Detecting Multiple Domains from User's Utterance in Spoken Dialog System

Seonghan Ryu, Jaiyoun Song, Sangjun Koo, Soonchoul Kwon, and Gary Geunbae Lee

Pohang University of Science and Technology, Pohang, Republic of Korea
{ryush,tichiel,giantpanda,theincluder,gblee}@postech.ac.kr

Abstract. Multi-domain spoken dialog system should be able to detect more than one domain from a user's utterance. However, it is difficult to train an accurate binary classifier of a domain based on only positive and unlabeled examples. This paper improves hierarchical clustering algorithm to automatically identify reliable negative examples among unlabeled examples. This paper also verifies three linkage criteria that measure the distance between two clusters. In experiments, the proposed method resulted in the highest gain of F_1 score compared to the existing methods.

Keywords: dialog system, domain selection, domain detection, learning from positive and unlabeled examples, hierarchical clustering, Support Vector Machine

1 Introduction

Spoken dialog system (SDS) provides natural language interface between human and computer. Especially, multi-domain SDS (MDSDS) provides dialog service to many domains including *restaurant guide*, *car navigation*, *movie guide*, and *movie ticketing*. MDSDS first selects domain that the user may desire and then performs domain-specific processes: natural language understanding, dialog management, and response generation. Therefore, the selection of the appropriate domain from a user's utterance is a bottleneck of MDSDS; the incorrect selection of domain drives MDSDS to generate nonsense response.

The domain selection component usually uses as a multi-class classifier which is trained from multi-domain corpora. However, the boundaries of domains are ambiguous in the real world [1]. For example, when a user says "*I'm planning to go to Busan*", the intended domain of the user could be *car navigation*, *hotel reservation*, or both. Therefore, for accurate service, the domain selection component should be able to detect more than one domain at one time from a user's utterance. We called this task multi-domain detection (MDD).

MDD is a multi-label classification problem and this problem can be solved by combining in-domain verifiers (IDVs); an IDV classifies whether a user's utterance belongs to that domain. The IDV can be implemented as a binary classifier (BC) trained from positive examples and negative examples. Initially, a BC can be trained by considering the target domain's corpus as positive examples and the rest of the domain's corpora as negative examples. However, that BC can cause many incorrect rejections because the rest of the domain's corpora have some in-domain utterances. So the nature of the rest of the domain's corpora is *unlabeled* examples, not necessarily *negative* examples.

In this paper, we solved the MDD task by using a two-step approach to train an accurate BC from only positive and unlabeled examples. In the first step, we automatically identified reliable negative examples among unlabeled examples. We first constructed one cluster of positive examples, and several clusters of unlabeled examples. Then we hierarchically merged clusters by fixing the cluster that was constructed from positive examples. We also verified three linkage criteria that measure the distance between two clusters.

In the second step, we trained a BC iteratively based on positive examples and the reliable negative examples identified in the first step. The obtained BC was more accurate than the basic BC that was trained using all the rest domain's corpora as negative examples. The effectiveness of this two-step approach in learning from positive and unlabeled examples has been demonstrated theoretically [2].

The remainder of this paper is organized as follows: Section 2 briefly introduces related work and explains the contributions of this paper compared to the related work. Section 3 describes the proposed method of hierarchical clustering in detail. Section 4 demonstrates the experimental design and results. Finally, Section 5 concludes the paper.

2 Related Work

Ryu et al. [1] first introduced the MDD task and proposed an automatic multi-domain label annotation method that uses a hierarchical domain model designed by humans. The method automatically assigns positive and negative labels to utterances based on their previously-annotated intents and named entities. The method performed well for a small-scale MDSDS. However, it is difficult for human to design complex hierarchical domain models. So the method can hardly be applied to large-scale MDSDS.

A similar task occurs in natural language question answering systems. In question answering systems, detecting multiple possible answer types for a question can give a chance to improve the answer. A two-step classification method can be used to solve this problem: the first step is to classify a question into several coarse-grained classes; the second step is to classify it into several fine-grained classes that belong to the coarse-grained classes [3]. However, this method is based on classical multi-

class classification and selects multiple answer types with an empirically determined confidence score threshold in the multi-class classification.

Some research considered learning from positive and unlabeled examples. PEBL [4] uses 1-disjunctive normal from (1-DNF) technique to identify reliable negative examples and then trains a support vector machine (SVM) iteratively. S-EM [2] first proposes a Spy technique to identify reliable negative examples and then uses Expectation Maximization (EM) [5] to train a Naïve Bayes classifier [6]. Roc-SVM [7] uses an existing Rocchio method [8] to identify reliable negative examples and then uses a classifier selection method to train an SVM iteratively. Biased-SVM [9] focused on biased formulation of SVM.

To our knowledge, no previous work applies learning from positive and unlabeled examples to MDD task. One contribution point of this paper is improving hierarchical clustering algorithm to identify reliable negative examples among unlabeled examples; we fixed a cluster which consists of positive examples at the beginning of hierarchical clustering. Another contribution point of this paper is verifying which linkage criterion works accurately for MDD task.

3 Methods

3.1 Hierarchical Clustering from Positive and Unlabeled Examples

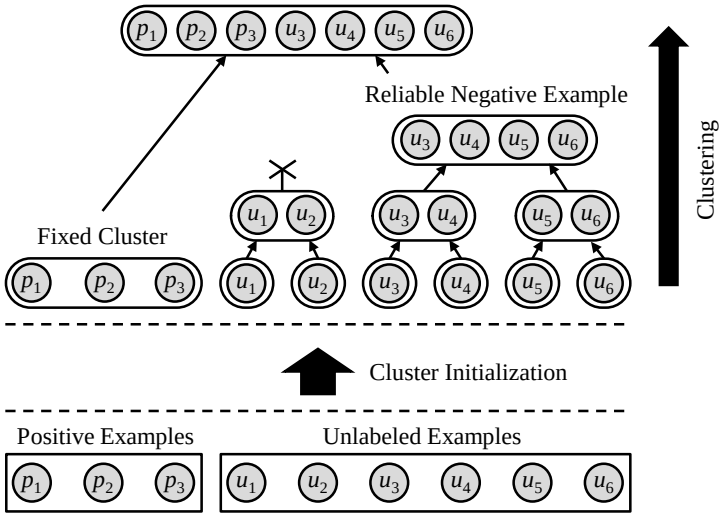

Fig. 1 An example of the proposed hierarchical clustering from positive and unlabeled examples: u_3, u_4, and u_5 are reliable negative examples.

Algorithm 1 Hierarchical clustering from positive and unlabeled examples

Input
- $P = \{P_1, \ldots P_{|P|}\}$: positive examples
- $U = \{U_1, \ldots U_{|U|}\}$: unlabeled examples
- t_d: maximum cluster distance criterion ($0 < t_d \leq 1$)
- t_k: minimum cluster number criterion ($2 \leq t_k < 1 + |U|$)

Local variable
- $C = \{C_1, \ldots C_{|C|}\}$: clusters
- $RN = \{RN_1, \ldots RN_{|RN|}\}$: reliable negative examples
- F: a fixed cluster

Output
- Reliable negative examples

1. $F \leftarrow \{P_1, ... P_{|P|}\}$;
2. $C \leftarrow \{F, U_1, ... U_{|U|}\}$;
3. **while** $|C| > t_k$ **do**
4. $(i^*, j^*) \leftarrow (null, null)$; $minDist \leftarrow 1.0$;
5. **foreach** (i, j), where $C_i \in C$, $C_j \in C$, and $C_i \neq C_j$ **do**
6. **if** $d_C(C_i, C_j) < minDist$ **then**
7. $(i^*, j^*) \leftarrow (i, j)$; $minDist \leftarrow d_C(C_i, C_j)$;
8. **if** $minDist < t_d$ **then**
9. **if** $C_{i^*} = F$ **then** $C \leftarrow C - C_{j^*}$;
10. **else if** $C_{j^*} = F$ **then** $C \leftarrow C - C_{i^*}$;
11. **else** $C \leftarrow C - C_{i^*} - C_{j^*}$; $C \leftarrow C \cup \{C_{i^*} \cup C_{j^*}\}$;
12. **else** escape loop;
13. **for** $i = 1$ to $|C|$, **where** $C_i \neq F$ **do**
14. **for** $k = 1$ to $|C_i|$ **do**
15. $RN \leftarrow RN \cup \{C_{i,k}\}$;
16. **return** RN;

3.1.1 Cluster Initialization

In our work, a cluster is a set of items and each item is a set of words in an example. We constructed clusters C from positive examples P and unlabeled examples U (Fig. 1; Algorithm 1 lines 1 – 2). We first constructed one cluster F of P and a total of $|U|$ clusters of each unlabeled example in U. So C initially consisted of $1 + |U|$ clusters.

3.1.2 Hierarchical Clustering

After initializing C, we performed the following hierarchical clustering iteratively (Algorithm 1 lines 3 – 12): merge two clusters X and Y when the distance $d_C(X, Y)$ between the two clusters is minimum in C. The distance measure is discussed in Section 3.2. When F and another cluster Z were selected to be merged during the iteration process, we did not perform the actual merge; we removed Z from C. We did this because we focused only identifying reliable negative examples and were

not interested in identifying additional positive examples. We terminated hierarchical clustering when the minimum distance between two clusters in *C* was too far or |*C*| was sufficiently small.

3.1.3 Reliable Negative Examples Selection

After hierarchical clustering, we regarded the remaining clusters in *C* as reliable negative examples *RN* except for *F* (Algorithm 1 lines 13 – 15). For example (Fig. 1), when $P = \{p_1, ... p_3\}$ and $U = \{u_1, ... u_6\}$, a total of seven clusters are constructed: $F = \{p_1, ... p_3\}$ from *P* and six clusters $\{u_1\}, \ldots \{u_6\}$ from each unlabeled example in *U*. When distance between $\{u_1, u_2\}$ and *F* reached a minimum value during the iteration process, $\{u_1, u_2\}$ was removed from *C*. When *F* and $\{u_3, ... u_6\}$ remain at the end of iteration, $\{u_3, ... u_6\}$ is selected as *RN* examples.

3.2 Linkage Criteria

We defined an item as a set of words. The distance $d_J(x, y)$ between two items *x* and *y* is the Jaccard distance:

$$d_J(x, y) = 1 - J(x, y) = 1 - \frac{|x \cap y|}{|x \cup y|}, \quad (1)$$

where $J(x, y)$ is the Jaccard similarity between *x* and *y*. So $|x \cap y|$ is the number of words in the intersection set of *x* and *y*; $|x \cup y|$ is the number of words in the union set of *x* and *y*. This Jaccard distance is used in all linkage criteria below.

In hierarchical clustering, the selected clusters to be merged are the closest pair based on one of the following linkage criteria: single linkage clustering $d_{SL}(X, Y)$, complete linkage clustering $d_{CL}(X, Y)$, or group average clustering $d_{GA}(X, Y)$ [10].

$$d_{SL}(X, Y) = \min_{x \in X, y \in Y} d_J(x, y) \quad (2)$$

$$d_{CL}(X, Y) = \max_{x \in X, y \in Y} d_J(x, y) \quad (3)$$

$$d_{GA}(X, Y) = \frac{1}{|X||Y|} \sum_{x \in X} \sum_{y \in Y} d_J(x, y) \quad (4)$$

However, some methods do not work well in MDD, because a domain in MDSDS includes various types of utterances. For example, both two utterances "*City Cinema in Gangnam*." and "*A Werewolf Boy seems interesting*!" are very different but both can be located in *F* for the *Movie Ticketing* domain. Therefore, a cluster can contain implicit sub-clusters: this structure can cause complete linkage clustering (3) and group average clustering (4) to fail.

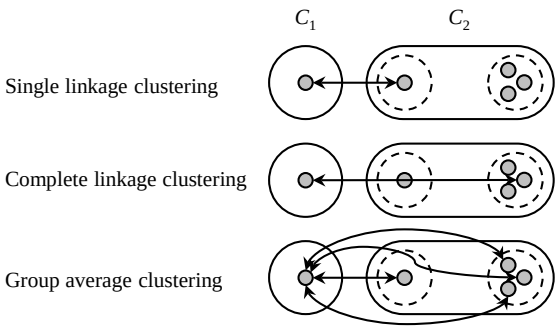

Fig. 2 Examples of the linkage criteria for MDD (filled circles: items, solid-line empty circles: clusters, dashed-line empty circles: implicit sub-clusters, arrows: considered distances in linkage criteria).

For example, C_1 is close to the left sub-cluster of C_2 (Fig. 2), where C_1 is constructed from an unlabeled example and C_2 is constructed from positive examples. Therefore, C_1 and C_2 should be merged. Single linkage clustering (2) gives a short distance between C_1 and C_2 so they are merged. In contrast, complete linkage clustering (3) and group average clustering (4) give a large distance between C_1 and C_2 so they are not merged. Therefore, we expected that single linkage clustering is more accurate than the other linkage criteria.

3.3 Iterative Training of the Binary Classifier

Algorithm 2 ***Iterative binary classifier training.***

Input
- $P = \{P_1, \ldots P_{|P|}\}$: positive examples
- $RU = \{RU_1, \ldots RU_{|RU|}\}$: remaining unlabeled examples
- $RN = \{RN_1, \ldots RN_{|RN|}\}$: reliable negative examples

Local variable
- Ω: binary classifier

Output
- Final binary classifier

1. **loop**
2. Train Ω using P and RN;
3. $N_{out} \leftarrow null$;
4. **for** $i = 1$ to $|RU|$ **do**
5. $c \leftarrow$ classify RU_i using Ω;
6. **if** c is negative **then** $N_{out} \leftarrow N_{out} \cup \{RU_i\}$;
7. **if** $|N_{out}| > 0$ **then** $RN \leftarrow RN \cup N_{out}$; $RU \leftarrow RU - N_{out}$;
8. **else** escape loop
9. **return** Ω;

We obtained the final BC by training BCs iteratively using positive examples P, reliable negative examples RN, and the remaining unlabeled examples RU (Algorithm 2). We first used P and RN to train a BC Ω. Then we classified RU and obtained a set of negative outputs N_{out}. We added N_{out} to RN and removed N_{out} from RU. We repeated this iteration until Ω converged. We used LIBSVM [11] as a BC: we used the radial basis function kernel, disabled shrinking heuristics, and used default settings for the remaining parameters in the experiment.

4 Experiments

4.1 Experimental Designs

Table. 1 The basic information of collected corpora.

Domain	Translated sentence example
D_1: Car Navigation	"*Please guide me the best path from Pohang to Gyeongju.*"
D_2: Civil Application Service	"*I want to renew my passport.*"
D_3: Home Control	"*What is in my refrigerator*?"
D_4: Movie Ticketing	"*City Cinema in Gangnam*."
D_5: Traffic Guide	"*I'm planning to go to Busan*."
D_6: Travel Reservation	"*I'm going to take a trip from Seoul to Busan*."
D_7: Weather Information	"*How will the weather be on Sunday*?"

We prepared Korean corpora of seven domains (Table 1). We used 80% of the corpora as training data and 20% of the corpora as test data. Each BC used the target domain's corpus as positive examples and the other six domains' corpora as unlabeled examples. For evaluation we labeled the test data as positive or negative.

We performed experiments with three existing methods and the proposed method.

- Baseline: we trained the SVM of a domain by using the rest six domain's corpora as negative examples directly.
- SVM: we trained the one-class SVM (O-SVM) [12] of a domain by using the target domain's corpus as positive examples and no negative examples.
- PEBL: we trained the SVM of a domain based on the PEBL framework [4].
- HCPU: we trained the SVM of a domain based on our hierarchical clustering from positive and unlabeled examples (HCPU). In HCPU, we tried three different linkage criteria: single linkage clustering (HCPU-SL), complete linkage clustering (HCPU-CL), and group average clustering (HCPU-GA).

We evaluated MDD performance by measuring the precision, recall, and F_1 score of each domain's BC. We also computed the macro-average precision, recall, and F_1 score.

4.2 Experimental Results

Table. 2 The precision, recall, and F_1 scores of MDD.

(a) Precision

	Baseline	**O-SVM**	**PEBL**	**HCPU**		
				SL	**CL**	**GA**
D_1	0.9535	0.7334	0.9286	0.9322	**0.9537**	0.9514
D_2	**0.9370**	0.7547	0.9132	0.9189	0.9353	0.9344
D_3	0.8631	0.7188	0.8503	**0.8681**	0.8629	0.8611
D_4	0.9138	0.7701	0.8824	0.9111	**0.9175**	0.9122
D_5	0.9085	0.7388	0.8487	0.9031	**0.9126**	0.9100
D_6	0.8561	0.7866	0.8555	**0.9117**	0.8539	0.5814
D_7	**0.9285**	0.7010	0.8701	0.9159	0.9274	0.9274
Avg.	0.9086	0.7433	0.8784	0.9087	**0.9090**	0.8683

(b) Recall

	Baseline	**O-SVM**	**PEBL**	**HCPU**		
				SL	**CL**	**GA**
D_1	0.5115	**0.7430**	0.5845	0.6949	0.5140	0.5246
D_2	0.5328	**0.7803**	0.6528	0.7032	0.4906	0.4924
D_3	0.4267	0.8185	0.6162	**0.8419**	0.4173	0.4193
D_4	0.3640	0.7756	0.4283	**0.8116**	0.3573	0.3568
D_5	0.5602	**0.7562**	0.5639	0.6998	0.5633	0.5583
D_6	0.3412	**0.7458**	0.5511	0.7071	0.3412	0.3367
D_7	0.4779	**0.7724**	0.6368	0.6877	0.4859	0.4859
Avg.	0.4592	**0.7703**	0.5762	0.7352	0.45288	0.4534

(c) F_1 score

	Baseline	**O-SVM**	**PEBL**	**HCPU**		
				SL	**CL**	**GA**
D_1	0.6659	0.7381	0.7175	**0.7963**	0.6680	0.6732
D_2	0.6793	0.7673	0.7614	**0.7967**	0.6436	0.6449
D_3	0.5710	0.7654	0.7146	**0.8548**	0.5625	0.5640
D_4	0.5206	0.7728	0.5767	**0.8585**	0.5144	0.5129
D_5	0.6930	0.7474	0.6776	**0.7885**	0.6966	0.6920
D_6	0.4879	0.7657	0.6703	**0.7965**	0.4876	0.4829
D_7	0.6310	0.7350	0.7354	**0.7856**	0.6377	0.6377
Avg.	0.6070	0.7560	0.6933	**0.8110**	0.6015	0.6011

The proposed method HCPU-SL resulted in the highest gain in F_1 scores from Baseline (Table 2): The macro-average F_1 score increased from 0.6070 (Baseline) to 0.8110 (HCPU-SL), because HCPU increased macro-average recall from 0.4592 to 0.7352 without decreasing macro-average precision. In contrast, HCPU-CL and HCPU-GA had no significant change compared to Baseline. Both OC-SVM and PEBL increased F_1 scores by increasing recall but they decreased precision.

5 Conclusion

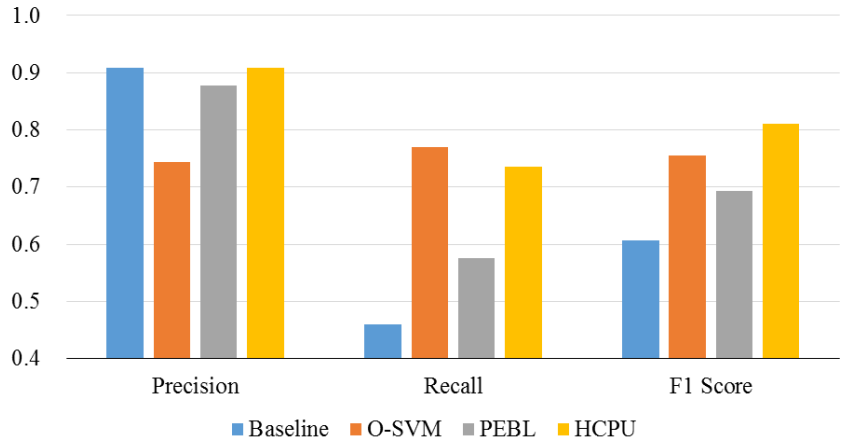

Fig. 3 Summary of MDD experiments

We improved a method of hierarchical clustering from positive and unlabeled examples to solve the MDD task. In the experimental results, the proposed method had higher F_1 score than the existing methods (Fig. 3). The proposed method reduced the number of false-negative errors and therefore achieved high recall compared to the baseline (Fig. 3). This is because the final BC was trained iteratively using identified reliable negative examples. We also verified that single linkage clustering is the most accurate linkage criterion for the MDD task. This is because the other linkage criteria identified incorrectly most unlabeled examples as negative examples.

We plan to perform research on out-of-domain (OOD) detection. MDSDS should detect OOD utterances and reject them. The problem is that detecting OOD without using actual OOD data for training is a difficult task [13]. However, we expect OOD detection problem can be solved by applying the proposed method into large-scale unlabeled examples such as conversational logs.

Acknowledgements

- This work was supported by ICT R&D program of MSIP/IITP. [14-824-09-014, Basic Software Research in Human-level Lifelong Machine Learning (Machine Learning Center)]
- This work was supported by National Research Foundation of Korean (NRF) [NRF-2014R1A2A1A01003041, Development of Multi-party Anticipatory Knowledge-Intensive Natural Language Dialog System].

References

[1] S. Ryu, D. Lee, I. Lee, S. Han, G. G. Lee, M. Kim, and K. Kim, "A Hierarchical Domain Model-Based Multi-Domain Selection Framework for Multi-Domain Dialog Systems," In *Proceedings of the 24th International Conference on Computational Linguistics*, Mumbai, India, December 2012.

[2] B. Liu, W. S. Lee, P. S. Yu, and X. Li, "Partially Supervised Classification of Text Documents," In *Proceedings of the 19th International Conference on Machine Learning*, New South Wales, Sydney, July 2002.

[3] X. Li and D. Roth, "Learning Question Classifiers," In *Proceedings of the 19th International Conference on Computational Linguistics*, Taipei, Taiwan, September 2002.

[4] H. Yu, J. Han, and K. C. Chang, "PEBL: Positive Example Based Learning for Web Page Classification using SVM," In *Proceedings of the 8th ACM SIGKDD International Conference of Knowledge Discovery and Data Mining*, Edmonton, Alberta, Canada, July 2002.

[5] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society, Series B* (*Methodological*), 39(1):1-38, 1997.

[6] A. McCallum and K. Nigam, "A comparison of event models for Naive Bayes text classification," In *Proceedings of the 15th Natural Conference on Artificial Intelligence*: *Workshop on Learning from Text Categorization*, Madison, Wisconsin, USA, July 1998.

[7] X. Li and B. Liu, "Learning to Classify Texts Using Positive and Unlabeled Data," In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, Acapulco, Mexico, August 2003.

[8] J. Rocchio, "Relevance Feedback in Information Retrieval." *The Smart Retrieval System*: *Experiments in Automatic Document Processing*, Englewood Cliffs, New Jersey, USA, 1971.

[9] B. Liu, Y. Dai, X. Li, W. S. Lee, and P. S. Yu, "Building Text Classifiers Using Positive and Unlabeled Examples," In *Proceedings of the 3rd IEEE International Conference on Data Mining*, Melbourne, Florida, USA, November 2003.

[10] T. Hastie, R. Tibshirani, J. Friedman, *The Elements of Statistical Learning* (2nd edition), New York: Springer, 520-528, 2009.

[11] C. Chang and C. Lin, "LIBSVM: A Library for Support Vector Machines," *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1-27:27, 2011.

[12] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, 13(7):1443-1471, 2001

[13] I. Lane, T. Kawahara, T. Matsui, and S. Nakamura, "Out-of-Domain Utterance Detection using Classification Confidences of Multiple Topics", *IEEE Transactions on Audio, Speech, and Language Processing*, 15(1):150-161, 2007.