured knowledge databases such as WordNet (Fellbaum, 1998), and BabelNet.

The LE has been applied effectively to many NLP tasks such as taxonomy creation (Snow et al., 2005), recognizing textual entailment (Dagan et al., 2013), text generation (Biran and McKeown). Actually, LE that is becoming a very important topic in NLP because of its applications for solving the NLP challenges of such as the metaphor detection (Mohler et al., 2013). Among many others, a good example is presented in (Turney and Mohammad, 2015) about recognizing entailment between sentences by identifying the lexical entailment relation between words, for example since *bitten* is a hyponym of *attacked*, and *dog* is a hyponym of *animal*, *"George was bitten by a dog"* and *George was attacked by an animal* have an entailment relation.

In recent years, word embeddings have established themselves as an integral part of NLP models, with its usefulness demonstrated across application areas such as parsing (Chen and Manning, 2014) , machine translation (Zou et al., 2013), temporal dimension (Hamilton et al., 2016), (Bamler and Mandt, 2017), relations detection (Levy et al., 2015), (Nguyen et al., 2017). Standard techniques for inducing word embeddings rely on the distributional hypothesis (Harris, 1954), this means that *similar words should have similar representations*. Using co-occurrence information from large textual corpora to learn meaningful word representations (Mikolov et al., 2013), (Levy and Goldberg, 2014), (Pennington et al., 2014), (Bojanowski et al., 2017). Recently, some methods have been proposed based on word embeddings which outperformed other approaches (Nayak, 2015), (Yu et al., 2015), (Luu et al., 2016); (Nguyen et al., 2017), (Vulic and Mrksic, 2017).

In the lexicon of languages, there are not only the single words but also the compound words which have many components[1]. In the technical vocabulary, concepts are practically compound words which are formed from two or more components. The Vietnamese WordNet is an example, concepts have two or more components are 92%, which have three or more components are 48%. Especially for the Vietnamese, words which have greater than more one component accounting for 70%. These words are created by two compound mechanisms that are subordinated compound and coordinated compound, they correspondingly create subordinated compound words and coordinated compound words. The semantic relationship between a word and the components of another word often manifests the lexical entailment relation of themselves. Consider a pair of words *hồng*<*rose*> and *hoa_hồng_bạch*<*white rose*>, Both words share *hồng* as the common component, we intuitively can recognize their relation that is the lexical entailment relation, some more examples presented in Table I. However, prior studies have not yet exploited this information such as a useful feature for recognizing the lexical entailment relation of the compound words as well as the good standard datasets which have been used for the experimental only consist of the single words.

In this paper, we introduce a novel method for the Vietnamese lexical entailment recognition problem. Our method based on a combination of specialisation word embeddings and the lexical structure (VLER). It is

[1]in this paper, single words in a compound word are called component instead of syllable because the meaning of syllable is dissimilarity in Vietnamese and English

inspired by the work (VDWN) which proposed by (Tan et al., 2018). This method represents our idea that combines between word vectors of VDWN model and lexical structure information features. These two components are combined in one attribute vector, then it is applied such as term features to identify positive hypernym-hyponym pairs using a supervised method. In addition, our method is compared with some other methods, the experimental results demonstrated that our model can give better results than others on overall three datasets which published in (Tan et al., 2018).

The rest of this paper is structured as follows. Section II presents some related methods. Section III describes our method. Section IV presents experimental results and evaluation. The last section gives conclusions.

II. Related work

The lexical entailment recognition problem is increasing attention because of its usefulness in downstream NLP tasks. Early work relied on asymmetric directional measures (Weeds et al., 2014) which were based on the distributional inclusion hypothesis (Geffet and Dagan, 2005a) or the distributional informativeness or generality hypothesis (Santus et al., 2014). However, these approaches have recently been superseded by methods based on word embeddings. In this approach, the methods can be divided two main groups: (1) these methods build dense real-valued vectors for capturing LE as well as their direction (Nguyen et al., 2017),(Vulic and Mrksic, 2017). (2) The methods use the vectors that gain from embedding models as features for supervised detection models (Luu et al., 2016), (Shwartz et al., 2016), (Tan et al., 2018).

Recently, (Yu et al., 2015) proposed a simple but effective supervised framework for identifying LE using distributed term representations. They designed a distance-margin neural network to learn word embeddings based on some pre-extracted LE data. Then, they applied word embeddings as features to identify positive LE pairs using a supervised method. However, the proposed method for learning term embedding did not consider the contextual information. Moreover, these studies (Levy et al., 2015), (Luu et al., 2015), (Velardi et al., 2013) showed that contextual information which between hypernym and hyponym is an important indicator to detect LE relations. (Luu et al., 2016) proposed a dynamic weighting neural network (DNW) to learn word embedding based on not only the hypernym and hyponym terms, but also the contextual information between them. The approach that is closest to our work is the one proposed by (Tan et al., 2018), it is an improved DWN method with an assumption that is context words should not be weighted uniformly. They assume that the role of contextual words is uneven, contextual words which are more similar to the hypernym can be assigned a higher weight. The method then to apply the word embedding as features for recognizing lexical entailment using support vector machine.

III. Methodology

A. The VDWN Framework

According to the DWN method (Luu et al., 2016), 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. By observing the triples extracted from the Vietnamese corpus, (Tan et al., 2018) pointed out that some of them have high number of contextual words; the semantic similarity between each contextual word and the hypernym is different. We assume that *the role of contextual words is uneven, words had higher semantic similarity with hypernym should be assigned a greater weight*. Therefore, we suppose that the weight for contextual words is proportional to the semantic similarity between them and hypernym. Through this weighting method, it is possible to reduce the bias of many contextual words that they themselves are less important.

To evaluate the semantic similarity between contextual words and hypernym, we use the Lesk algorithm (Lesk, 1986) which was 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 their definitions share common words. This algorithm is used because of the following reasons. Firstly, it only uses the brief definition of words in the dictionary instead of using the structural information of Vietnamese WordNet. Second, its performance is better than other knowledge-based methods. Furthermore, a study has shown that this algorithm gives the best results for the semantic similarity problem in Vietnamese (Tan et al., 2017). The similarity of a pair of words is defined as a function that overlaps the corresponding definitions (glosses) that are provided by the dictionary (Equation 1).

$$Sim_{Lesk}(w_1, w_2) = overlap(gloss(w_1), gloss(w_2)) \tag{1}$$

In a triple $< hype, hypo, contextual\ words >$, with each contextual word x_{ct}, we define a coefficient α_t is proportional to the semantic similarity between x_{ct} and $hype$ ($\sum_1^k \alpha_i = 1$, where k is the number of contextual words).

$$\alpha_t = \frac{Sim_{Lesk}(x_{ct}, hype)}{\sum_1^k Sim_{Lesk}(x_{ci}, hype)} \tag{2}$$

Denote $x_{contexts}$ as the summation vector of the context vectors, k-context word in each triple is calculated as follows:

$$x_{contexts} = \sum_1^k \alpha_i x_{c_i} \tag{3}$$

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

$$v_t = x_t^T W \tag{4}$$

$$v_{contexts} = x_{contexts}^T W$$

The output of hidden layer h is calculated as:

$$h = \frac{v_{hypo} + v_{contexts}}{2} \tag{5}$$

From the hidden layer to the output layer, there is a different weight matrix $W'_{N\times V}$. Each column of W' is a n-dimensional vector v'_t represents the output vector of word t. Using these weights, we can compute a score u_t for each word in the vocabulary:

$$u_t = v_h'^T.h \tag{6}$$

We use the Softmax function as a log-linear classification model to obtain the posterior distribution of hypernym word. In another word, it is a multinomial distribution (Equation 7).

$$\begin{aligned} p(hype|hypo, c_1, c_2, ..., c_k) &= \frac{e^{u_{hype}}}{\sum_1^V e^{u_i}} \\ &= \frac{e^{v_{hype}'^T \times \frac{v_{hypo} + v_{contexts}}{2}}}{\sum_1^V e^{v_i'^T \times \frac{v_{hypo} + v_{contexts}}{2}}} \end{aligned} \tag{7}$$

Then objective function is defined as:

$$O = \frac{1}{T}\sum_{t=1}^{T} Log(p(hype_t|hypo_t, c_{1t}, c_{2t}, ..., c_{kt})) \tag{8}$$

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 8 over the entire training set using stochastic gradient descent, the word embeddings are learned accordingly.

Both Word2vec and VDWN are prediction models. The word2vec model relies on the *distributional hypothesis* (Harris, 1954; Firth, 1957), ***in which words with similar distributions (shared context) have related meaning (have the same vector)***. The word2vec model predicts contextually when has target word on each training sample (Skip-gram), or vice versa (CBOW). Different from the Word2vec model, the VDWN model predicts a hypernym when has a hyponym and a context on each triple in the training corpus. ***According to this objective training, achieved vectors to obey a hypothesis, in which words have similar hyponyms and share contexts to have near vectors***.

Table I
SOME VIETNAMESE LE PAIRS

Hypernym	**Hyponyms**
xe<vehicle>	*xe_đạp<bicycle>, xe_ôtô_tải<lorry>, xe_đạp_điện<electrical_bicycle>,...*
hoa<flower>	*hoa_hồng<rose>, hoa_hồng_bạch<white_rose>, hoa_hồng_nhung<rose_velvet>,...*
rau<vegetables>	*rau_cải<brassica>, rau_cải_ngọt<brassica_integrifolia>,...*

B. The Lexical Structure Feature

In this study, we hypothesize that lexical structure information is useful for the LE recognition. How to create a feature vector that represents the correlate of the lexical structure between two words is the major purpose of our research, then combine this vector and embedding vectors, thereby enhancing the LE prediction results of the unsupervised learning model. According to our observation of English words, *if a pair of words which share some parts, then it tends to have LE relation*. For example, *student - computer_science_student*, *science - biological_science*,...

When observed LE pairs on Vietnamese, we can see that there is a strong relevance between the lexical structure information of two words in each LE pair. Vietnamese phrases contain classified information such as *cây<tree>*, *con<child>*,...(Table I).

To construct a vector represents the correlate of lexical structure between two words, we provide some definitions as follows.
Let V be the vocabulary, each word $w < w_1w_2w_3...w_n >$ denote $S(w)$ as a set of all component of w:

$$S(w) = \{x_i...x_j | i, j \in 1...n, i \leq j\} \tag{9}$$

for example (see Figure 1): *S(xe_đạp_điện<electrical_bicycle>) = {xe<vehicle>, xe_đạp<bicycle>, xe_đạp_điện<electrical_bicycle>, đạp<trample>, đạp_điện<electrical_motor>, điện<electric>}*; *S(computer_science) = {computer, computer_science, science*. we define $F_{lsc}(u, v)$ is an asymmetric function to measure the lexical structure correlation between u to v. The F_{lsc} is defined as:

$$F_{lsc}(u, v) = Max_{s_i \in S(v)}(Sim(u, s_i)) \tag{10}$$

where Sim is a function measured similarity between two words according to the fastText model used cosine distance, this model was selected because it can measure out of vocabulary words. Note that, the lexical structure correlation function that is asymmetric measurement, therefore: $F_{lsc}(u, v) \neq F_{lsc}(v, u)$ (Figure 1).

The vector contained lexical structure information feature of word pair can be expressed as:

$$V_{lsf} = \{F_{lsc}(u, v), F_{lsc}(v, u)\} \tag{11}$$

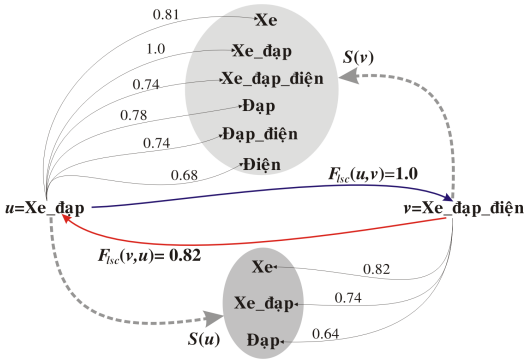

Figure 1. An illustration of lexical structure feature extraction method.

Here V_{lsf} is a vector established from two components $F_{lsc}(u, v)$ and $F_{lsc}(v, u)$. Though on experiments, these components are repeated in this vector for the following reasons: (1) when concatenation V_{lsf} with a k-dimensional word embedding then it must also be a k-dimensional vector; (2) to reduce the bias problem between V_{lsf} and the word embeddings vectors for the unsupervised learning method.

The features are extracted from a pair of words make the classification algorithm work more efficient, actually it is useful features. Therefore, the distribution of these features must be as strong classifiable as possible. To visualize classifiable ability of the lexical structure feature, we generate the V_{lsf} for pairs in the dataset, each of which is presented as a point in 2-dimensional space. In the figure 2, red points indicated for the V_{lsf} of pairs are labeled as negative, conversely blue points represent to the V_{lsf} of pairs are labeled as positive. As shown in the picture, the region which contains red points is separated from the blue region. Furthermore, there are many blue points that have x-coordinate or y-coordinate is 1. It represents for the pairs which have a component of a word that is completely similar to one of components of remaining word, that is these pairs have a denotation of the entailment relation strongly.

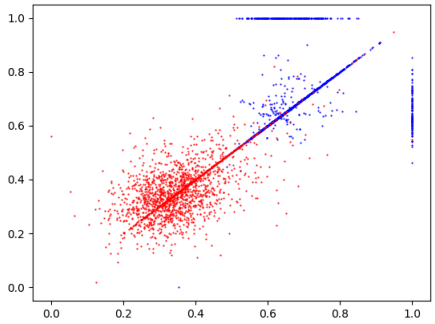

Figure 2. The visualization of the lexical structure feature vectors as points in the two-dimensional space

IV. Experimental Setup

This study's experiments used three standard datasets for LE recognition problem in Vietnamese which have been published in (Tan et al., 2018)[2]. Experiments focus on Vietnamese LE recognition. However, the proposed method can be easily adapted to other languages.

A. Datasets

Dataset plays an important role in the field of relation detection problem. The following we present statical information about the dataset used in this study.

Table II
Statistics of three datasets

Dataset	Relation	Instance	Total
$Vds1$	LE co-hyponym Random	976 8283 1026	10285
$Vds2$	LE Random	1657 1657	3314
$Vds3_{animal}$	LE	2284	2284
$Vds3_{plant}$	LE	2267	2267

B. Evaluation

We conduct experiments to evaluate the performance of the proposed method compared to other methods. Six models are implemented consisting of: *Word2Vec*[3], *fastText*[4] , *GloVe*[5] ,*DWN*,*VDWN* and our model (*VLER*). To train *Word2Vec*, *fastText*, *GloVe* models in Vietnamese, we used 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. The models in our experiments are trained with 300 dimensions, the learning rate α is initialized to 0.025. Data for training *DWN*, *VDWN* and *VLER* models has 2,985,618 triples (about 76 million words) and 138,062 individual LE pairs which are extracted from the above corpus and Vietnamese WordNet, words of this corpus which appear less than 10 times are removed[6].

Recently, a number of studies use support vector machine (SVM) (Cortes and Vapnik, 1995) for relation detection especially for LE recognition (Levy et al., 2015), (Tan et al., 2018). In this work, SVM is also used to identify pair of words represented by embeddings vectors are LE relation or not. Linear SVM is used because of its speed and simplicity. We used the *ScikitLearn*[7] implementations with default settings. We create unique feature vector for the SVM's input from

[2]https://github.com/BuiTan/VLER/tree/master/data
[3]http://code.google.com/p/word2vec/
[4]https://github.com/facebookresearch/fastText
[5]https://nlp.stanford.edu/projects/glove/
[6]https://github.com/BuiTan/VLER/tree/master/triples_corpus
[7]http://scikit-learn.org

two distributional vectors of words. Inspired by the experiments of (Weeds et al., 2014), several combinations of vectors are experimented and reported (Table III).

Table III
SEVERAL COMBINATIONS OF VECTORS

V_{DIFF}	the vector difference $(v_{hype} - v_{hypo})$
V_{MULT}	the pointwise product vector $(v_{hype} * v_{hypo})$
V_{ADD}	the vector sum $(v_{hype} + v_{hypo})$
V_{CAT}	the vector concatenation $(v_{hype} \oplus v_{hypo})$
V_{CATs}	the concatenation vector of sum and difference vector $(< v_{hype} + v_{hypo} > \oplus < v_{hype} - v_{hypo} >)$

To combine lexical structure feature and word embedding vectors in a unique vector, we used the concatenation operator ($\oplus$), the last feature vector is defined as:

$$V = V_{lsf} \oplus V_{embeddings} \quad (12)$$

We conducted the experiments on three above datasets, the experimental data consist of pairs are labeled as positive or negative. 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.

Experiment 1.(Vds1 dataset) the data includes 976 LE pairs (positive labels), and 1,026 pairs which are not LE (negative labels). The results shown in Table IV are the accuracy of methods when using different combinations of vectors.

Table IV
LE RECOGNITION RESULTS FOR THE VDS1 DATASET.

Model	$DIFF$	$MULT$	ADD	CAT	V_{CATs}
W2V	0.82	0.77	0.81	0.80	0.79
fastText	0.83	0.78	0.80	0.82	0.83
GloVe	0.80	0.75	0.79	0.77	0.81
DWN	0.81	0.79	0.82	0.82	0.84
VDWN	0.86	0.83	0.84	0.87	0.89
VLER	0.89	0.85	0.87	0.89	0.94

Experiment 2 (Vds2 dataset). The data includes 1,657 LE pairs (positive labels), and 1,657 pairs which are not LE (negative labels). The results shown in Table V are the performance of methods that are measured in terms of precision, recall and F1. Experiment 3 (Vds3 dataset). 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, V_{CATs} is used for combinations of vectors. Experimental results are presented in Table VII.

The result of this experiment shows that the proposed method has recognized exactly the relationship of pairs which contain long concepts, these long concepts have many of the components (Table VI). The vector of long concept doesn't have meaningful because of them is fewer appearances than sort concepts in the corpus. In these cases, the lexical structure information is a useful feature that supplements to realize the lexical entailment relation.

Table V
LE RECOGNITION RESULTS FOR THE $Vds2$ DATASET

Model	**Precision**	**Recall**	**F1**
Word2vec	0.85	0.87	0.86
fastText	0.86	0.88	0.87
GloVe	0.82	0.84	0.83
DWN	0.88	0.88	0.88
VDWN	0.90	0.94	0.92
VLER	0.93	0.96	0.93

Table VI
SOME PAIRS OF LONG CONCEPTS ARE EXACTLY RECOGNIZED THE LEXICAL ENTAILMENT RELATION

Hyernym	**Hyponym**
thực_vật_hạt_kín	*cây_dền_hoang*
thực_vật_họ_loa_kèn	*thực_vật_chi_hành*
ngành_hạt_trần	*cây_vân_sam*
cây_họ_huệ_tây	*hoa_loa_kèn*
động_vật_chân_khớp	*động_vật_thuộc_lớp_nhện*
động_vật_có_nhau_thai	*thú_có_móng_guốc*
động_vật_chân_đầu	*động_vật_giáp_xác_mười_chân*
động_vật_có_vú_và_nhau_thai	*động_vật_linh_trưởng*

Table VII
LE RECOGNITION RESULTS FOR THE $Vds3$ DATASET

Model	**Training**	**Testing**	**Precision**	**Recall**	**F1**
Word2Vec	$animal$	$plant$	0.50	0.60	0.55
fastText			0.51	0.63	0.57
GloVe			0.50	0.58	0.54
DWN			0.52	0.64	0.57
VDWN			0.61	0.76	0.68
VLER			0.67	0.80	0.73
Word2Vec	$plant$	$animal$	0.58	0.72	0.64
fastText			0.59	0.74	0.66
GloVe			0.52	0.69	0.59
DWN			0.57	0.73	0.64
VDWN			0.62	0.78	0.69
VLER			0.66	0.83	0.74

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 V and VII, the proposed method produced better results than the original one, not only in term of the precision but also the recall. Observing the results achieved by six methods we realize that prediction values are often wrong on the pairs have compound words. Normally, compound words have more component is less appearance than another or exist in the corpus, that goes along vectors of these words is less meaningful. In this case, the lexical structure feature

is a useful supplement for supervised learning algorithm to recognize relations exactly. Therefore, the proposed method outperforming than others herein.

V. CONCLUSION

This paper proposed the VLER method for the LE recognition problem which based on the combination of specializing word vectors and lexical structure feature. A number of LE recognition methods based on word embedding and supervised learning have been experimenting for Vietnamese. Experimental results demonstrated that our method achieves the best results, thereby confirm that the lexicon structure features are useful for this problem. We intend to apply our method to detect other kinds of semantic relations also other languages.

REFERENCES

Or Biran and Kathleen McKeown. Classifying taxonomic relations between pairs of wikipedia articles. In *Sixth International Joint Conference on Natural Language Processing, pages = 788–794, year = 2013*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 2017.

Danqi Chen and Christopher Manning. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Mach. Learn.*, pages 273–297, 1995.

Ido Dagan, Dan Roth, Mark Sammons, and Fabio Massimo Zanzotto. *Recognizing Textual Entailment: Models and Applications*. Synthesis Lectures on Human Language Technologies. 2013.

Maayan Geffet and Ido Dagan. The distributional inclusion hypotheses and lexical entailment. In *ACL 2005, Proceedings of the Conference, 25-30 June 2005*, 2005a.

Maayan Geffet and Ido Dagan. The distributional inclusion hypotheses and lexical entailment. In *ACL*, 2005b.

William L. Hamilton, Jure Leskovec, and Dan Jurafsky. Diachronic word embeddings reveal statistical laws of semantic change. *CoRR*, 2016.

Michael Lesk. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In *Proceedings of SIGDOC*, 1986.

Omer Levy and Yoav Goldberg. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2014.

Omer Levy, Steffen Remus, Chris Biemann, and Ido Dagan. Do supervised distributional methods really learn lexical inference relations? In *HLT-NAACL*, 2015.

Anh Tuan Luu, Jung-jae Kim, and See-Kiong Ng. Incorporating trustiness and collective synonym/contrastive evidence into taxonomy construction. In *Proceedings of the 2015 Conference EMNLP*, 2015.

Anh Tuan Luu, Yi Tay, Siu Cheung Hui, and See-Kiong Ng. Learning term embeddings for taxonomic relation identification using dynamic weighting neural network. In *EMNLP*, pages 403–413, 2016.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Michael Mohler, David Bracewell, David Hinote, and Marc Tomlinson. Semantic signatures for example-based linguistic metaphor detection, 2013.

N. Nayak. Learning hypernymy over word embeddings. *arXiv*, 2015.

Kim Anh Nguyen, Maximilian Köper, Sabine Schulte im Walde, and Ngoc Thang Vu. Hierarchical embeddings for hypernymy detection and directionality. *CoRR*, 2017.

Phuong-Thai Nguyen, Van-Lam Pham, Hoang-Anh Nguyen, Huy-Hien Vu, Ngoc-Anh Tran, and Thi-Thu Ha Truong. A two-phase approach for building vietnamese wordnet. *the 8th Global WordNet Conference*, pages 259 – 264, 2016.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.

Enrico Santus, Alessandro Lenci, Qin Lu, and Sabine Schulte im Walde. Chasing hypernyms in vector spaces with entropy. In *EACL*, pages 38–42, 2014.

Vered Shwartz, Enrico Santus, and Dominik Schlechtweg. Hypernyms under siege: Linguistically-motivated artillery for hypernymy detection. *CoRR*, abs/1612.04460, 2016.

R. Snow, D. Jurafsky, and A.Y. Ng. Learning syntactic patterns for automatic hypernym discovery. *Advances in Neural Information Processing Systems*, 2005.

Bui Van Tan, Nguyen Phuong Thai, and Pham Van Lam. Construction of a word similarity dataset and evaluation of word similarity techniques for vietnamese. In *9th International Conference on Knowledge and Systems Engineering, KSE 2017, Hue, Vietnam, October 19-21, 2017*, pages 65–70, 2017. doi: 10.1109/KSE.2017.8119436.

Bui Van Tan, Nguyen Phuong Thai, and Pham Van Lam. Hypernymy detection for vietnamese using dynamic weighting neural network. *In Proceedings of the 19th International Conference on Computational Linguistics and Intelligent Text Processing*, 2018.

Peter D. Turney. Domain and function: A dual-space model of semantic relations and compositions. *J. Artif. Intell. Res.*, 2012.

Peter D. Turney and Saif M. Mohammad. Experiments with three approaches to recognizing lexical entailment. *Natural Language Engineering*, 2015.

Paola Velardi, Stefano Faralli, and Roberto Navigli. Ontolearn reloaded: A graph-based algorithm for taxonomy induction. *Computational Linguistics*, 2013.

Ivan Vulic and Nikola Mrksic. Specialising word vectors for lexical entailment. *CoRR*, 2017.

Julie Weeds, Daoud Clarke, Jeremy Reffin, David J. Weir, and Bill Keller. Learning to distinguish hypernyms and co-hyponyms. In *COLING 2014, 25th International Conference on Computational Linguistics, 2014, Dublin, Ireland*, 2014.

Zheng Yu, Haixun Wang, Xuemin Lin, and Min Wang. Learning term embeddings for hypernymy identification. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, 2015*, 2015.

Will Y. Zou, Richard Socher, Daniel M. Cer, and Christopher D. Manning. Bilingual word embeddings for phrase-based machine translation. In *EMNLP*, 2013.