Information and Media Technologies 3 (1) : 112–130 (2008)
reprinted from: Journal of Natural Language Processing 15 (1) : 3–21 (2008)

Using Semi-supervised Learning for Question Classification

Tri Thanh Nguyen†, Le Minh Nguyen† and Akira Shimazu†

Question classification, an important phase in question answering systems, is the task of identifying the type of a given question among a set of predefined types. This study uses unlabeled questions in combination with labeled questions for semi-supervised learning, to improve the precision of question classification task. For semi-supervised algorithm, we selected Tri-training because it is a simple but efficient co-training style algorithm. However, Tri-training is not well suitable for question data, so we give two proposals to modify Tri-training, to make it more suitable. In order to enable its three classifiers to have different initial hypotheses, Tri-training bootstrap-samples the originally labeled set to get different sets for training the three classifiers. The precisions of three classifiers are decreased because of the bootstrap-sampling. With the purpose to avoid this drawback by allowing each classifier to be initially trained on the originally labeled set while still ensuring the diversity of three classifiers, our first proposal is to use multiple algorithms for classifiers in Tri-training; the second proposal is to use multiple algorithms for classifiers in combination with multiple views, and our experiments show promising results.

Key Words: *Computational Linguistics, Question classification, Semi-supervised learning, Tri-training algorithm*

1 Introduction

Question classification is the task of identifying the type of a given question among a predefined set of question types. The type of a question can be used as a clue to narrow down the search space to extract the answer, and used for query generation in a question-answering (QA) system (Li and Roth 2002). Therefore, it has a significant impact on the overall performance of QA systems.

There have been several studies to solve this problem focusing on supervised learning (Zhang and Lee 2003; Kadri and Wayne 2003; Li and Roth 2002). However, the cost of making labeled (training) data is high, and a large training data set is needed to make significant impact on the performance. Also the above methods do not use unlabeled questions, which are readily available to improve the performance of classification. In order to utilize both labeled and unlabeled data, we propose to use semi-supervised learning. For the semi-supervised learning algorithm, we adopted the Tri-training (Zhou and Li 2005), since it has a simple but efficient method of

† School of Information Science, Japan Advanced Institute of Science and Technology

Information and Media Technologies 3(1): 112–130 (2008)
reprinted from: Journal of Natural Language Processing 15(1): 3–21 (2008)

deciding how to label an unlabeled instance (Nguyen et al. 2006). Tri-training uses three classifiers of the same algorithm, and if any two classifiers of the three classifiers predict the same label for an unlabeled instance, while the confidence of the labelling of the classifiers are not needed to be explicitly measured, then that instance is used for further training the other classifier. Such simplicity gives Tri-training advantages over other Co-training algorithms, such as the Co-training algorithm presented by (Goldman and Zhou 2000), which frequently uses 10-fold cross validation on the labeled set to determine how to label the unlabeled instances and how to produce the final hypothesis. If the original labeled set is rather small, cross validation will give high variance and is not useful for model selection.

The simplicity also makes Tri-training faster than the algorithm of Goldman, in which the frequent use of cross validation makes the learning process time-consuming. At the beginning, Tri-training bootstrap-samples the labeled data to generate different training sets for three classifiers in order to make the three classifiers diverse enough so that the Tri-training algorithm does not degenerate into *self-training* (Nigam and Ghani 2000) with a single classifier. However, question data is sparse and imbalanced. A question class may include only a few questions in a corpus, so if the bootstrap-sampling procedure duplicates some questions while omitting some questions in the classes with few questions, then classifiers being trained on these bootstrap-sampled sets have higher error rates than those of classifiers being trained on the labeled set. In order to avoid this drawback, while still keeping classifiers diverse, we propose to use more than one classifier with different algorithms. The original training set is initially used by the three classifiers without bootstrap-sampling. Another proposal is to apply more than one *views* (feature spaces) in the learning process. This allows the three classifiers to initially be trained from the labeled set with different feature spaces and still have diversity. In the second proposal, for the sake of simplicity, in the experiments, we used two different classification algorithms: Support Vector Machines (Cortes and Vapnik 1995) and Maximum Entropy Models (Berger et al. 1996) in combination with two views: *bag-of-word* and *bag-of-pos&word* features. Two classifiers which use the first algorithm are assigned different views, i.e., the first classifier gets bag-of-word and the other gets bag-of-pos&word features. The third classifier uses the second algorithm with bag-of-word features. With this strategy, three classifiers have initially different hypotheses. Our experiments show promising results.

The remainder of the paper is organized as follows: Section 2 gives summaries of related work; Section 3 gives details about the Tri-training algorithm and our modifications. Section 4 describes data sets and feature selection. The experimental results are given in Section 5 and conclusions are given in Section 6.

Information and Media Technologies 3 (1) : 112–130 (2008)
reprinted from: Journal of Natural Language Processing 15 (1) : 3–21 (2008)

2 Related work

There are two broad classes of approaches to question classification: rule-based and statistical. In rule-based approaches, an expert manually constructs a number of regular expressions and keywords corresponding to each type of question. Meanwhile, in statistical approaches, a model is assumed and trained on a sufficiently large set of labelled questions in order to automatically find out useful patterns for classification.

Statistical approach have advantages over rule-based approach, because they require less expert labor and are easily portable to other domains. Thus, recent work has concentrated on the approach, especially on the supervised learning approach which is a branch of the statistical approach.

(Zhang and Lee 2003) and (Li and Roth 2002) explored different types of features for improving the classification accuracy. Zhang and Lee considered *bag-of-word*, *bag-of-ngram* (all continuous word sequences in a question) features. Especially, they proposed a kernel function called *tree kernel* to enable support vector machine (SVM) to take advantage of the syntactic structures of questions. Li and Roth focused on several features: *words*, *pos tags*, *chunks* (non overlapping phrases), *named entities*, *head chunks* (e.g., the first noun chunk in a question) and *semantically related words* (words that often occur in a specific question type). They also used hierarchical classifiers, in which a question is classified by two classifiers: the first one classifies it into a coarse category; the second determines the fine category from the result produced by the first classifier. (Kadri and Wayne 2003) employed error correcting codes in combination with support vector machine to improve the results of classification.

3 Tri-training semi-supervised learning and its modifications

In this section, we describe the original Tri-training algorithm and give two proposals to improve it.

3.1 Semi-supervised Tri-training algorithm

In the Tri-training algorithm (Zhou and Li 2005), three classifiers: h_1, h_2 and h_3 are initially trained from a set by bootstrap-sampling the labeled set L. For any classifier, an unlabeled instance can be labeled as long as the other two classifiers predict the same label. For example, if h_1 and h_2 agree on the labelling of an instance x in the unlabeled set U, then x can be labeled for h_3. Obviously, in this scheme, if the prediction of h_1 and h_2 on x is correct, then h_3 will

Information and Media Technologies 3 (1) : 112–130 (2008)
reprinted from: Journal of Natural Language Processing 15 (1) : 3–21 (2008)

receive a valid new instance for further training; otherwise, h_3 will get an instance with a noisy label. Nonetheless, as claimed in (Zhou and Li 2005), even in the worse case, the increase in the classification noise rate can be compensated for, if the number of newly labeled instances is sufficient.

Also in the algorithm, each classifier is initially trained from a data set generated by bootstrap-sampling the original labeled set, in order to make classifiers diverse. If all the classifiers are identical, then for any of three classifiers, the unlabeled instances labeled by the other two classifiers will be the same as those labeled by itself, thus, Tri-training becomes *self-training* with

1 **tri-training**$(L,U,Learn)$
2 **for** $i \in \{1..3\}$ **do**
3 $S_i \leftarrow BootstrapSample(L)$
4 $h_i \leftarrow Learn(S_i)$
5 $e'_i \leftarrow 0.5;\ l'_i \leftarrow 0$
6 **end for**
7 **repeat until** none of h_i $(i \in \{1..3\})$ changes
8 **for** $i \in \{1..3\}$ **do**
9 $L_i \leftarrow \emptyset; update_i \leftarrow FALSE$
10 $e_i \leftarrow MeasureError(h_j \& h_k)$ $(j,k \neq i)$
11 **if** $(e_i < e'_i)$ **then**
12 **for** every $x \in U$ **do**
13 **if** $h_j(x) = h_k(x)$ $(j,k \neq i)$
14 **then** $L_i \leftarrow L_i \cup \{(x, h_j(x))\}$
15 **end for**
16 **if** $(l'_i = 0)$ **then** $l'_i \leftarrow \lfloor \frac{e_i}{e'_i - e_i} + 1 \rfloor$
17 **if** $(l'_i < |L_i|)$ **then**
18 **if**$(e_i |L_i| < e'_i l'_i)$ **then** $update_i \leftarrow TRUE$
19 **else if** $l'_i > \frac{e_i}{e'_i - e_i}$
20 **then** $L_i \leftarrow Subsample(L_i, \lceil \frac{e'_i l'_i}{e_i} - 1 \rceil)$;
21 $update_i \leftarrow TRUE$
22 **end for**
23 **for** $i \in \{1..3\}$ **do**
24 **if** $update_i = TRUE$ **then**
25 $h_i \leftarrow Learn(L \cup L_i); e'_i \leftarrow e_i; l'_i \leftarrow |L_i|$
26 **end for**
29 **end repeat**
30 **Output**:$h(x) \leftarrow \arg\max_{y \in label} \sum_{i: h_i(x) = y} 1$

a) Original Tri-training algorithm

1 **tri-training**$(L,U,Learn_1, Learn_2, Learn_3)$
2 **for** $i \in \{1..3\}$ **do**
3
4 $h_i \leftarrow Learn_i(L)$
5 $e'_i \leftarrow 0.5;\ l'_i \leftarrow 0$
6 **end for**
...
25 $h_i \leftarrow Learn_i(L \cup L_i)$;
$e'_i \leftarrow e_i; l'_i \leftarrow |L_i|$
...

b) Tri-training with multiple learning algorithms

1 **tri-training**$(L,U,Learn_1, Learn_2, Learn_3)$
2 **for** $i \in \{1..3\}$ **do**
3
4 $h_i \leftarrow Learn_i(view_i(L))$
5 $e'_i \leftarrow 0.5; l'_i \leftarrow 0$
6 **end for**
...
25 $h_i \leftarrow Learn_i(view_i(L \cup L_i))$;
$e'_i \leftarrow e_i; l'_i \leftarrow |L_i|$
...

c) Tri-training with multiple learning algorithms and views

Fig. 1 Original and modified versions of Tri-training

Information and Media Technologies 3(1): 112–130 (2008)
reprinted from: Journal of Natural Language Processing 15(1): 3–21 (2008)

a single classifier. The pseudo-code of the algorithm is described in Fig. 1a, where *Learn* is a classification algorithm; S_i is a labeled set bootstrap-sampled from the labeled set L. e'_i is the error rate of h_i in the (t-1)th round. With the assumption that the beginning error rate is less than 0.5, therefore e'_i is initially set to 0.5; e_i is the error rate of h_i in the t^{th} round; L_i is the set of instances that are labeled for h_i in the t^{th} round; l'_i is the size of L_i at (t-1)th round, and in the first round it is estimated by $\lfloor \frac{e_i}{e'_i - e_i} + 1 \rfloor$; $Subsample(L_i, s)$ function randomly removes $|L_i| - s$ number of instances from L_i in order to make current round have better performance than that of the previous round, as proved in (Zhou and Li 2005); $MeasureError(h_j \& h_k)$ function attempts to estimate the classification error rate of the hypothesis derived from the combination of h_j and h_k. Because it is difficult to estimate the classification error rate on the unlabeled instances, the algorithm only estimates on the labeled set with the assumption that both the labeled and unlabeled instance sets have the same distribution. In each iteration, L_i is not merged with the original labeled set L. It is put into the unlabeled set U as unlabeled instances.

The interesting point in the Tri-training algorithm is that, in order to ensure that the current round of training has better performance than that of the previous round, the size of each newly labelled set L_i must not be greater than $\lceil \frac{e'_i l'_i}{e_i} - 1 \rceil$. If it is greater than this value, the function $Subsample(L_i, s)$ is used to randomly remove redundant instances. The three classifiers are refined in the training process, and the final hypothesis is produced via *majority voting*. For the sake of saving space, other details can be seen in (Zhou and Li 2005).

3.2 Modified versions of Tri-training

Due to its nature, question data type is very sparse and imbalanced as shown in Table 1. As stated in (Joachims 1998), text data type, when represented in the vector space model, is very sparse. For each document, the corresponding document vector contains only a few entries which are non-zero. A question contains quite a few words in comparison with a document, so question data is even more sparse than text data. Because of the imbalance, after bootstrap-sampling, each newly created labeled set misses a number of questions as compared to the original labeled set. If the missed questions are in a class which contains only few questions, then the initial error rate of each classifier increases when being trained from these data sets. The final improvement after learning sometimes does not compensate for this problem. In order to avoid this drawback, we propose to use more than one algorithm for the three classifiers. Each classifier is initially trained on the labeled set. Our experiments showed that, if the performance of one of the three classifiers is much better (or worse) than that of the others, the final result is not improved. For this reason, a constraint on three classifiers is that their performances are similar. The modified

Information and Media Technologies 3(1): 112–130 (2008)
reprinted from: Journal of Natural Language Processing 15(1): 3–21 (2008)

Table 1 Question distribution. #Tr and #Te are the number of labeled and testing questions.

Class	#Tr	#Te	Class	#Tr	#Te	Class	#Tr	#Te
ABBREV.	**86**	**9**	letter	9	0	country	155	3
abb	16	1	other	217	12	mountain	21	3
exp	70	8	plant	13	5	other	464	50
DESC.	**1162**	**138**	product	42	4	state	66	7
definition	421	123	religion	4	0	NUMERIC	**896**	**113**
description	274	7	sport	62	1	code	9	0
manner	276	2	substance	41	15	count	363	9
reason	191	6	symbol	11	0	date	218	47
ENTITY	**1250**	**94**	technique	38	1	distance	34	16
animal	112	16	term	93	7	money	71	3
body	16	2	vehicle	27	4	order	6	0
color	40	10	word	26	0	other	52	12
creative	207	0	HUMAN	**1223**	**65**	period	27	8
currency	4	6	group	47	6	percent	75	3
dis.med.	103	2	individual	189	55	speed	9	6
event	56	2	title	962	1	temp	8	5
food	103	4	description	25	3	size	13	0
instrument	10	1	LOCATION	**835**	**81**	weight	11	4
lang	16	2	city	129	18			

version is depicted in Fig. 1b, where $Learn_i$ stands for different algorithms. We omit other lines that are identical to those of the original algorithm in Fig. 1a.

Another proposal to avoid bootstrap-sampling is to use more than one views, such as two or three views in the learning process, so that each classifier can be trained from the original labeled set with different feature spaces while still making sure that they are diverse enough. The modified algorithm seems to have the standard Co-training style in the framework of Tri-training. The modified version according to this proposal is given in Fig. 1c, where $view_i(L)$ is the i^{th} view of the data set L. Other lines that are the same as those in Fig. 1a are ignored. One important aspect of Tri-training algorithm is the needless of redundant views, so it can be applied to problems which have only one view. In this domain, it is easy to get redundant views, that is the reason of this proposal.

4 Question data sets and feature selection

4.1 Question data sets

The Question Answering Track in Text Retrieval Conference (TREC) (Voorhees 1999, 2000,

Information and Media Technologies 3(1) : 112–130 (2008)
reprinted from: Journal of Natural Language Processing 15(1) : 3–21 (2008)

2001) defines six question classes, namely, *abbreviation*, *description*, *entity*, *human*, *location* and *numeric*. However, for a question answering system in an open domain, six classes are not sufficient enough. The larger the number of question classes, the better a QA system locates and extracts answers to questions (Li and Roth 2002). Hence, from six coarse classes defined by TREC, (Li and Roth 2002) proposed to divide questions into 50 fine-grained classes. We follow this proposal to classify questions into these finer-grained classes. In the experiments, the data sets were those used in (Li and Roth 2002) with the total of about 6000 questions (the exact number is 5952), of which 500 questions from TREC 10 (Voorhees 2001) were the test set, and 4 subsets of size 1000, 2000, 3000 and 4000 were created by randomly selecting from other 5500 questions. These data sets are all available on http://L2R.cs.uiuc.edu/∼cogcomp/. We used the 4 subsets as labeled sets, and created 4 correspondingly unlabeled sets by selecting questions that do not belong to the labeled sets.

The distribution of training and testing data is shown in Table 1, where the coarse classes are in capitals, followed by the corresponding fine classes. As listed in the table, some classes consist of few questions, such as 4 questions in the *currency* and *religion* classes.

4.2 Feature selection

In experiments, we used two primitive feature types which were automatically extracted for each question, namely, bag-of-word and bag-of-pos&word.

Question classification is a little different from text classification, because a question contains a small number of words, while a document can have a large number of words. In text classification, common words like 'what', 'is', etc. are considered to be "*stop-words*" and omitted as a dimension reduction step in the process of creating features. This is an important step in improving the performance of classification as proven in (Joachims 1998). However, these words are very important for question classification. Also, word frequencies play an important role in document classification, whereas those frequencies are usually equal to 1 in a question, thus, they do not significantly contribute to the classification precision. In order to keep these words while still reducing the dimension space, we used a preprocessing step: all verbs were restored into their infinitive forms. For example, the verb forms 'is', 'were', 'was', 'are' and 'am' were converted to 'be'; plural nouns were changed to their singular forms, such as 'children' was converted to 'child'; words having the CD (*cardinal number*) part-of-speech were made the same value, such as '1998', '2000', '12' were changed into '100'. Given the question:

Who was President of Afghanistan in 1994?

After the reduction step, it becomes:

Information and Media Technologies 3 (1) : 112–130 (2008)
reprinted from: Journal of Natural Language Processing 15 (1) : 3–21 (2008)

Who be President of Afghanistan in 100?

After the reduction step, the vector (or vocabulary) V of all distinct words of questions in the corpus was constructed. Let the size of V be N, then each question q was converted into a vector $(q_1, q_2, \ldots, q_N)$, where q_i is 1 if the word w_i in V appears in q, otherwise q_i is 0. These vectors of numbers were the input of classifiers.

Interestingly, this dimension reduction step makes SVM reach the precision of 81.4% training on 5500 questions, while the same features with SVM used in (Zhang and Lee 2003) gives the precision of 80.2% training on the same data set and with the same *linear* kernel.

For bag-of-pos&word features, each *word* in a question was converted into the form of *POS-word*, where *POS* is the part-of-speech tag of *word*. We also used the preprocessing step similarly to what applied to the process to generate bag-of-word features, for example 'how' was transformed into 'WRB-how', 'who' was converted to 'WP-who', 'are', 'is', 'am', 'were' and 'was' were converted to 'AUX-be', etc. Given the question:

Who was President of Afghanistan in 1994?

After the reduction step, it becomes:

WP-Who AUX-be NN-president IN-of NN-Afghanistan IN-in CD-100?

The process of converting questions into vectors of numbers was similar to that of bag-of-word features. There is a difference between bag-of-word and bag-of-pos&word features. A word, such as 'plan' may play different roles in different questions. It can be a verb in this question while being a noun in another one. The role of the word can be distinguished in bag-of-pos&word features, because it is converted into 'VB-plan' (if it is a verb) or 'NN-plan'(if it is a noun) as depicted in Fig. 2. The bag-of-word features do not have this ability, so the bag-of-pos&word features provide a richer set of features. Concretely, for the dataset used in our experiments, the size of the vocabulary V for bag-of-word and bag-of-pos&word features is 7953 and 9876, respectively. Thus, bag-of-pos&word features may make classification algorithms perform better

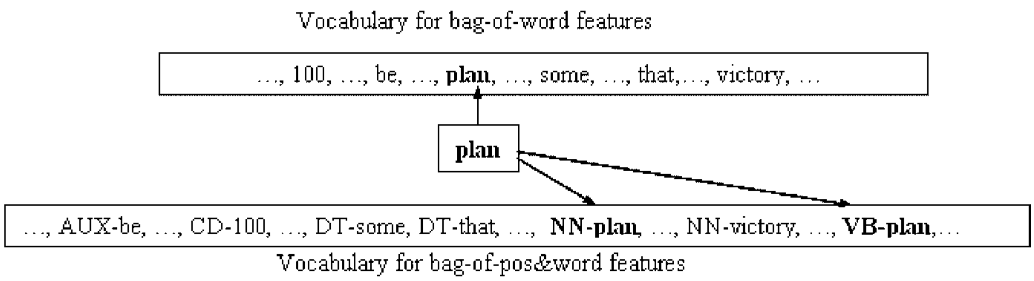

Fig. 2 The difference between bag-of-word and bag-of-pos&word features

Information and Media Technologies 3(1): 112–130 (2008)
reprinted from: Journal of Natural Language Processing 15(1): 3–21 (2008)

than bag-of-word features.

We tested the supervised learning with SVM algorithm on the labeled set of size 4000 with bag-of-word and bag-of-pos&word features. The statistics is recorded in Table 2, where $\#T$ shows the number of test questions belonging to each question class; $\#W$ and $\#P$, respectively, show the correctly predicted questions of each question class with bag-of-word and bag-of-pos&word; $\%W$ and $\%P$, respectively, are precisions of classification with bag-of-word and bag-of-pos&word. The table shows that SVM fails to classify some question classes, such as *currency*, *event* or *product* with bag-of-word and bag-of-pos&word features. SVM fails to classify the *currency* class because in the labeled set of size 4000, there is only one question belonging to the class *currency*. Another possible reason that make SVM fails to correctly classify other classes is the lack of semantics of bag-of-word and bag-of-pos&word as seen in the three questions from the labeled set:

+ *What is a fear of shadows?* in the class *ENTITY:disease.medicine.*
+ *What is the origin of head lice?* in the class *DESCRIPTION:description.*

Table 2 Precision of classification of SVM with bag-of-word and bag-of-pos&word features

Class	#T	#W	%W	#P	%P	Class	#T	#W	%W	#P	%P
abb	1	1	100	1	100	term	7	7	100	7	100
exp	8	6	75	6	75	vehicle	4	1	25	1	25
definition	123	123	100	123	100	HUM:desc	3	3	100	3	100
description	7	6	85.7	6	85.7	group	6	3	50	3	50
manner	2	2	100	2	100	individual	55	52	94.5	53	96.7
reason	6	5	83.3	5	83.3	title	1	0	0	0	0
animal	16	8	50	9	56.3	city	18	15	83.3	14	77.8
body	2	1	50	2	100	country	3	3	100	3	100
color	10	10	100	10	100	mountain	3	2	66.7	2	66.7
currency	6	0	0	0	0	LOC:other	50	41	82	41	82
dis.med	2	0	0	1	50	state	7	7	100	7	100
event	2	0	0	0	0	count	9	9	100	9	100
food	4	1	25	1	25	date	47	44	93.6	44	93.6
instrument	1	1	100	1	100	distance	16	9	56.3	8	50
lang	2	2	100	2	100	money	3	0	0	0	0
ENT:other	12	6	50	5	41.7	NUM:other	12	5	41.7	5	41.7
plant	5	1	20	1	20	percent	3	0	0	1	33.3
product	4	0	0	0	0	period	8	7	87.5	7	87.5
sport	1	1	100	1	100	speed	6	3	50	3	50
substance	15	6	40	5	33.3	temp	5	0	0	0	0
technique	1	1	100	1	100	weight	4	1	25	1	25
						TOTAL	500	393	78.6	395	79

+ *What is the nickname for the state of Mississippi?* in the class *LOCATION:state*.

Though these three questions belong to different classes, they have relatively similar forms. This causes ambiguity for classification algorithms. For improving classification precision, semantic features should be added, such as class-specific *related words* used in (Li and Roth 2002). For each question class, class-specific *related words* are a list of words that frequently appear in this class. With this method, a word in a question may have both syntactic and semantic roles, thus the feature is better, and the classification precision is improved.

5 Experiments

This section gives details about our implementation and evaluation. Because the function *Subsample*(.) (in line 20 of Fig. 1a) uses randomness to remove redundant questions, so the set L_i generated for each h_i may be different in each run; the final result of each run may be different, and the result of the first run is not always the best one. Thus, in all experiments, each algorithm was run 4 times and the best as well as the average results were recorded.

5.1 Experiments with multiple classifiers

In the first experiment, we developed our programs based on the Sparse Network of Winnows (SNoW) learning architecture[1] (Carlson et al. 1999), which implements three learning algorithms: Perceptron, Bayes and Winnow. We used these three learning algorithms to apply for the three classifiers of the Tri-training algorithm. Besides, we implemented the original Tri-training algorithm with a single classification algorithm, such as Bayes, Perceptron or Winnow. All the parameters of these algorithms, such as the learning rate α, threshold and the initial weight of Perceptron and Winnow were default values. The bag-of-word features were used in the experiment.

The best and average precision (of 4 runs) of the experiment is listed in Table 3, where TB, TP and TW respectively stand for the original Tri-training with a single classification algorithm Bayes, Perceptron and Winnow; TBPW stands for the modified Tri-training with Bayes, Perceptron and Winnow following the algorithm depicted in Fig. 1b. For the original Tri-training with a single classification algorithm Bayes, Perceptron or Winnow, we compare their precision with the baseline produced by the correspondingly supervised learning algorithm being trained on the same labeled set. For example, the baseline of the original Tri-training with Bayesian algorithm

[1] The software is freely available at http://L2R.cs.uiuc.edu/∼cogcomp/software.php

Information and Media Technologies 3(1): 112–130 (2008)
reprinted from: Journal of Natural Language Processing 15(1): 3–21 (2008)

Table 3 The best and average precision (%) of the original Tri-training with single algorithm (TB, TP and TW)and the modified Tri-training with Bayes, Perceptron and Winnow (TBPW)

The best precision								
	Bayes		Perceptron		Winnow		Mod. TriTraining	
#	Base.	TB	Base.	TP	Base.	TW	TBPW	N
1000	59.8	58.0	*60.2*	60.4	58.0	60.4	**65.8**	0
2000	58.4	58.0	*67.2*	67.8	67.0	64.8	**68.8**	1
3000	57.2	56.4	*68.4*	70.0	49.4	65.4	**72.0**	2
4000	51.8	51.8	66.4	65.8	*71.6*	71.4	**72.0**	6

The average precision							
	Bayes		Perceptron		Winnow		Mod. TriTraining
#	Base.	TB	Base.	TP	Base.	TW	TBPW
1000	59.8	55.85	*60.2*	60.15	58.0	59.85	**64.15**
2000	58.4	57.80	*67.2*	66.80	67.0	64.15	**68.60**
3000	57.2	56.30	*68.4*	69.35	49.4	65.00	**70.35**
4000	51.8	51.65	66.4	65.50	*71.6*	69.65	69.70

is the precision of the supervised learning of Bayes on the same labeled set. For our modified algorithm TBPW, we compared its precision with the best precision of individually supervised learning of the three classifiers (values in *italic*) as the baseline. We also carried out the sign test (Kanji 1994) for our modified Tri-training algorithm, with a total number of 25 subsets at the 95% significance level (p=0.05), in which the corresponding critical value is 7. The column 'N' shows the number of tests on subsets in which the precision of semi-supervised learning is less than the baseline. According to the sign test theory, a test is significant if the value in the column 'N' is less than or equal the critical value. The sign test shows that our algorithm is significant at the level of 95% for all tests.

The results show that the precision of supervised learning of Bayes, Perceptron and Winnow is not sensitive to the size of labeled sets. Concretely, when the size of the labeled set increases, the corresponding precision does not increase. Maybe, question data type and bag-of-word features are not suitable for these learning algorithms.

In the second experiment, we used two algorithms: Maximum Entropy Model[2](MEM), and SVM[3] which has been proven to perform well for text classification (Joachims 1998). The selection of MEM is based on our investigation. It has better performance than Bayes, Winnow and

[2] We used a free open source implementation of Maximum Entropy Model available at http://homepages.inf.ed.ac.uk/s0450736/pmwiki/pmwiki.php
[3] We used a free implementation of SVM available at http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/

Information and Media Technologies 3(1): 112–130 (2008)
reprinted from: Journal of Natural Language Processing 15(1): 3–21 (2008)

Perceptron. In this domain, SVM classifier has better performance than that of MEM classifier, thus, we used two SVM classifiers and one MEM classifier in the implementation with the expectation of making two SVM classifiers to have high degree of decision on final hypothesis. With SVM classifiers, we used *linear* kernel, and other parameters (e.g., parameter C) were default. In this domain, other kernels of SVM, such as *polynomial*, *radial basic function* or *sigmoid*, give poor performance. For MEM classifier, we used Gaussian smoothing, and all default values of parameters (e.g., L-BFGS parameter estimation). Bag-of-word features were used for all classifiers. In this configuration, the two SVM classifiers are identical at the beginning. In the learning loop, because of the randomness, the $Subsample(.)$ procedure (in the line 20 of the algorithm in Fig. 1) creates different L_i sets for the two SVM classifiers. As the results, the two SVM classifiers have different hypotheses when they are re-trained (in line 25 of the algorithm in Fig. 1 a).

Table 4 shows the best and the average precision (of 4 runs) of different algorithms, where TSW and TMW, respectively, stand for the original Tri-training algorithm with SVM and MEM algorithms; TSSM stands for the modified Tri-training with two SVM classifiers and a MEM classifier following the algorithm described in Fig. 1b. We used the precision of supervised learning with MEM and SVM on the same labeled sets as the baseline to compare with the precision of TMW and TSW. For TSSM, we selected the best precision of supervised learning with MEM and SVM on the same labeled sets (values in *italic*) as the baseline. Similar to our first experiment, we carried out the sign test on 25 subsets and at the 95% significance level. The

Table 4 The best and average precision (%) of the original Tri-training with single MEM, SVM algorithm (TMW and TSW) and the modified Tri-training with both MEM and SVM (TSSM)

The best precision						
	MEM		SVM		Mod. Tritraining	
#	Base.	TMW	Base.	TSW	TSSM	N
1000	67.6	68.0	*68.4*	67.6	68.4	-
2000	74.8	75.2	*75.6*	76.2	**76.4**	4
3000	76.8	76.4	*78.2*	78.4	**78.6**	6
4000	77.2	78.2	*78.6*	78.6	**78.8**	7

The average precision					
	MEM		SVM		Mod. Tritraining
#	Base.	TMW	Base.	TSW	TSSM
1000	67.6	67.40	*68.4*	66.95	68.25
2000	74.8	74.10	*75.6*	75.75	**76.10**
3000	76.8	76.20	*78.2*	78.00	78.20
4000	77.2	77.40	*78.6*	78.50	78.50

Information and Media Technologies 3(1): 112–130 (2008)
reprinted from: Journal of Natural Language Processing 15(1): 3–21 (2008)

column 'N' records the number of tests on subsets, in which the precision of semi-supervised is less than the baseline. Except for the test on the labeled set size of 1000 which is not improved, our other tests are significant at the level of 95%.

As shown in the table, MEM and SVM are sensitive to the size of the labeled sets. The precision is increased when the size of labeled set increases. This indicates that MEM and SVM are suitable for question data with bag-of-word features.

5.2 Experiments with two different algorithms and two views

In the third experiment, we implemented the second proposal of using more than one views following the algorithm described in Fig. 1c. In theory, we can use three different algorithms with distinct views, however, our primary purpose is to make the three classifiers diverse at the initial step, so two different algorithms, two views and a suitable assignment of views to classifiers are sufficient. Concretely, among the three classifiers, two of them were SVM classifiers and the third one was a MEM classifier. The first view (feature space) was bag-of-word, and the second view was bag-of-pos&word. We set two SVM classifiers two different views, while the MEM classifier used either of them. Concretely, the first SVM classifier used bag-of-word features, the second SVM classifier used bag-of-pos&word features and the MEM classifier used bag-of-word features.

Let TMW and TMP be the original Tri-training algorithm with MEM using bag-of-word and bag-of-pos&word features, respectively; Let TSW and TSP respectively be the original Tri-training with SVM using bag-of-word and bag-of-pos&word features; Let TSSM2 be the modified Tri-training with two SVM and a MEM classifiers following the algorithm described in Fig. 1c using two views: bag-of-word and bag-of-pos&word. For TMW, TMP, TSW and TSP, the baseline is the precision of supervised learning with corresponding algorithm and feature space. The best precision of the experiment is given in Table 5. The sign test similar to previous experiments is also carried out. Except for the test with the size of 1000, the other tests are

Table 5 The best precision (%) of the original Tri-training with single algorithm (TMW, TMP, TSW and TSP) and the modified Tri-training with MEM, SVM with two views (TSSM2)

	MEM-word		MEM-pos		SVM-word		SVM-pos		Two views	
#	Base.	TMW	Base.	TMP	Base.	TSW	Base.	TSP	TSSM2	N
1000	67.6	68.0	68.8	**69.0**	68.4	67.6	*69.2*	66.6	68.4	-
2000	74.8	75.2	75.4	74.2	*75.6*	76.2	75.2	74.6	**76.0**	5
3000	76.8	76.4	76.8	76.2	*78.2*	78.4	77.0	77.0	**79.0**	3
4000	77.2	78.2	77.8	77.8	78.6	78.6	*79.0*	78.4	**80.4**	2

Information and Media Technologies 3(1): 112–130 (2008)
reprinted from: Journal of Natural Language Processing 15(1): 3–21 (2008)

significant at the level of 95%.

We recorded the average precision (of 4 runs) of each algorithm of the experiment in Table 6. Table 7 recorded the number of new questions (Li) added for each classifier in each iteration of the experiment in Table 5, where 'Iter.' stands for iteration. The average values (in 4 tests) of these Li are recorded in Table 8. In these experiments, TMW, TMP, TSW and TSP took two iterations while TSSM2 took at most two iterations. The initial classifiers were very different because of the use of function *BootstrapSample*(.) in Line 3 of Fig. 1a, however after having

Table 6 The average precision (%) of the original Tri-training with single algorithm (TMW, TMP, TSW and TSP) and the modified Tri-training with MEM, SVM with two views (TSSM2)

	MEM-word		MEM-pos		SVM-word		SVM-pos		Two views
#	Base.	TMW	Base.	TMP	Base.	TSW	Base.	TSP	TSSM2
1000	67.6	67.40	68.8	68.55	68.4	66.95	*69.2*	66.30	67.90
2000	74.8	74.10	75.4	73.55	*75.6*	75.75	75.2	74.30	**75.85**
3000	76.8	76.20	76.8	75.90	*78.2*	78.00	77.0	76.85	**78.45**
4000	77.2	77.40	77.8	77.45	78.6	78.50	*79.0*	78.30	**79.65**

Table 7 The size of L_i in each round corresponding to the experiment in Table 5

#	Iter.	TMW			TMP			TSW		
		L_1	L_2	L_3	L_1	L_2	L_3	L_1	L_2	L_3
1000	1	30	30	5	42	42	5	26	30	30
	2	4262	4262	4452	4122	4122	4452	489	492	496
2000	1	50	50	6	32	47	30	34	23	28
	2	3199	3199	3452	3198	3216	3309	489	486	493
3000	1	41	36	48	54	81	42	41	33	32
	2	1491	1486	1471	2294	2351	2316	748	732	734
4000	1	40	41	43	65	47	55	46	39	42
	2	648	245	196	990	665	332	391	395	390

#	Iter.	TSP			TSSM2		
		L_1	L_2	L_3	L_1	L_2	L_3
1000	1	32	23	27	3226	499	499
	2	4256	4258	4236	—	—	—
2000	1	29	24	31	999	499	499
	2	984	974	975	—	—	—
3000	1	29	46	32	187	750	187
	2	1497	1497	1480	373	373	0
4000	1	29	39	37	222	399	199
	2	993	997	986	—	—	—

Table 8 The average size of L_i in each round corresponding to the experiments in Table 6

#	Iter.	TMW			TMP			TSW		
		L_1	L_2	L_3	L_1	L_2	L_3	L_1	L_2	L_3
1000	1	22	30.5	13.5	4.75	22	22	24	33	28.25
	2	4273	4212.25	4288	4018	3973	3979.5	490	494	488.5
2000	1	32.25	47.25	31.5	34.25	45.5	50	29	33	26.75
	2	3220.75	3203.25	3335.25	3221.5	3240.75	3225	488.25	490.5	488.75
3000	1	43	44	44	48.25	60	42	36	33.75	36.75
	2	1485.5	1480.75	1479.75	2297.75	2325.75	1904.5	740	733.75	741.5
4000	1	46.5	41	46	49.75	45	47	41.25	41.75	36
	2	655.25	554.5	512.25	527.5	641.75	672.25	395.5	395	390.5

#	Iter.	TSP			TSSM2		
		L_1	L_2	L_3	L_1	L_2	L_3
1000	1	30.25	28.75	25.75	3226	499	499
	2	4244.25	4236.75	4233.25	—	—	—
2000	1	26.75	30.25	28	999	499	499
	2	980	980	986.25	—	—	—
3000	1	29.75	31.25	34	187	750	187
	2	1474.25	1485.25	1476	302	283.5	0
4000	1	35.75	36.25	35	222	399	199
	2	983	982.5	988.25	—	—	—

been re-trained in Line 25 of Fig. 1, the three classifiers became very similar, and took many unlabelled questions in the second iteration, and stopped.

5.3 Experiments with self-training algorithm

This section implemented a self-training algorithm to compare the results with our modified Tri-training algorithm. In self-training, a single classifier is used to label questions in the unlabeled set to augment the labeled set for further training. The pseudo-code of the self-training algorithm is depicted in Fig. 3 (Nigam and Ghani 2000), where L, U are the labeled and unlabeled sets, correspondingly; θ is a threshold in the range of [0,1]; m is the number of iterations (m is 20 in our experiments); *Learn* is a classification algorithm; U' is a subset of unlabeled questions ($U' \subseteq U$); L' is a set of questions that are labeled at each iteration. In the training loop, we select a pool U' of unlabeled questions smaller than U, as suggested by (Blum and Mitchell 1998).

In each iteration, a subset U' of unlabeled questions is selected, and the set L' is created by selecting questions from U' which are predicted by the hypothesis h with confidence (prediction probability) greater than a threshold θ (θ is 0.9 in our experiments). The union of L' and L

Information and Media Technologies 3(1): 112–130 (2008)
reprinted from: Journal of Natural Language Processing 15(1): 3–21 (2008)


```
self-training(L, U, Learn, θ, m)
  Create a subset U′ by randomly selecting examples from U
  h ← Learn(L)
  repeat m times
      L′ ← ∅
      for every x ∈ U′
          if the prediction h(x) has the confidence greater than θ then
              L′ ← L′ ∪ {(x, h(x))}
      end for
      h ← Learn(L ∪ L′)
      Re-create the subset U′ by randomly selecting examples from U
  end repeat
Output: the learned hypothesis h
```

Fig. 3 Self-training algorithm

Table 9 The precision of self-training with SVM

#	1000		2000		3000		4000	
	Base.	Self.	Base.	Self.	Base.	Self.	Base.	Self.
Precision	68.4	65.8	75.6	73.4	78.2	76.2	78.6	78.4

is used to train the classifier. Note that L' is not merged with L in each iteration. Instead, it is regarded as unlabeled questions, and put back into the unlabeled set U again. The training process terminates after m iterations.

We carried out self-training on labeled sets of different size (1000, 2000, 3000 and 4000), and the classification algorithm is SVM with bag-of-word features. The results of our experiments are given in Table 9, where '*Base.*' is the precision of supervised learning which is used as the baseline; '*Self.*' is the precision of the self-training. The results show that most final precision of self-training is not improved. Though only questions in the unlabeled set U' with high prediction probability are selected to form the labeled set L', it can not guarantee that those questions are correctly predicted as our observation. Thus, in each iteration, the newly created labeled set may contain mislabeled questions, and the error rate may consequently increase. In general, the self-training is not well suitable for question data type with bag-of-word features.

5.4 Discussion

Through experiments we can see that self-training is not suitable for solving this task, because its method to add unlabeled questions for further training the classifier is not good. The original Tri-training algorithm has a better method of adding unlabeled questions based on the agreement

Information and Media Technologies 3(1): 112–130 (2008)
reprinted from: Journal of Natural Language Processing 15(1): 3–21 (2008)

of two classifiers. However, the bootstrap-sampling step may decrease the initial precision of each classifier and the final precision is hard to be improved. Our two proposals remove the bootstrap-sampling while still ensure the three classifiers to have different hypotheses, and the experiments have proved the proposals to be suitable.

6 Conclusion

This paper applied semi-supervised learning to exploit unlabeled questions to improve the performance of question classification task and proposed two ways of modifying the Tri-training algorithm presented by (Zhou and Li 2005) to make it more suitable for question data type. The proposals dealt with a problem at the initial step of Tri-training, where the original labeled set is bootstrap-sampled to generate three different labeled sets, in order to make the three classifiers have different hypotheses, which may make the initial error rate of each classifier increase. With the purpose of using the original labeled set for all classifiers, while ensuring that they are still diverse, in the first proposal, we used more than one learning algorithm for the three classifiers and the second proposal is to use multiple learning algorithms in combination with more than one views. Our experiments indicate that the performance is improved.

In the current implementation, we have not considered to select other better feature types, such as those used in (Li and Roth 2002). This is one interesting issue to explore in future to achieve higher precision.

Our modified versions of Tri-training algorithm do not have any constraints on data types, therefore, one more issue which is worth studying in the future is to apply these algorithms in other domains, such as text classification.

Acknowledgment

This study was supported by Japan Advanced Institute of Science and Technology, the 21^{st} Century COE Program: "Verifiable and Evolvable e-Society". We would like to express special thanks to the blind reviewer of this paper for his/her careful review as well as corrections and suggestions.

Information and Media Technologies 3(1): 112–130 (2008)
reprinted from: Journal of Natural Language Processing 15(1): 3–21 (2008)

Reference

Berger, A., Pietra, S. D., and Pietra, V. D. (1996). "A maximum entropy approach to natural language processing." *Computational Linguistics*, **22** (1), pp. 39–71.

Blum, A. and Mitchell, T. (1998). "Combining labeled and unlabeled data with co-training." *Proceedings of the 11th Annual Conference on Computational Learning Theory, Madison, WI*, pp. 92–100.

Carlson, A., Cumby, C., and Roth, D. (1999). "The SNoW learning architecture." *Technical Report UIUC-DCS-R-99-2101, UIUC Computer Science Department.*

Cortes, C., and Vapnik, V. (1995). "Support vector networks." *Machine Learning*, **20** (3), pp. 273–297.

Goldman, S. and Zhou, Y. (2000). "Enhancing supervised learning with unlabeled data." *Proceedings of the 17th International Conference on Machine Learning*, pp. 327–334.

Joachims, T. (1998). "Text categorization with Support vector machines: Learning with many relevant features." *Proceedings of ECML-98, the 10th European Conference on Machine Learning*, pp. 137–142.

Kanji, G. (1994). "100 Statistical tests," *SAGE Publications.*

Kadri, H. and Wayne, W. (2003). "Question classification with Support vector machines and error correcting codes." *Proceedings of NAACL/Human Language Technology Conference*, pp. 28–30.

Li, X. and Roth, D. (2002). "Learning question classifiers." *Proceedings of the 19th International Conference on Computational Linguistics*, pp. 556–562.

Nguyen, T. T., Nguyen, L. M., and Shimazu, A. (2006). "Using semi-supervised learning for question classification." *Proceedings of the 21st International Conference on the Computer Processing of Oriental Languages*, pp. 31–41.

Nigam, K. and Ghani, R. (2000). "Analyzing the effectiveness and applicability of co-training." *Proceedings of the 9th International Conference on Information and Knowledge (CIKM-2000)*, pp. 86–93.

Voorhees, E. (1999). "The TREC-8 Question answering track report." *Proceedings of the 8th Text Retrieval Conference (TREC8)*, pp. 77–82.

Voorhees, E. (2000). "The TREC-9 Question answering track." *Proceedings of the 9th Text Retrieval Conference (TREC9)*, pp. 71–80.

Voorhees, E. (2001). "Overview of the TREC 2001 Question answering track." *Proceedings of the 10th Text Retrieval Conference (TREC10)*, pp. 157–165.

Information and Media Technologies 3(1): 112–130 (2008)
reprinted from: Journal of Natural Language Processing 15(1): 3–21 (2008)

Zhang, D. and Lee, W. S. (2003). "Question classification using Support vector machine." *Proceedings of the 26th Annual International ACM SIGIR Conference*, pp. 26–32.

Zhou, Z. and Li, M. (2005). "Tri-training: Exploiting unlabeled data using three classifiers." *IEEE Transactions on Knowledge and Data Engineering*, **17** (11), pp. 1529–1541.

Tri Thanh Nguyen: received the Bachelor degree in Faculty of Information Technology, Vietnam National University of Hanoi (VNUH) in 1999, and Master degree at Asian Institute of Technology (AIT), Thailand in 2002. From February 2003, he worked as a lecturer for Faculty of Information Technology, College of Technology, VNUH. Since April 2005, he has been a PhD student in Natural Language Processing Laboratory, School of Information Science, Japan Advanced Institute of Science and Technology (JAIST).

Le Minh Nguyen: received the Bachelor in Information Technology from Hanoi University of Science, and Master degrees in Information Technology from VNUH, in 1998 and 2001, respectively. He received Doctoral degree in School of Information Science, JAIST in 2004. Since 2005, he has been a Post-doctoral Fellow at Natural Language Processing Laboratory, School of Information Science, JAIST. His research interests include Text Summarization, Natural Language Understanding, Machine Translation, and Information Retrieval.

Akira Shimazu: received the Bachelor and Master degrees in mathematics from Kyushu University in 1971 and 1973, respectively, and a Doctoral degree in Natural Language Processing from Kyushu University in 1991. From 1973 to 1997, he worked at Musashino Electrical Communication Laboratories of Nippon Telegram and Telephone Public Corporation, and at Basic Research Laboratories of Nippon Telegraph and Telephone Corporation. From 2002 to 2004, he was the president of the Association for Natural Language Processing. He has been a professor in the Graduate school of Information Science, JAIST since 1997.

(Received October 20, 2006)
(Revised April 3, 2007)
(Rerevised June 18, 2007)
(Accepted August 1, 2007)