VNU Journal of Science: Comp. Science & Com. Eng., Vol. 33, No. 2 (2017) 39-58

Vietnamese Semantic Role Labelling

Le-Hong Phuong[1,*], Pham Thai Hoang[2], Pham Xuan Khoai[2],
Nguyen Thi Minh Huyen[1], Nguyen Thi Luong[3], Nguyen Minh Hiep[3]

[1]*VNU University of Science, 334 Nguyen Trai, Thanh Xuan, Hanoi, Vietnam*
[2]*FPT University, Hanoi, Vietnam*
[3]*Dalat University, Lamdong, Vietnam*

Abstract

In this paper, we study semantic role labelling (SRL), a subtask of semantic parsing of natural language sentences and its application for the Vietnamese language. We present our effort in building Vietnamese PropBank, the first Vietnamese SRL corpus and a software system for labelling semantic roles of Vietnamese texts. In particular, we present a novel constituent extraction algorithm in the argument candidate identification step which is more suitable and more accurate than the common node-mapping method. In the machine learning part, our system integrates distributed word features produced by two recent unsupervised learning models in two learned statistical classifiers and makes use of integer linear programming inference procedure to improve the accuracy. The system is evaluated in a series of experiments and achieves a good result, an F_1 score of 74.77%. Our system, including corpus and software, is available as an open source project for free research and we believe that it is a good baseline for the development of future Vietnamese SRL systems.

Received 27 June 2017; Revised 23 October 2017; Accepted 20 November 2017

Keywords: Distributed word representation, integer linear programming, semantic role labelling, Vietnamese, Vietnamese PropBank.

1. Introduction

In this paper, we study semantic role labelling (SRL), a subtask of semantic parsing of natural language sentences. SRL is the task of identifying semantic roles of arguments of each predicate in a sentence. In particular, it answers a question *Who did what to whom, when, where, why?*. For each predicate in a sentence, the goal is to identify all constituents that fill a semantic role, and to determine their roles, such as agent, patient, or instrument, and their adjuncts, such as locative, temporal or manner.

Figure 1 shows the SRL of a simple Vietnamese sentence. In this example, the arguments of the predicate *giúp* (helped) are labelled with their semantic roles. The meaning of the labels will be described in detail in Section 2.2.

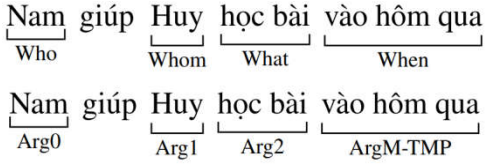


Figure 1. SRL of the Vietnamese sentence. "*Nam giúp Huy học bài vào hôm qua*" (Nam helped Huy to do homework yesterday).

* Corresponding author. Email: phuonglh@vnu.edu.vn
https://doi.org/10.25073/2588-1086/vnucsce.166

SRL has been used in many natural language processing (NLP) applications such as question answering [1], machine translation [2], document summarization [3] and information extraction [4]. Therefore, SRL is an important task in NLP. The first SRL system was developed by Gildea and Jurafsky [5]. This system was performed on the English FrameNet corpus. Since then, SRL task has been widely studied by the NLP community. In particular, there have been two shared-tasks, CoNLL-2004 [6] and CoNLL-2005 [7], focusing on SRL task for English. Most of the systems participating in these shared-tasks treated this problem as a classification problem which can be solved by supervised machine learning techniques. There exists also several systems for other well-studied languages like Chinese [8] or Japanese [9].

This paper covers not only the contents of two works published in conference proceedings [10] (in Vietnamese) and [11] on the construction and the evaluation of a first SRL system for Vietnamese, but also an extended investigation of techniques used in SRL. More concretely, the use of integer linear programming inference procedure and distributed word representations in our semantic role labelling system, which leads to improved results over our previous work, as well as a more elaborate evaluation are new for this article.

Our system includes two main components, a SRL corpus and a SRL software which is thoroughly evaluated. We employ the same development methodology of the English PropBank to build a SRL corpus for Vietnamese containing a large number of syntactically parsed sentences with predicate-argument structures.

We then use this SRL corpus and supervised machine learning models to develop a SRL software for Vietnamese. We demonstrate that a simple application of SRL techniques developed for English or other languages could not give a good accuracy for Vietnamese. In particular, in the constituent identification step, the widely used 1-1 node-mapping algorithm for extracting argument candidates performs poorly on the Vietnamese dataset, having F_1 score of 35.93%. We thus introduce a new algorithm for extracting candidates, which is much more accurate, achieving an F_1 score of 84.08%. In the classification step, in addition to the common linguistic features, we propose novel and useful features for use in SRL, including function tags and distributed word representations. These features are employed in two statistical classification models, maximum entropy and support vector machines, which are proved to be good at many classification problems. In order to incorporate important grammatical constraints into the system to improve further the performance, we combine machine learning techniques with an inference procedure based on integer linear programming. Finally, we use distributed word representations produced by two recent unsupervised models, the Skip-gram model and the GloVe model, on a large corpus to alleviate the data sparseness problem. These word embeddings help our SRL software system generalize well on unseen words. Our final system achieves an F_1 score of 74.77% on a test corpus. This system, including corpus and software, is available as an open source project for free research and we believe that it is a good baseline for the development of future Vietnamese SRL systems.

The remainder of this paper is structured as follows. Section 2 describes the construction of a SRL corpus for Vietnamese. Section 3 presents the development of a SRL software, including the methodologies of existing systems and of our system. Section 4 presents the evaluation results and discussion. Finally, Section 5 concludes the paper and suggests some directions for future work.

2. Vietnamese SRL corpus

Like many other problems in NLP, annotated corpora are essential for statistical learning as well as evaluation of SRL systems. In this section, we start with an introduction of existing English SRL corpora. Then we present our work on the construction of the first reference SRL corpus for Vietnamese.

2.1. Existing English SRL corpora

2.1.1. FrameNet

The FrameNet project is a lexical database of English. It was built by annotating examples of how words are used in actual texts. It consists of more than 10,000 word senses, most of them with annotated examples that show the meaning and usage and more than 170,000 manually annotated sentences [12]. This is the most widely used dataset upon which SRL systems for English have been developed and tested.

FrameNet is based on the Frame Semantics theory [13]. The basic idea is that the meanings of most words can be best understood on the basis of a semantic frame: a description of a type of event, relation, or entity and the participants in it. All members in semantic frames are called frame elements. For example, a sentence in FrameNet is annotated in cooking concept as shown in Figure 2.

Figure 2. example sentence in the FrameNet corpus.

2.1.2. PropBank

PropBank is a corpus that is annotated with verbal propositions and their arguments [14]. PropBank tries to supply a general purpose labelling of semantic roles for a large corpus to support the training of automatic semantic role labelling systems. However, defining such a universal set of semantic roles for all types of predicates is a difficult task; therefore, only Arg0 and Arg1 semantic roles can be generalized. In addition to the core roles, PropBank defines several adjunct roles that can apply to any verb. It is called Argument Modifier. The semantic roles covered by the PropBank are the following:

• Core Arguments (Arg0-Arg5, ArgA): Arguments define predicate specific roles. Their semantics depend on predicates in the sentence.

• Adjunct Arguments (ArgM-*): General arguments that can belong to any predicate. There are 13 types of adjuncts.

• Reference Arguments (R-*): Arguments represent arguments realized in other parts of the sentence.

• Predicate (V): Participant realizing the verb of the proposition.

For example, the sentence of Figure 2 can be annotated in the PropBank role schema as shown in Figure 3.

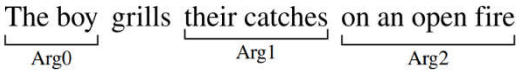


Figure 3. An example sentence in the PropBank corpus.

The English PropBank methodology is currently implemented for a wide variety of languages such as Chinese, Arabic or Hindi with the aim of creating parallel PropBanks[1]. This SRL resource has a great impact on many natural language processing tasks and applications.

2.1.3. VerbNet

VerbNet is a verb lexicon of English, which was developed by Karin Kipper-Schuler and colleagues [15]. It contains more than 5800 English verbs, which are classified into 270 groups, according to the verb classification method of Beth Levin [16]. In this approach, the behavior of a verb is mostly determined by its meaning.

Once classified into groups, each verb group is added semantic roles. VerbNet has 23 semantic roles, for example

• Actor, the participant that is the investigator of an event.

• Agent, the actor in an event who initiates and carries out the event and who exists independently of the event.

• Attribute, the undergoer that is a property of an entity or entities.

• Destination, the goal that is a concrete, physical location.

These syntactic roles normally answer who, what, when and how questions. A SRL annotation guidelines of this project is available online[2]. In summary, SRL corpora have been constructed for English and other well-resourced languages. They are important

[1] http://verbs.colorado.edu/ mpalmer/projects/ace/ EPB-annotation-guidelines.pdf

[2]http://verbs.colorado.edu/verb-index/VerbNet_Guidelines.pdf

resources which are very useful for many natural language processing applications.

For the Vietnamese language, there has not existed any SRL corpus which with a similar level like those of English corpora described above. In the following sections, we report our initiatives for constructing and evaluating a SRL corpus for Vietnamese.

2.2. Building a Vietnamese propBank

In this section, we present the construction of a Vietnamese SRL corpus, which is referred as Vietnamese PropBank hereafter. We first describe annotation guidelines and then describe the SRL corpus which has been developed.

2.2.1. Vietnamese SRL Annotation Guidelines

The determination of semantic roles in the Vietnamese language is a difficult problem and it has been investigated with different opinions. In general, Vietnamese linguists have not reached a consensus on a list of semantic roles for the language. Different linguists proposed different lists; some used the same name but with different meaning of a role, or different names having the same meaning.

Nevertheless, one can use an important principle for determining semantic roles: "Semantic role is the actual role a participant plays in some situation and it always depends on the nature of that situation" [17]. This means that when identifying the meaning of a phrase or of a sentence, one must not separate it out of the underlying situation that it appears. While there might be some controversy about the exact semantic role names should be, one can list common semantic roles which have been accepted by most of Vietnamese linguists [18].

The syntactic sub-categorization frames are closely related to the verb meanings. That is, the meaning of a sentence can be captured by the subcategorization frame of the verb predicate. In consequence, the sentence meaning can be described by labelling the semantic roles for each participant in the sub-categorization frame of the predicate. This approach is adopted by many Vietnamese linguists and different semantic roles set have been proposed. For example, *Cao Xuân Hạo* [17] makes use of the argument (obligatory participants) roles as agent, actor, processed, force, carrier, patient, experiencer, goal, etc., while *Diệp Quang Ban* [19] makes use of fact categories: dynamic, static, mental, existential, verbal, relational etc. For adjuncts (optional participants), *Cao Xuân Hạo* uses the roles: manner, mean, result, path, etc., while *Diệp Quang Ban* makes use of circumstance types: time, space, cause, condition, goal, result, path, etc.

In this work, we took a pragmatic standpoint during the design of a semantic role tagset and focused our attention on the SRL categories that we expect to be most necessary and useful in practical applications. We have constructed a semantic role tagset based on two following principles:

• The semantic roles are well-defined and commonly accepted by the Vietnamese linguist community.

• The semantic roles are comparable to those of the English PropBank corpus, which make them helpful and advantageous for constructing multi-lingual corpora and applications in later steps. Furthermore, it seems fairly indisputable that there are structural and semantic correspondences accross languages.

We have selected a SRL tagset which is basically similar to that of the PropBank. However, some roles are made more fine-grained accounting for idiosyncratic properties of the Vietnamese language. In addition, some new roles are added to better distinguish predicate arguments when the predicate is an adjective, a numeral, a noun or a preposition, which is a common phenomenon in Vietnamese besides the popular verbal predicate.

The following paragraph describes some semantic roles of predicative arguments where the predicate is a verb:

• Arg0: The agent semantic role representing a person or thing who is the doer of an event. For example,

Nam đến trường (Nam goes to school.)
Arg0

• Arg0-Identified and Arg1-Identifier: The semantic roles representing identified entity and

identifier respectively, normally used with the copula "là". For example,

Cầu thủ giỏi nhất ở đây là anh ấy
Arg0–Identified Arg1–Identifier

(He is the best player here.)

• Arg1-Patient: The semantic role which is the surface object of a predicate indicating the person or thing affected. For example,

Bộ đội phá cầu
Arg1-Patient

(The soldiers broke a bridge.)

• Arg2: The semantic role of a beneficiary indicating a referent who is advantaged or disadvantaged by an event. For example,

Nó chữa cái xe cho chị ấy
Arg2

(He repaired a bike for her.)

Figure 4 presents an example of the SRL analysis of a syntactically bracketed sentence "*Ba đứa con anh đã có việc làm ổn định*." (His three children have had a permanent job.). The semantic roles of this sentence include:

• Arg0: "*ba đứa con anh*" (his three children) is the agent

• ArgM-TMP: "*đã*" is a temporal modifier

• Rel: "*có*" (have) is the predicate

• Arg1: "*việc làm ổn định*" (a permanent job) is the patient.

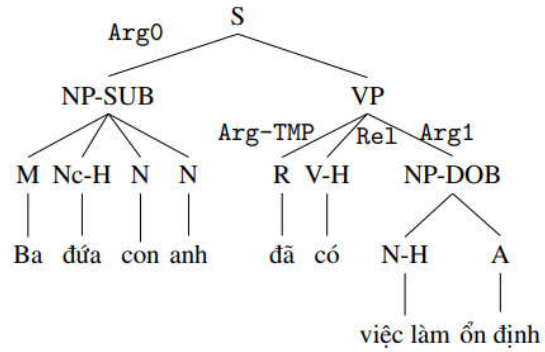


Figure 4. A SRL annotated Vietnamese sentence.

2.2.2. Vietnamese SRL Corpus

Once the SRL annotation guidelines have been designed, we built a Vietnamese SRL corpus by following two main steps.

In the first step, we proposed a set of conversion rules to convert automatically a syntactically annotated treebank containing 10,000 manually annotated sentences (the VietTreeBank) to a coarse-grained SRL annotated corpus.

The Vietnamese treebank is one result of a national project which aims to develop basic resources and tools for Vietnamese language and speech processing[3]. The raw texts of the treebank are collected from the social and political sections of the Youth online daily newspaper. The corpus is divided into three sets corresponding to three annotation levels: word-segmented, part-of-speech-tagged and syntax-annotated set. The syntax-annotated corpus, a subset of the part-of-speech-tagged set, is currently composed of 10,471 sentences (225,085 tokens). Sentences range from 2 to 105 words, with an average length of 21.75 words. There are 9,314 sentences of length 40 words or less. The tagset of the treebank has 38 syntactic labels (18 part-of-speech tags, 17 syntactic category tags, 3 empty categories) and 17 function tags. For details, please refer to [20][4]. The meanings of some common tags are listed in Table 1.

Table 1. Some Vietnamese treebank tags

No.	Category	Description
1.	S	simple declarative clause
2.	VP	verb phrase
3.	NP	noun phrase
4.	PP	preposition phrase
5.	N	common noun
6.	V	verb
7.	P	pronoun
8.	R	adverb
9.	E	preposition
10.	CC	coordinating conjunction

The coarse-grained semantic role tagset contains 24 role names which are all based on the main roles of the PropBank. We carefully investigated the tagset of the VietTreeBank based on detailed guidelines of constituency structures, phrasal types, functional tags, clauses, parts-of-speech and adverbial

[3] VLSP Project, https://vlsp.hpda.vn/demo/

[4] All the resources are available at the website of the VLSP project.

functional tagset to propose a set of rules for determining high-level semantic roles. Some rules for coarse-grained annotation are shown in Table 2. Each rule is used to determine a semantic role for a phrase of a sentence.

As an example, consider the constituency analysis of a sentence in the VietTreeBank "*Kia là những ngôi nhà vách đất.*" (Over there are soil-wall houses.)

(S (NP-SUB (P-H Kia)) (VP (V-H là) (NP (L những) (Nc-H ngôi) (N nhà) (NP (N-H vách) (N đất)))) (. .))

First, using the annotation rule for Arg0, the phrase having syntactical function SUB or preceding the predicate of the sentence, we can annotate the semantic role Arg0 for the word "*Kia*". The predicate "*là*" is annotated with semantic role REL. Finally, the noun phrase following the predicate "*những ngôi nhà vách đất*" is annotated with Arg1.

Table 2. Some rules for coarse-grained SRL annotation

Role	Description	Rule
ARG0	Agent	SUB | Phrasal types (NP, ...) preceding predicate
ARG1	Patient	DOB | phrasal types (NP, ...) following predicate
ARG2	Beneficiary	IOB phrases
ARGM-NEG	Negation	Negative words "*không, chẳng, chớ, chả*"
ARGM-LOC	Locatives	LOC phrases
ARGM-MNR	Manner markers	MNR phrases
ARGM-CAU	Cause clauses	PRP | causal words "*do, bởi vì, vì, bởi*,"
ARGM-DIR	Directionals	DIR phrases
ARGM-DIS	Conjunctive clauses	CC phrases or C word
ARGM-EXT	Extent markers	EXT phrases

In the second step, we developed a software to help a team of Vietnamese linguists manually revise and annotate the converted corpus with fine-grained semantic roles. The software is web-based, friendly and easy for correction and edition of multiple linguists. In addition, it also permits a collaborative work where any edition at sentence level is versionized and logged with meta-information so as to facilitate cross validation and discussion between linguists if necessary.

We have completed the semantic role annotation of 5,460 sentences of the VietTreeBank, covering 7,525 verbal and adjectival predicatives. The annotation guidelines as well as the current SRL corpus are published as open resources for free research. n the next section, we present our effort in developing a SRL software system for Vietnamese which is constructed and evaluated on this SRL corpus.

3. Vietnamese SRL system

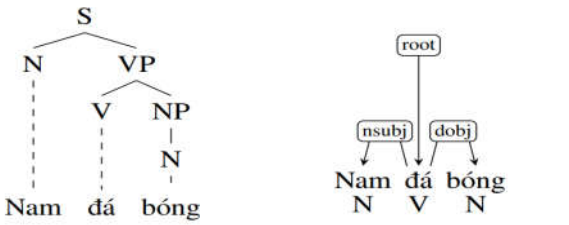
3.1. Existing approaches

This section gives a brief survey of common approaches which are used by many existing SRL systems of well-studied languages. These systems are investigated in two aspects: (a) the data type that the systems use and (b) their approaches for labelling semantic roles, including model types, labelling strategies, degrees of granularity and post-processing.

3.1.1. Data types

The input data of a SRL system are typically syntactically parsed sentences. There are two common syntactic representations namely bracketed trees and dependency trees. Some systems use bracketed trees of sentences as input data. A bracketed tree of a sentence is the tree of nested constituents representing its constituency structure. Some systems use dependency trees of a sentence, which represents dependencies between individual words of a sentence. The syntactic dependency represents the fact that the presence of a word is licensed by another word which is its governor. In a typed dependency analysis, grammatical labels are added to the dependencies to mark their grammatical relations, for example *nominal subject* (nsubj) or *direct object* (dobj).

Figure 5 shows the bracketed tree and the dependency tree of an example sentence.



(a) The bracketed tree (b) The dependency tree

Figure 5. Bracketed and dependency trees for sentence *Nam đá bóng* (Nam plays football).

3.1.2. SRL strategy

Input structures

The first step of a SRL system is to extract constituents that are more likely to be arguments or parts of arguments. This step is called argument candidate extraction. Most of SRL systems for English use 1-1 node mapping method to find candidates. This method searches all nodes in a parse tree and maps constituents and arguments. Many systems use a pruning strategy on bracketed trees to better identify argument candidates [8].

Model types

In a second step, each argument candidate is labelled with a semantic role. Every SRL system has a classification model which can be classified into two types, independent model or joint model. While an independent model decides the label of each argument candidate independently of other candidates, a joint model finds the best overall labelling for all candidates in the sentence at the same time. Independent models are fast but are prone to inconsistencies such as argument overlap, argument repetition or argument missing. For example, Figure 6 shows some examples of these inconsistencies when analyzing the Vietnamese sentence *Do học chăm, Nam đã đạt thành tích cao* (By studying hard, Nam got a high achievement).

Do học chăm, Nam đã *đạt* thành tích cao.
Arg1

Do học chăm, Nam đã *đạt* thành tích cao.
Arg1

(a) Overlapping argument

Do học chăm, Nam đã *đạt* thành tích cao.
Arg1 *Arg1*

(b) Repeated argument

Do học chăm, Nam đã *đạt* thành tích cao .
Arg0 *Arg0*

(c) Missing argument

Figure 6. Examples of some inconsistencies.

Labelling strategies

Strategies for labelling semantic roles are diverse, but they can be classified into three main strategies. Most of the systems use a two-step approach consisting of identification and classification [21, 22]. The first step identifies arguments from many candidates, which is essentially a binary classification problem. The second step classifies the identified arguments into particular semantic roles. Some systems use a single classification step by adding a "null" label into semantic roles, denoting that this is not an argument [23]. Other systems consider SRL as a sequence tagging problem [24, 25].

Granularity

Existing SRL systems use different degrees of granularity when considering constituents. Some systems use individual words as their input and perform sequence tagging to identify arguments. This method is called word-by-word (W-by-W) approach. Other systems use syntactic phrases as input constituents. This method is called constituent-by-constituent (C-by-C) approach. Compared to the W-by-W approach, C-by-C approach has two main advantages. First, phrase boundaries are usually consistent with argument boundaries. Second, C-by-C approach allows us to work with larger contexts due to a smaller number of candidates in comparison to the W-by-W approach. Figure 7 presents an example of C-by-C and W-by-W approaches.

Nam *giúp* Huy học bài vào hôm qua

(a) Example of C-by-C

Nam *giúp* Huy học bài vào hôm qua

(b) Example of W-by-W

Figure 7. C-by-C and W-by-W approaches.

Post-processing

To improve the final result, some systems use post-processing to correct argument labels. Common post-processing methods include re-ranking, Viterbi search and integer linear programming (ILP).

3.2. Our approach

The previous subsection has reviewed existing techniques for SRL which have been published so far for well-studied languages. In this section, we first show that these techniques per se cannot give a good result for Vietnamese SRL, due to some inherent difficulties, both in terms of language characteristics and of the available corpus. We then develop a new algorithm for extracting candidate constituents for use in the identification step.

Some difficulties of Vietnamese SRL are related to its SRL corpus. As presented in the previous section, this SRL corpus has 5,460 annotated sentences, which is much smaller than SRL corpora of other languages. For example, the English PropBank contains about 50,000 sentences, which is about ten times larger. While smaller in size, the Vietnamese PropBank has more semantic roles than the English PropBank has – 28 roles compared to 21 roles. This makes the unavoidable data sparseness problem more severe for Vietnamese SRL than for English SRL.

In addition, our extensive inspection and experiments on the Vietnamese PropBank have uncovered that this corpus has many annotation errors, largely due to encoding problems and inconsistencies in annotation. In many cases, we have to fix these annotation errors by ourselves. In other cases where only a proposition of a complex sentence is incorrectly annotated, we perform an automatic preprocessing procedure to drop it out, leave the correctly annotated propositions untouched. We finally come up with a corpus of 4,800 sentences which are semantic role annotated.

A major difficulty of Vietnamese SRL is due to the nature of the language, where its linguistic characteristics are different from occidental languages [26]. We first try to apply the common node-mapping algorithm which is widely used in English SRL systems to the Vietnamese corpus. However, this application gives us a very poor performance. Therefore, in the identification step, we develop a new algorithm for extracting candidate constituents which is much more accurate for Vietnamese than the node-mapping algorithm. Details of experimental results will be provided in the Section 4.

In order to improve the accuracy of the classification step, and hence of our SRL system as a whole, we have integrated many useful features for use in two statistical classification models, namely Maximum Entropy (ME) and Support Vector Machines (SVM). On the one hand, we adapt the features which have been proved to be good for SRL of English. On the other hand, we propose some novel features, including function tags, predicate type and distance. Moreover, to improve further the performance of our system, we introduce some appropriate constraints and apply a post-processing method by using ILP. Finally, to better handle unseen words, we generalize the system by integrating distributed word representations.

In the next paragraphs, we first present our constituent extraction algorithm to get inputs for the identification step and then the ILP post-processing method. Details of the features used in the classification step and the effect of distributed word representations in SRL will be presented in Section 4.

3.2.1. Constituent extraction algorithm

Our algorithm derives from the pruning algorithm for English [27] with some modifications. While the original algorithm collects sisters of the current node, our algorithm checks the condition whether or not children of each sister have the same phrase label and have different function label from their parent. If they have the same phrase labels and different function labels from their parent, our algorithm collects each of them as an argument candidate. Otherwise, their parent is collected as a candidate. In addition, we remove the constraint that does not collect coordinated nodes from the original algorithm.

This algorithm aims to extract constituents from a bracketed tree which are associated to their corresponding predicates of the sentence.

If the sentence has multiple predicates, multiple constituent sets corresponding to the predicates are extracted. The pseudo code of the algorithm is described in Algorithm 1.

Algorithm 1: Constituent Extraction Algorithm

Data: A bracketed tree T and its predicate
Result: A tree with constituents for the predicate
begin
  $currentNode \leftarrow predicateNode$
  while $currentNode \neq$ T.root() **do**
    for $S \in currentNode$.sibling() **do**
      if $|S.children()| > 1$ and
        $S.children().get(0).isPhrase()$ **then**
          $sameType \leftarrow true$
          $diffTag \leftarrow true$
          $phraseType \leftarrow$
            $S.children().get(0).phraseType()$
          $funcTag \leftarrow$
            $S.children().get(0).functionTag()$
          for $i \leftarrow 1$ **to** $|S.children()| - 1$ **do**
            if
              $S.children().get(i).phraseType() \neq$
                $phraseType$ **then**
                  $sameType \leftarrow false$
                  break
            if
              $S.children().get(i).functionTag() =$
                $funcTag$ **then**
                  $diffTag \leftarrow false$
                  break
          if $sameType$ and $diffTag$ **then**
            for $child \in S.children()$ **do**
              $T.collect(child)$
        else
          $T.collect(S)$
    $currentNode \leftarrow currentNode.parent()$
  return T

This algorithm uses several simple functions. The *root*() function gets the root of a tree. The *children*() function gets the children of a node. The *sibling*() function gets the sisters of a node. The *isPhrase*() function checks whether a node is of phrasal type or not. The *phraseType*() function and *functionTag*() function extracts the phrase type and function tag of a node, respectively. Finally, the *collect*(*node*) function collects words from leaves of the subtree rooted at a node and creates a constituent.

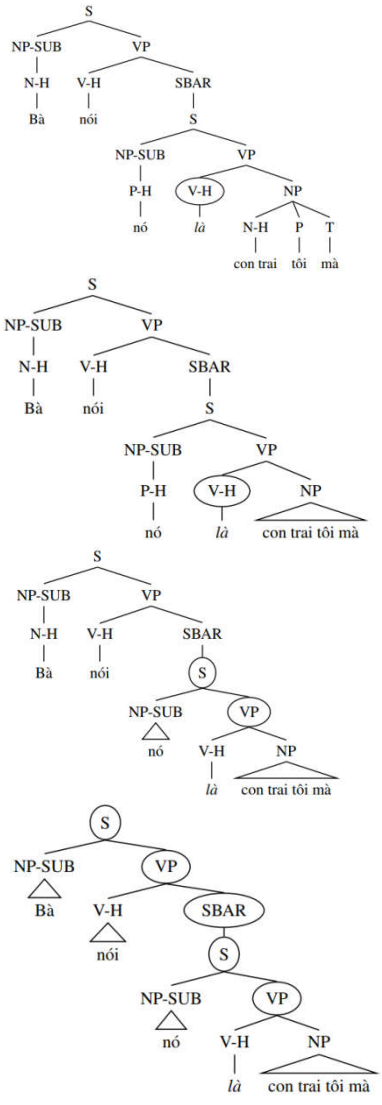


Figure 8. Extracting constituents of the sentence "Bà nói nó là con trai tôi mà" at predicate "*là*".

Figure 8 shows an example of running the algorithm on a sentence *Bà nói nó là con trai tôi mà* (You said that he is my son). First, we

find the current predicate node V-H *là* (is). The current node has only one sibling NP node. This NP node has three children where some of them have different labels from their parents, so this node and its associated words are collected. After that, we set current node to its parent and repeat the process until reaching the root of the tree. Finally, we obtain a tree with the following constituents for predicate *là*: *Bà*, *nói*, *nó*, and *con trai tôi mà*.

3.2.2. Integer linear programming

Because the system classifies arguments independently, labels assigned to arguments in a sentence may violate Vietnamese grammatical constraints. To prevent such violation and improve the result, we propose a post-processing process which finds the best global assignment that also satisfies grammatical constraints. Our work is based on the ILP method of English PropBank [28]. Some constraints that are unique to Vietnamese are also introduced and incorporated.

Integer programs are almost identical to linear programs. The cost function and the constraints are all in linear form. The only difference is that the variables in ILP can only take integer values. A general binary ILP can be stated as follows.

Given a cost vector $\vec{p} \in \mathsf{R}^d$, a set of variables $\vec{z} = (z_1, \ldots, z_d) \in \mathsf{R}^d$, and cost matrices $\mathbf{C}_1 \in \mathsf{R}^{t_1} \times \mathsf{R}^d$, $\mathbf{C}_2 \in \mathsf{R}^{t_2} \times \mathsf{R}^d$, where t_1, t_2 are the number of inequality and equality constraints and d is the number of binary variables. The ILP solution $\hat{\vec{z}}$ is the vector that maximizes the cost function:

$$\hat{\vec{z}} = \underset{\vec{z} \in 0,1^d}{argmax}\ \vec{p} \cdot \vec{z} \quad \text{subject to} \begin{cases} \mathbf{C}_1 \vec{z} \geq \vec{b}_1 \\ \mathbf{C}_2 \vec{z} = \vec{b}_2 \end{cases} \qquad (1)$$

where $\vec{b}_1, \vec{b}_2 \in \mathsf{R}^d$.

Our system attempts to find exact roles for argument candidate set for each sentence. This set is denoted as $S^{1:M}$, where the index ranged from 1 to M; and the argument role set is denoted as P. Assuming that the classifier returns a score, $score(S^i = c^i)$, corresponding to the likelihood of assigning label c^i to argument S^i. The aim of the system is to find the maximal overall score of the arguments:

$$\hat{c}^{1:M} = \underset{c^{1:M} \in \mathsf{P}^M}{argmax}\ score(S^{1:M} = c^{1:M}) \qquad (2)$$

$$= \underset{c^{1:M} \in \mathsf{P}^M}{argmax} \sum_{i=1}^{M} score(S^i = c^i) \qquad (3)$$

ILP Constraints

In this paragraph, we propose a constraint set for our SRL system. Some of them are directly inspired and derived from results for English SRL, others are constraints that we specify uniquely to account for Vietnamese specificities. The constraint set includes:

1. One argument can take only one type.
2. Arguments cannot overlap with the predicate in the sentence.
3. Arguments cannot overlap other arguments in the sentence.
4. There is no duplicating argument phenomenon for core arguments in the sentence.
5. If the predicate is not verb type, there are only 2 types of core argument Arg0 and Arg1.

In particular, constraints from 1 to 4 are derived from the ILP method for English [28], while constraint 5 is designed specifically for Vietnamese.

ILP Formulation

To find the best overall labelling satisfying these constraints, we transform our system to an ILP problem. First, let $z_{ic} = [S^i = c]$ be the binary variable that shows whether or not S^i is labelled argument type c. We denote $p_{ic} = score(S^i = c)$. The objective function of the optimization problem can be written as:

$$\underset{z \in 0,1}{argmax} \sum_{i=1}^{M} \sum_{c=1}^{|\mathsf{P}|} p_{ic} z_{ic}. \qquad (4)$$

Next, each constraint proposed above can be reformulated as follows:

1. One argument can take only one type.

$$\sum_{c=1}^{|\mathsf{P}|} z_{ic} = 1, \quad \forall i \in [1, M]. \quad (5)$$

2. Arguments cannot overlap with the predicate in the sentence.
3. Arguments cannot overlap other arguments in the sentence. If there are k

arguments $S^1, S^2, ..., S^k$ that appear in a same word in the sentence, we can conclude that there are at least $k-1$ arguments that are classified as "null":

$$\sum_{i=1}^{k} z_{ic} \geq k-1 \quad (c = ``\text{null}''). \qquad (6)$$

This constraint has been satisfied by our constituent extraction approach. Thus, we do not need to add this constraint in the post-processing step if the constituent extraction algorithm has been used.

4. There is no duplicating argument phenomenon for core arguments in the sentence.

$$\sum_{i=1}^{M} z_{ic} \leq 1, \quad \forall c \in \{\text{Arg0}, \text{Arg1}, \text{Arg2}, \text{Arg3}, \text{Arg4}\}. \qquad (7)$$

5. If the predicate is not verb type, there are only 2 types of core argument Arg0 and Arg1

$$\sum_{i=1}^{M} z_{ic} = 0 \quad \forall c \in \{\text{Arg2}, \text{Arg3}, \text{Arg4}\}. \qquad (8)$$

In the next section, we present experimental results, system evaluation and discussions.

4. Evaluation

In this section, we describe the evaluation of our SRL system. First, we first introduce two feature sets used in machine learning classifiers. Then, the evaluation results are presented and discussed. Next, we report the improved results by using integer linear programming inference method. Finally, we present the efficacy of distributed word representations in generalizing the system to unseen words.

4.1. Feature sets

We use two feature sets in this study. The first one is composed of basic features which are commonly used in SRL system for English. This feature set is used in the SRL system of Gildea and Jurafsky [5] on the FrameNet corpus.

4.1.1. Basic features

This feature set consists of 6 feature templates, as follows:

1. Phrase type: This is very useful feature in classifying semantic roles because different roles tend to have different syntactic categories. For example, in the sentence in Figure 8 *Bà nói nó là con trai tôi mà*, the phrase type of constituent *nó* is *NP*.

2. Parse tree path: This feature captures the syntactic relation between a constituent and a predicate in a bracketed tree. This is the shortest path from a constituent node to a predicate node in the tree. We use either symbol ↑ or symbol ↓ to indicate the upward direction or the downward direction, respectively. For example, the parse tree path from constituent *nó* to the predicate *là* is *NP*↑ *S*↓ *VP*↓ *V*.

3. Position: Position is a binary feature that describes whether the constituent occurs after or before the predicate. It takes value *0* if the constituent appears before the predicate in the sentence or value *1* otherwise. For example, the position of constituent *nó* in Figure 8 is *0* since it appears before predicate *là*.

4. Voice: Sometimes, the differentiation between active and passive voice is useful. For example, in an active sentence, the subject is usually an *Arg0* while in a passive sentence, it is often an *Arg1*. Voice feature is also binary feature, taking value *1* for active voice or *0* for passive voice. The sentence in Figure 8 is of active voice, thus its voice feature value is *1*.

5. Head word: This is the first word of a phrase. For example, the head word for the phrase *con trai tôi mà* is *con trai*.

6. Subcategorization: Subcategorization feature captures the tree that has the concerned predicate as its child. For example, in Figure 8, the subcategorization of the predicate *là* is *VP(V, NP)*.

4.1.2. New features

Preliminary investigations on the basic feature set give us a rather poor result. Therefore, we propose some novel features so as to improve the accuracy of the system. These features are as follows:

1. Function tag: Function tag is a useful information, especially for classifying adjunct arguments. It determines a constituent's role, for example, the function tag of constituent *nó* is *SUB*, indicating that this has a subjective role.

2. Distance: This feature records the length of the full parse tree path before pruning. For example, the distance from constituent *nó* to the predicate *là* is *3*.

3. Predicate type: Unlike in English, the type of predicates in Vietnamese is much more complicated. It is not only a verb, but is also a noun, an adjective, or a preposition. Therefore, we propose a new feature which captures predicate types. For example, the predicate type of the concerned predicate is *V*.

4.2. Results and discussions

4.2.1. Evaluation Method

We use a 10-fold cross-validation method to evaluate our system. The final accuracy scores is the average scores of the 10 runs.

The evaluation metrics are the precision, recall and F_1-measure. The precision (P) is the proportion of labelled arguments identified by the system which are correct; the recall (R) is the proportion of labelled arguments in the gold results which are correctly identified by the system; and the F_1-measure is the harmonic mean of P and R, that is $F_1 = 2PR/(P+R)$.

4.2.2. Baseline system

In the first experiment, we compare our constituent extraction algorithm to the 1-1 node mapping and the pruning algorithm [28]. Table 3 shows the performance of two extraction algorithms.

Table 3. Accuracy of three extraction algorithms

	1-1 Node Mapping Alg.	Pruning Alg.	Our Extraction Alg.
Precision	29.58%	85.05%	82.15%
Recall	45.82%	79.39%	86.12%
F_1	35.93%	82.12%	**84.08%**

We see that our extraction algorithm outperforms significantly the 1-1 node mapping algorithm, in both of the precision and the recall ratios. It is also better than the pruning algorithm. In particular, the precision of the 1-1 node mapping algorithm is only 29.58%; it means that this method captures many candidates which are not arguments. In contrast, our algorithm is able to identify a large number of correct argument candidates, particularly with the recall ratio of 86.12% compared to 79.39% of the pruning algorithm. This result also shows that we cannot take for granted that a good algorithm for English could also work well for another language of different characteristics.

In the second experiment, we continue to compare the performance of the two extraction algorithms, this time at the final classification step and get the baseline for Vietnamese SRL. The classifier we use in this experiment is a Support Vector Machine (SVM) classifier[5]. Table 4 shows the accuracy of the baseline system.

Table 4. Accuracy of the baseline system

	1-1 Node Mapping Alg.	Pruning Alg.	Our Extraction Alg.
Precision	66.19%	73.63%	73.02%
Recall	29.34%	62.79%	67.16%
F_1	40.66%	67.78%	**69.96%**

Once again, this result confirms that our algorithm achieves the better result. The F_1 of our baseline SRL system is 69.96%, compared to 40.66% of the 1-1 node mapping and 67.78% of the pruning system. This result can be explained by the fact that the 1-1 node mapping and the pruning algorithm have a low recall ratio, because it identifies incorrectly many argument candidates.

4.2.3. Labelling strategy

In the third experiment, we compare two labelling strategies for Vietnamese SRL. In addition to the SVM classifier, we also try the Maximum Entropy (ME) classifier, which usually gives good accuracy in a wide variety of classification problems[6]. Table 5 shows the F_1 scores of different labelling strategies.

[5] We use the linear SVM classifier with L_2 regularization provided by the scikit-learn software package. The regularization term is fixed at 0.1.

[6] We use the logistic regression classifier with L_2 regularization provided by the scikit-learn software package. The regularization term is fixed at 1.

Table 5. Accuracy of two labelling strategies

	ME	SVM
1-step strategy	69.79%	69.96%
2-step strategy	69.28%	69.38%

We see that the performance of SVM classifier is slightly better than the performance of ME classifier. The best accuracy is obtained by using 1-step strategy with SVM classifier. The current SRL system achieves an F_1 score of 69.96%.

4.2.4. Feature analysis

In the fourth experiment, we analyse and evaluate the impact of each individual feature to the accuracy of our system so as to find the best feature set for our Vietnamese SRL system. We start with the basic feature set presented previously, denoted by Φ_0 and augment it with modified and new features as shown in Table 6. The accuracy of these feature sets are shown in Table 7.

Table 6. Feature sets

Feature Set	Description
Φ_1	$\Phi_0 \cup$ {Function Tag}
Φ_2	$\Phi_0 \cup$ {Predicate Type}
Φ_3	$\Phi_0 \cup$ {Distance}

Table 7. Accuracy of feature sets in Table 6

Feature Set	Precision	Recall	F_1
Φ_0	73.02%	67.16%	69.96%
Φ_1	**77.38%**	**71.20%**	**74.16%**
Φ_2	72.98%	67.15%	69.94%
Φ_3	**73.04%**	**67.21%**	**70.00%**

We notice that amongst the three features, function tag is the most important feature which increases the accuracy of the baseline feature set by about 4% of F_1 score. The distance feature also helps increase slightly the accuracy. We thus consider the fourth feature set Φ_4 defined as

$$\Phi_4 = \Phi_0 \cup \{\text{FunctionTag}\} \cup \{\text{Distance}\}.$$

In the fifth experiment, we investigate the significance of individual features to the system by removing them, one by one from the feature set Φ_4. By doing this, we can evaluate the importance of each feature to our overall system. The feature sets and their corresponding accuracy are presented in Table 8 and Table 9 respectively.

Table 8. Feature sets (continued)

Feature Set	Description
Φ_5	$\Phi_4 \setminus$ {Function Tag}
Φ_6	$\Phi_4 \setminus$ {Distance}
Φ_7	$\Phi_4 \setminus$ {Head Word}
Φ_8	$\Phi_4 \setminus$ {Path}
Φ_9	$\Phi_4 \setminus$ {Position}
Φ_{10}	$\Phi_4 \setminus$ {Voice}
Φ_{11}	$\Phi_4 \setminus$ {Subcategorization}
Φ_{12}	$\Phi_4 \setminus$ {Predicate}
Φ_{13}	$\Phi_4 \setminus$ {Phrase Type}

Table 9. Accuracy of feature sets in Table 8

Feature Set	Precision	Recall	F_1
Φ_4	77.53%	71.29%	74.27%
Φ_5	73.04%	67.21%	70.00%
Φ_6	77.38%	71.20%	74.16%
Φ_7	73.74%	67.17%	70.29%
Φ_8	77.58%	71.10%	74.20%
Φ_9	77.39%	71.39%	74.26%
Φ_{10}	77.51%	71.24%	74.24%
Φ_{11}	**77.53%**	**71.46%**	**74.37%**
Φ_{12}	77.38%	71.41%	74.27%
Φ_{13}	77.86%	70.99%	74.26%

We see that the accuracy increases slightly when the subcategorization feature (Φ_{11}) is removed. For this reason, we remove only the subcategorization feature. The best feature set includes the following features: predicate, phrase type, function tag, parse tree path,

distance, voice, position and head word. The accuracy of our system with this feature set is 74.37% of F_1 score.

4.2.5. Improvement via integer linear programming

Table 10. The impact of ILP

	Precision	Recall	F_1
A	77.53%	71.46%	74.37%
B	78.28%	71.48%	74.72%
C	78.29%	71.48%	**74.73%**

A: Without ILP
B: With ILP (not using constraint 5)
C: With ILP (using constraint 5)

As discussed previously, after classifying the arguments, we use ILP method to help improve the overall accuracy. In the sixth experiment, we set up an ILP to find the best performance satisfying constraints presented earlier[7]. The score $p_{ic} = score(S^i = c)$ is the signed distance of that argument to the hyperplane. We also compare our ILP system with the ILP method for English by using only constraints from 1 to 4. The improvement given by ILP is shown in Table 10. We see that ILP increases the performance of about 0.4% and when adding constraint 5, the result is slightly better. The accuracy of for each argument is shown in Table 11.

Table 11. Accuracy of each argument type

	Precision	Recall	F_1
Arg0	93.92%	97.34%	95.59%
Arg1	68.97%	82.38%	75.03%
Arg2	56.87%	46.62%	50.78%
Arg3	3.33%	5.00%	4.00%
Arg4	61.62%	22.01%	31.17%
ArgM-ADJ	0.00%	0.00%	0.00%
ArgM-ADV	60.18%	44.80%	51.17%
ArgM-CAU	61.96%	47.63%	50.25%
ArgM-COM	41.90%	78.72%	52.53%
ArgM-DIR	41.21%	23.01%	29.30%
ArgM-DIS	60.79%	56.37%	58.25%
ArgM-DSP	0.00%	0.00%	0.00%
ArgM-EXT	70.10%	77.78%	73.19%
ArgM-GOL	0.00%	0.00%	0.00%
ArgM-I	0.00%	0.00%	0.00%
ArgM-LOC	59.26%	75.56%	66.21%
ArgM-LVB	0.00%	0.00%	0.00%
ArgM-MNR	56.06%	52.00%	53.70%
ArgM-MOD	76.57%	84.77%	80.33%
ArgM-NEG	85.21%	94.24%	89.46%
ArgM-PRD	22.00%	13.67%	15.91%
ArgM-PRP	70.38%	70.96%	70.26%
ArgM-Partice	38.76%	17.51%	22.96%
ArgM-REC	45.00%	48.00%	45.56%
ArgM-RES	2.00%	6.67%	9.52%
ArgM-TMP	78.86%	93.09%	85.36%

A detailed investigation of our constituent extraction algorithm reveals that it can account for about 86% of possible argument candidates. Although this coverage ratio is relatively high, it is not exhaustive. One natural question to ask is whether an exhaustive search of argument candidates could improve the accuracy of the system or not. Thus, in the seventh experiment, we replace our constituent extraction algorithm by an exhaustive search where all nodes of a syntactic tree are taken as possible argument candidates. Then, we add the third constraint to the ILP post-processing step as presented above (*Arguments cannot overlap other arguments in the sentence*). An accuracy comparison of two constituent extraction algorithms is shown in Table 12.

Table 12. Accuracy of two extraction algorithms

	Getting All Nodes	Our Extraction Alg.
Precision	19.56%	82.15%
Recall	**93.25**%	86.12%
F_1	32.23%	84.08%

Taking all nodes of a syntactic tree help increase the number of candidate argument to a coverage ratio of 93.25%. However, it also proposes many wrong candidates as shown by a low precision ratio. Table 13 shows the accuracy of our system in the two candidate extraction approaches.

Table 13. Accuracy of our system

	Getting All Nodes	Our Extraction Alg.
Precision	77.99%	78.29%
Recall	62.50%	71.48%
F_1	69.39%	74.73%

[7] We use the GLPK solver provided by the PuLP software package, available at https://pythonhosted.org/PuLP/.

We see that an exhaustive search of candidates help present more possible constituent candidates but it makes the performance of the system worse than the constituent extraction algorithm (69.39% compared to 74.73% of F_1 ratio). One plausible explanation is that the more a classifier has candidates to consider, the more it is likely to make wrong classification decision, which results in worse accuracy of the overall system. In addition, a large number of candidates makes the system lower to run. In our experiment, we see the training time increased fourfold when the exhaustive search approach was used instead of our constituent extraction algorithm.

4.2.6. Learning curve

In the ninth experiment, we investigate the dependence of accuracy to the size of the training dataset. Figure 9 depicts the learning curve of our system when the data size is varied.

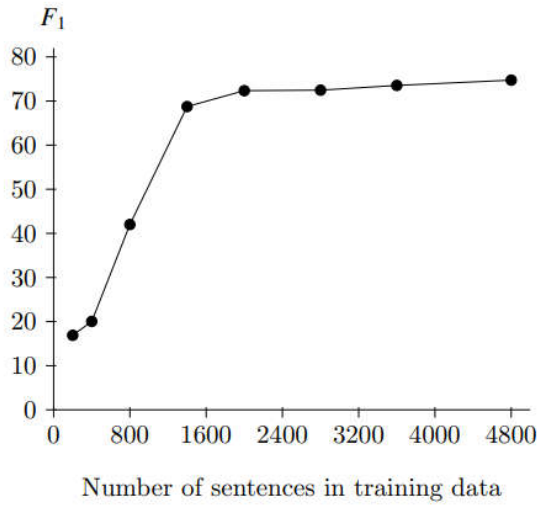


Figure 9. Learning curve of the system.

It seems that the accuracy of our system improves only slightly starting from the dataset of about 2,000 sentences. Nevertheless, the curve has not converged, indicating that the system could achieve a better accuracy when a larger dataset is available.

4.3. Generalizing to unseen words

In this section, we report our effort to extend the applicability of our SRL system to new text domain where rare or unknown words are common. As seen in the previous systems, some important features of our SRL system are word features including predicates and head words.

As in most NLP tasks, the words are usually encoded as symbolic identifiers which are drawn from a vocabulary. Therefore, they are often represented by one-hot vectors (also called indicator vectors) of the same length as the size of the vocabulary. This representation suffers from two major problems. The first problem is data sparseness, that is, the parameters corresponding to rare or unknown words are poorly estimated. The second problem is that it is not able to capture the semantic similarity between closely related words. This limitation of the one-hot word representation has motivated unsupervised methods for inducing word representations over large, unlabelled corpora.

Recently, distributed representations of words have been shown to be advantageous for many natural language processing tasks. A distributed representation is dense, low dimensional and real-valued. Distributed word representations are called word embeddings. Each dimension of the embedding represents a latent feature of the word which hopefully captures useful syntactic and semantic similarities [29].

Word embeddings are typically induced using neural language models, which use neural networks as the underlying predictive model. Historically, training and testing of neural language models has been slow, scaling as the size of the vocabulary for each model computation [30]. However, many approaches have been recently proposed to speed up the training process, allowing scaling to very large corpora [31, 32, 33, 34].

Another method to produce word embeddings has been introduced recently by the natural language processing group at the Stanford university [35]. They proposed a global log-bilinear regression model that combines the advantages of the two major model families in the literature: global matrix factorization and local context window methods.

We present in the subsections 4.3.1 and 4.3.2 how we use a neural language model and a global log-bilinear regression model, respectively, to produce word embeddings for Vietnamese which are used in this study.

4.3.1 Skip-gram Model

We use word embeddings produced by Mikolov's continuous Skip-gram model using the neural network and source code introduced in [36]. The continuous skip-gram model itself is described in details in [34].

For our experiments we used a continuous skip-gram window of size 2, *i.e.* the actual context size for each training sample is a random number up to 2. The neural network uses the central word in the context to predict the other words, by maximizing the average conditional log probability

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{j=-c}^{c}\log p(w_{t+j}\,|\,w_t), \qquad (9)$$

where $\{w_i : i \in T\}$ is the whole training set, w_t is the central word and the w_{t+j} are on either side of the context. The conditional probabilities are defined by the softmax function

$$p(a\,|\,b)=\frac{\exp(o_a^{\mathrm{T}} i_b)}{\sum_{w\in \mathsf{V}}\exp(o_w^{\mathrm{T}} i_b)}, \qquad (10)$$

where i_w and o_w are the input and output vector of w respectively, and V is the vocabulary. For computational efficiency, Mikolov's training code approximates the softmax function by the hierarchical softmax, as defined in [31]. Here the hierarchical softmax is built on a binary Huffman tree with one word at each leaf node. The conditional probabilities are calculated according to the decomposition:

$$p(a\,|\,b)=\prod_{i=1}^{l} p(d_i(a)\,|\,d_1(a)...d_{i-1}(a),b), \qquad (11)$$

where l is the path length from the root to the node a, and $d_i(a)$ is the decision at step i on the path (for example 0 if the next node the left child of the current node, and 1 if it is the right child). If the tree is balanced, the hierarchical softmax only needs to compute around $\log_2|\mathsf{V}|$ nodes in the tree, while the true softmax requires computing over all $|\mathsf{V}|$ words.

The training code was obtained from the tool word2vec[8] and we used frequent word subsampling as well as a word appearance threshold of 5. The output dimension is set to 50, *i.e.* each word is mapped to a unit vector in R^{50}. This is deemed adequate for our purpose without overfitting the training data. Figure 10 shows the scatter plot of some Vietnamese words which are projected onto the first two principal components after performing the principal component analysis of all the word distributed representations. We can see that semantically related words are grouped closely together.

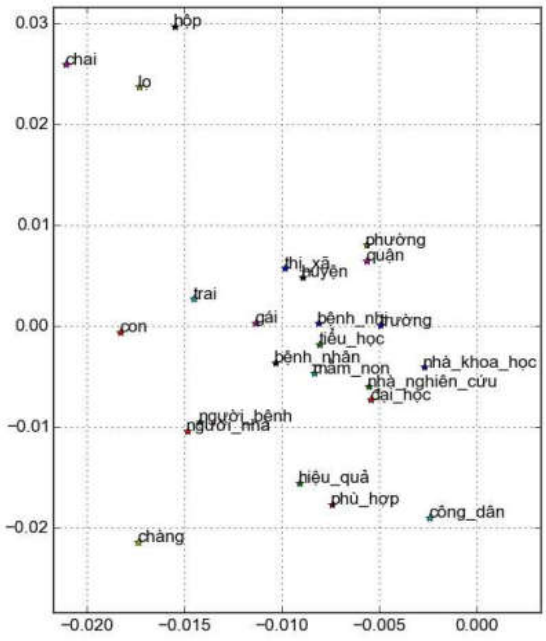


Figure 10. Some Vietnamese words produced by the Skip-gram model, projected onto two dimensions.

4.3.2. GloVe model

Pennington, Socher, and Manning [35] introduced the global vector model for learning word representations (GloVe). Similar to the Skip-gram model, GloVe is a local context window method but it has the advantages of the global matrix factorization method.

[8] http://code.google.com/p/word2vec/

The main idea of GloVe is to use word-word occurrence counts to estimate the co-occurrence probabilities rather than the probabilities by themselves. Let P_{ij} denote the probability that word j appear in the context of word i; $w_i \in \mathbb{R}^d$ and $w_j \in \mathbb{R}^d$ denote the word vectors of word i and word j respectively. It is shown that

$$w_i^{\mathrm{T}} w_j = \log(P_{ij}) = \log(C_{ij}) - \log(C_i), \quad (12)$$

where C_{ij} is the number of times word j occurs in the context of word i.

It turns out that GloVe is a global log-bilinear regression model. Finding word vectors is equivalent to solving a weighted least-squares regression model with the cost function:

$$J = \sum_{i,j=1}^{n} f(C_{ij})(w_i^{\mathrm{T}} w_j + b_i + b_j - \log(C_{ij}))^2, \quad (13)$$

where n is the size of the vocabulary, b_i and b_j are additional bias terms and $f(C_{ij})$ is a weighting function. A class of weighting functions which are found to work well can be parameterized as

$$f(x) = \begin{cases} \left(\frac{x}{x_{\max}}\right)^{\alpha} & \text{if} x < x_{\max} \\ 1 & \text{otherwise} \end{cases} \quad (14)$$

The training code was obtained from the tool GloVe[9] and we used a word appearance threshold of 2,000. Figure 11 shows the scatter plot of the same words in Figure 10, but this time their word vectors are produced by the GloVe model.

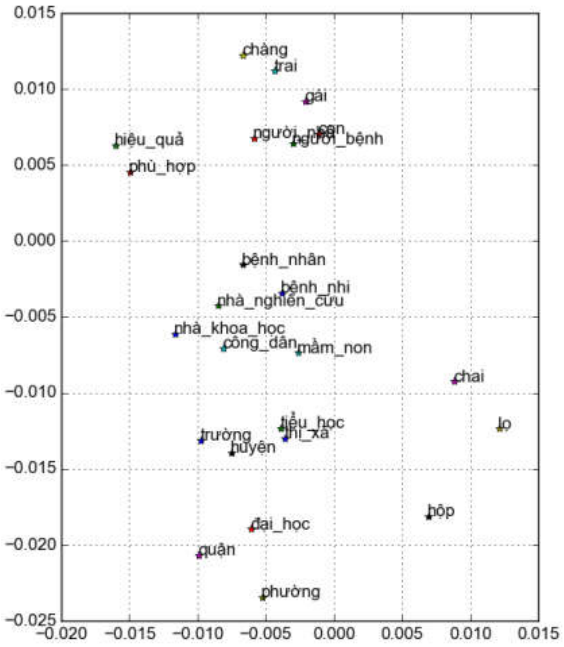


Figure 11. Some Vietnamese words produced by the GloVe model, projected onto two dimensions.

4.3.3. Text corpus

To create distributed word representations, we use a dataset consisting of 7.3GB of text from 2 million articles collected through a Vietnamese news portal[10]. The text is first normalized to lower case and all special characters are removed except these common symbols: the comma, the semicolon, the colon, the full stop and the percentage sign. All numeral sequences are replaced with the special token < number>, so that correlations between certain words and numbers are correctly recognized by the neural network or the log-bilinear regression model.

Each word in the Vietnamese language may consist of more than one syllables with spaces in between, which could be regarded as multiple words by the unsupervised models. Hence it is necessary to replace the spaces within each word with underscores to create full word tokens. The tokenization process follows the method described in [37].

After removal of special characters and tokenization, the articles add up to 969 million

[9] http://nlp.stanford.edu/projects/glove/

[10] http://www.baomoi.com

word tokens, spanning a vocabulary of 1.5 million unique tokens. We train the unsupervised models with the full vocabulary to obtain the representation vectors, and then prune the collection of word vectors to the 65,000 most frequent words, excluding special symbols and the token < number> representing numeral sequences.

4.3.4. SRL with distributed word representations

We train the two word embedding models on the same text corpus presented in the previous subsections to produce distributed word representations, where each word is represented by a real-valued vector of 50 dimensions.

In the last experiment, we replace predicate or head word features in our SRL system by their corresponding word vectors. For predicates which are composed of multiple words, we first tokenize them into individual words and then average their vectors to get vector representations. Table 14 and Table 15 shows performances of the Skip-gram and GloVe models for predicate feature and for head word feature, respectively.

Table 14. The impact of word embeddings of predicate

	Precision	Recall	F_1
A	78.29%	71.48%	74.73%
B	78.37%	71.49%	74.77%
C	78.29%	71.38%	74.67%

A: Predicate word
B: Skip-gram vector
C: GloVe vector

Table 15. The impact of word embeddings of head word

	Precision	Recall	F_1
A	78.29%	71.48%	74.73%
B	77.53%	70.76%	73.99%
C	78.12%	71.58%	74.71%

A: Head word
B: Skip-gram vector
C: GloVe vector

We see that both of the two types of word embeddings do not decrease the accuracy of the system. In other words, their use can help generalize the system to unseen words.

5. Conclusion

We have presented our work on developing a semantic role labelling system for the Vietnamese language. The system comprises two main component, a corpus and a software. Our system achieves a good accuracy of about 74.8% of F_1 score.

We have argued that one cannot assume a good applicability of existing methods and tools developed for English and other occidental languages and that they may not offer a cross-language validity. For an isolating language such as Vietnamese, techniques developed for inflectional languages cannot be applied "as is". In particular, we have developed an algorithm for extracting argument candidates which has a better accuracy than the 1-1 node mapping algorithm. We have proposed some novel features which are proved to be useful for Vietnamese semantic role labelling, notably and function tags and distributed word representations. We have employed integer linear programming, a recent inference technique capable of incorporate a wide variety of linguistic constraints to improve the performance of the system. We have also demonstrated the efficacy of distributed word representations produced by two unsupervised learning models in dealing with unknown words.

In the future, we plan to improve further our system, in the one hand, by enlarging our corpus so as to provide more data for the system. On the other hand, we would like to investigate different models used in SRL, for example joint models [38], where arguments and semantic roles are jointly embedded in a shared vector space for a given predicate. In addition, we would like to explore the possibility of integrating dynamic constraints in the integer linear programming procedure. We expect the overall performance of our SRL system to improve.

Our system, including software and corpus, is available as an open source project for free research purpose and we believe that it is a

good baseline for the development and comparison of future Vietnamese SRL systems[11]. We plan to integrate this tool to Vitk, an open-source toolkit for processing Vietnamese text, which contains fundamental processing tools and are readily scalable for processing very large text data[12].

References

[1] Shen, D., and Lapata, M. 2007, "Using semantic roles to improve question answering", In Proceedings of Conference on Empirical Methods on Natural Language Processing and Computational Natural Language Learning, Czech Republic: Prague, pp. 12–21.

[2] Lo, C. K., and Wu, D. 2010, "Evaluating machine translation utility via semantic role labels", In Proceedings of The International Conference on Language Resources and Evaluation, Malta: Valletta, pp. 2873–7.

[3] Aksoy, C., Bugdayci, A., Gur, T., Uysal, I., and Can, F. 2009, "Semantic argument frequency-based multi-document summarization", In Proceedings of the 24th of the International Symposium on Computer and Information Sciences, Guzelyurt, Turkey, pp. 460–4.

[4] Christensen, J., Soderland, S., and Etzioni, O. 2010, "Semantic role labeling for open information extraction", In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies, USA: Los Angeles, CA, pp. 52–60.

[5] Gildea, D., and Jurafsky D. 2002, "Automatic labeling of semantic roles", Computational Linguistics, 28(3): 245–88.

[6] Carreras, X., and Màrquez, L. 2004, "Introduction to the CoNLL-2004 shared task: semantic role labeling", In Proceedings of the 8th Conference on Computational Natural Language Learning, USA: Boston, MA, pp. 89–97.

[7] Carreras X., and Màrquez, L. 2005, "Introduction to the CoNLL-2005 shared task: semantic role labeling", In Proceedings of the 9th Conference on Computational Natural Language Learning, USA: Ann Arbor, MI, pp. 152–64.

[8] Xue, N., and Palmer, M. 2005, "Automatic semantic role labeling for Chinese verbs", In Proceedings of International Joint Conferences on Artificial Intelligence, Scotland: Edinburgh, pp. 1160–5.

[9] Tagami, H., Hizuka, S., and Saito, H. 2009, "Automatic semantic role labeling based on Japanese FrameNet–A Progress Report", In Proceedings of Conference of the Pacific Association for Computational Linguistics, Japan: Hokkaido University, Sapporo, pp. 181–6.

[10]Nguyen, T.-L., Ha, M.-L., Nguyen, V.-H., Nguyen, T.-M.-H., Le-Hong, P. and Phan, T.-H. 2014, "Building a semantic role annotated corpus for Vietnamese", in Proceedings of the 17th National Symposium on Information and Communication Technology, Daklak, Vietnam, pp. 409–414.

[11] Pham, T. H., Pham, X. K., and Le-Hong, P. 2015, "Building a semantic role labelling system for Vietnamese", In Proceedings of the 10th International Conference on Digital Information Management", South Korea: Jeju Island, pp. 77–84

[12] Baker, C. F., Fillmore, C. J., and Cronin, B. 2003, "The structure of the FrameNet database", International Journal of Lexicography, 16(3): 281–96.

[13] Boas, H. C. 2005, "From theory to practice: Frame semantics and the design of FrameNet", Semantisches Wissen im Lexikon: 129–60.

[14] Palmer, M., Kingsbury, P., and Gildea, D. 2005. "The proposition bank: An annotated corpus of semantic roles", Computational Linguistics, 31(1): 71–106.

[15 Schuler, K. K. 2006, "VerbNet: A broad-coverage, comprehensive verb lexicon", PhD Thesis, University of Pennsylvania.

[16] Levin, B. 1993, "English Verb Classes and Alternation: A Preliminary Investigation", Chicago: The University of Chicago Press.

[17] Cao, X. H. 2006, "Tiếng Việt - Sơ thảo ngữ pháp chức năng (Vietnamese - Introduction to Functional Grammar)", Hà Nội: NXB Giáo dục

[18] Nguyễn, V. H. 2008, "Cơ sở ngữ nghĩa phân tích cú pháp (Semantic Basis of Grammatical Parsing)", Hà Nội: NXB Giáo dục.

[19] Diệp, Q. B. 1998, "Ngữ pháp tiếng Việt, Tập I, II (Vietnamese Grammar, Volume I, II)", Hà Nội: NXB Giáo dục.

[20] Nguyen, P. T., Vu, X. L., Nguyen, T. M. H., Nguyen, V. H., and Le-Hong, P. 2009, "Building a large syntactically-annotated corpus of Vietnamese", In Proceedings of the 3rd Linguistic Annotation Workshop, ACL-IJCNLP, Singapore: Suntec City, pp. 182–5.

[21] Koomen, P., Punyakanok, V., Roth, D., and Yih, W. T. 2005, "Generalized inference with multiple semantic role labeling systems", In Proceedings of the 9th Conference on Computational Natural Language Learning, USA: Ann Arbor, MI, pp. 181–4.

[11] https://github.com/pth1993/vnSRL
[12] https://github.com/phuonglh/vn.vitk

[22] Haghighi, A., Toutanova, K., and Manning, C. D. 2005, "A joint model for semantic role labeling", In Proceedings of the 9th Conference on Computational Natural Language Learning, USA: Ann Arbor, MI, pp. 173–6.

[23] Surdeanu, M., and Turmo, J. 2005, "Semantic role labeling using complete syntactic analysis", In Proceedings of the 9th Conference on Computational Natural Language Learning, USA: Ann Arbor, MI, pp. 221–4.

[24] Màrquez, L., Comas, P., Gimenez, J., and Catala, N. 2005, "Semantic role labeling as sequential tagging", In Proceedings of the 9th Conference on Computational Natural Language Learning, USA: Ann Arbor, MI, pp. 193–6.

[25] Pradhan, S., Hacioglu, K., Ward, W., Martin, J. H., and Jurafsky, D. 2005, "Semantic role chunking combining complementary syntactic views", In Proceedings of the 9th Conference on Computational Natural Language Learning, USA: Ann Arbor, MI, pp. 217–20.

[26] Le-Hong, P., Roussanaly, A., and Nguyen, T. M. H. 2015, "A syntactic component for Vietnamese language processing", Journal of Language Modelling, 3(1): 145–84.

[27] Xue, N., and Palmer, M. 2004, "Calibrating features for semantic role labeling", In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, Spain: Barcelona, pp. 88–94.

[28] Punyakanok, V., Roth, D., Yih, W. T., and Zimak, D. 2004, "Semantic role labeling via integer linear programming inference" In Proceedings of the 20th International Conference on Computational Linguistics, Switzerland: University of Geneva, pp. 1346–52.

[29] Turian, J., Ratinov, L., and Bengio, Y. 2010, "Word representations: A simple and general method for semi-supervised learning", In Proceedings of ACL, Sweden: Uppsala, pp. 384–94.

[30] Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. 2003, "A neural probabilistic language model", Journal of Machine Learning Research 3: 1137–55.

[31] Morin, F., and Bengio, Y. 2005, "Hierarchical probabilistic neural network language model", In Proceedings of AISTATS, Barbados, pp. 246–52.

[32] Collobert, R., and Weston, J. 2008, "A unified architecture for natural language processing: deep neural networks with multitask learning", In Proceedings of ICML, USA: New York, NY, pp. 160–7.

[33] Mnih, A., and Hinton, G. E. 2009, "A scalable hierarchical distributed language model", In Koller, D., Schuurmans, D., Bengio, Y., and Bottou, L. (ed.) Advances in Neural Information Processing Systems 21, Curran Associates, Inc. pp. 1081–8.

[34] Mikolov, T., Chen, K., Corrado, G., and Dean, J. 2013, "Efficient estimation of word representations in vector space", In Proceedings of Workshop at ICLR, USA: Scottsdale, AZ, pp. 1–12.

[35] Pennington, J., Socher, R., and Manning, C. D. 2014, "GloVe: Global vectors for word representation", In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing", Qatar: Doha, pp. 1532–43

[36] Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. 2013, "Distributed representations of words and phrases and their compositionality", In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q. (ed.), Advances in Neural Information Processing Systems 26, Curran Associates, Inc. pp. 3111–19.

[37] Le-Hong, P., Nguyen, T. M. H, Roussanaly, A., and Ho, T. V. 2008, "A hybrid approach to word segmentation of Vietnamese texts", In Carlos, M-V., Friedrich, O., and Henning, F. (ed.), Language and Automata Theory and Applications, Lecture Notes in Computer Science. Berlin: Springer Berlin Heidelberg, pp. 240–49.

[38] FitzGerald, N., Täckström, O., Ganchev, K., and Das, D. 2015, "Semantic role labeling with neural network factors", In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Portugal: Lisbon, pp. 960–70.