Proceedings of the World Congress on Engineering 2010 Vol I
WCE 2010, June 30 - July 2, 2010, London, U.K.

Logic Relation Refinement Using Unlabeled Data*

Ki Chan†, Tak-Lam Wong‡ and Wai Lam§

Abstract— **The difference in distributions between datasets from different domains, such as different information sources, hinders the direct application of a learned model from one domain to another. We have developed a framework for the adaptation of relational logic models, in particular, Markov Logic Network (MLN), from a source domain to a target domain solving the same task using only unlabeled data in the target domain. In our proposed framework, we modify the model from two aspects, the dependency information across the two domains and within the target domain. First, the relational logic models of the two domains should share certain amount of similarities due to the same goal and similar nature of the data. Hence, we perform model adaptation by penalizing the difference in the two domains and jointly maximizing the likelihood of the target domain and minimizing the difference between the source and the target domain MLNs. Second, closely related information appeared within the target domain is used as additional clues in resolving ambiguous decision making. Potential ambiguity of the model is identified and is refined through incorporating such closely related information. As a result, the adapted model is tailored to the target domain. Our experimental results demonstrate that our adaptation framework is able to improve the performance on the target domain.**

Keywords: model adaptation, relational logic models, Markov logic networks

1 Introduction

Domain adaptation is an actively investigated task in the natural language processing community. Very often, in solving a particular task, we may have plenty of labeled data from one information source. However, we may need to process data from another information source with different distributions. The two datasets from different information sources are referred as two different domains. Domains may refer to documents from different information sources, different topical categories, or different registers in linguistics. For example, one may have labeled documents from the Wall Street Journal, but the actual goal is to develop a model for performing part-of-speech tagging for biomedical texts. Documents from the Wall Street Journal and the biomedical texts are referred as the source and the target domains respectively. Due to the difference in the distributions between the two domains, the learned model for the source domain may not be adequate for the target domain. One approach to handling such situation is to perform annotations and prepare labeled data for the target domain so that a model specific to the target domain can be induced. However, in many natural language processing tasks, limited annotated data is produced with expensive cost. Therefore, how to improve the performance in the target domain with minimal efforts is the primary goal of domain adaptation. Different from common works in domain adaptation which are given with labeled source domain data, our paper focuses on the setting where we are given a model that is already trained for the source domain. With only unlabeled data from the target domain, we adapt the source domain model to the target domain.

Though domain adaptation has been actively studied, little investigations have been carried for relational logic models. The reason is that knowledge represented in a relational logic model for a domain may be very specific to that particular domain and very often expert knowledge is used in constructing the models. Minor changes to the model may result in large degradation of performance. Adapting the model to another domain becomes even more challenging when only unlabeled target domain data is given. In this paper, we propose a framework of model adaptation for a relational logic model, specifically Markov Logic Networks (MLN) [10]. MLN is a combination of probabilistic and first-order logic graphical models. The representation of first-order logic enables flexible model construction involving relations between entities. A standard MLN consists of a set of first-order logic formulae describing the logic relations of the task and a set of weights, in which a weight is associated with each formula. Our framework performs model adaptation

*Manuscript received March 23, 2010. The work described in this paper is substantially supported by grants from the Research Grant Council of the Hong Kong Special Administrative Region, China (Project No: CUHK4128/07) and the Direct Grant of the Faculty of Engineering, CUHK (Project Codes: 2050442 and 2050476). This work is also affiliated with the Microsoft-CUHK Joint Laboratory for Human-centric Computing and Interface Technologies.

†K. Chan is with the Department of Systems Engineering & Engineering Management, The Chinese University of Hong Kong, Shatin, Hong Kong (email:kchan@se.cuhk.edu.hk).

‡T. L. Wong is with the Department of Computer Science & Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong (email:wongtl@cse.cuhk.edu.hk).

§W. Lam is with the Department of Systems Engineering & Engineering Management, The Chinese University of Hong Kong, Shatin, Hong Kong (email:wlam@se.cuhk.edu.hk).

ISBN: 978-988-17012-9-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

in a situation where an MLN has already been learned for a source domain and unlabeled data from the target domain is given.

Two major issues are considered in our framework. First, the weights of the logic formulae have to be refined to capture the difference in the distributions between the source and the target domain. Though the distributions are different, similarities exist both the source and the target domains. Therefore, we hypothesize that the distribution of the target domain may not deviate far from the source domain. Hence, by penalizing the difference in the two domains, we jointly seek to maximize the likelihood of the target domain and minimize the difference between the MLNs of the source and the target domains. Second, there must be some formulae in the source domain MLN which lead to ambiguous decisions for the target domain. By analyzing the adapted weights of the target domain, those ambiguous logic formulae are identified. We propose an approach to refining the logic formulae by analyzing dependency information in the target domain. As a result, an MLN specific to the target domain can be discovered. Our experimental results demonstrate that our adaptation framework is able to improve the performance on the target domain. Both the logic formulae and the weights are refined such that they are more suitable for the target domain.

2 Related Work

Traditional MLN structure learning methods aim at constructing logic formulae of MLN with labeled training data. Kok and Domingos [3] first introduced a probabilistic method for learning MLN structure which outperforms previous inductive logic programming (ILP) methods. More recently, Kok and Domingos [4] presented an approach which constructs candidate clauses by considering the relational database as a hypergraph. Although existing works of structure learning can refine an existing MLN structure, considerable amount of labeled data on the target domain has to be provided.

A related task known as transfer learning on MLN is actively investigated. It focuses on mapping a knowledge base from one task to another, where the predicates and variables are different. This problem setting of these transfer learning methods is different from the one we intend to solve in this paper. Mihalkova et al. [7] proposed to first map the predicates in a source MLN to the target domain and then revise the mapped MLN. The transfer learning problem was also referred as deep transfer by Davis and Domingos [1].

Domain adaptation has been widely studied for many other learning algorithms. More recently, domain adaptation models using unlabeled data from the target domain have been investigated. Some works have attempted to learn a new representation for bridging the source and the target domain [9]. Other works try to evaluate the difference in distributions between two domains by a non-parametric distance estimate. Pan et al. [8] applied the Kernel Maximum Mean Discrepancy to learn the embedded space where the distance between distributions of the source and the target domain is minimized. Guo et al. [2] developed a model using latent semantic association to overcome the distribution gap between domains. Another research direction for domain adaptation is instance weight assignment. Zhong et al. [12] seek a common feature space by utilizing the Kernel Discriminative Analysis (KDA) and then re-selects and re-weighs source domain examples to remove the bias of the mapping. However, most of these works assume that the conditional distribution of the label values given a data instance is unchanged between the source and target domains.

3 Background and Problem Definition

3.1 Background of Markov Logic Network

Markov Logic Network (MLN), proposed by Richardson et al. [10], aims at representing the knowledge in first-order logic in a probabilistic manner to handle uncertainty. It is composed of a knowledge base containing a set of first-order formulae and a set of weights, each of which is associated with a formula. Given an MLN and a set of constants, a ground Markov network is automatically obtained by applying the formulae to the set of constants, i.e. grounding of formulae. Of the Markov network constructed, a node corresponds to each grounding of the predicates specified in the formulae and each node can be equal to 0 or 1 representing the truth value of the grounding. Two nodes are connected by an edge if their corresponding ground predicates appear together in the same formula. Essentially, the probability distribution of a ground Markov network, X, can be expressed as follows:

$$P(X = x) = \frac{1}{Z} exp(\sum_{i}^{F} w_i n_i(x)) = \frac{1}{Z} \prod \phi_i(x_{\{i\}})^{n_i(x)} \quad (3.1)$$

where $P(X)$ refers to the probability distribution over all possible worlds x, the assignment of truth values; F represents the number of formulae in the MLN; w_i refers to the weight for the i-th formula; $n_i(x)$ refers to the number of true groundings of a formula in the possible world x, and $x_{\{i\}}$ is the truth value of the atoms, the groundings of of the predicates, in the formula, and $\phi_i(x_{\{i\}}) = e^{w_i}$; Z is the normalizing factor which is the sum of the probabilities over all possible worlds.

3.2 Learning MLN

A domain D_s in a text mining problem may refer to a set of documents collected from the same information source. To solve the problem, we can define two sets of predicates, namely, evidential predicates and query predicates. Evidential predicates refer to the predicates whose

ISBN: 978-988-17012-9-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

truth value can be determined from the observation of the text, while query predicates refer to the predicates whose truth value cannot be determined directly. For example, in citation segmentation, the objective is to extract the fields of a citation. We can define a query predicate called $InField(i, f, c)$ which represents if the i-th position of the citation c is part of the field f, where $f \in \{Title, Author, Venue\}$. An evidential predicate can be $HasToken(t, i, c)$ representing that the citation c has a token t in the i-th position. We can construct an MLN, denoted by $MLN_s = \langle F_s, W_s \rangle$, which consists of a set of logic formulae F_s associated with the weights W_s, for the domain D_s to solve the text mining problem. The objective is to infer the truth value of the groundings for the query predicates given the truth values of the groundings of the evidential predicates.

Given a set of training examples in the domain D_s, such that the truth values of the groundings for query predicates are known, MLN weight learning aims at automatically learning the weight W_s for each of the logic formulae in MLN_s. To achieve this, we used a more efficient alternative to Equation 3.1 as follows [10]:

$$P_{W_s}(X = x) = \prod_{l=1}^{n} P_{W_s}(X_l = x_l | MB_X(X_l)) \quad (3.2)$$

where X_l refers to the l-th grounded atom; x_l refers to the state of X_l in the training data; $MB_X(X_l)$ refers to the state of the Markov blanket of X_l; and n is the total number of grounded atoms in the training data; the subscript W_s denotes that the probability is computed using the weight W_s. The learned MLN can then be applied to the testing data in the same domain D_s.

3.3 MLN Adaptation

One major limitation of existing MLN learning is that the learned MLN_s for the *source* domain D_s cannot be effectively applied to a *target* domain D_t, unseen documents from another information source, with satisfactory performance. The objective of MLN adaptation is to reduce the human work for learning an MLN for a new target domain. MLN adaptation can be defined as follows: Given an $MLN_s = \langle F_s, W_s \rangle$ trained from a source domain D_s, and a set of unlabeled data in another target domain D_t, where unlabeled data refers to the data in which the truth values of the groundings of the query predicates are unknown, MLN adaptation aims at learning an MLN, denoted as $MLN_t = \langle F_t, W_t \rangle$, tailored to the target domain D_t.

4 Our Proposed Framework

The direct application of an existing source domain model to another target domain would lead to performance degradation even for solving the same task. The inadequate formulae and weights have to be modified to fit to the target domain. Therefore, we develop a framework to perform weight adaptation and formulae adaptation to improve the performance of the target domain model. One characteristic of our proposed framework is that it starts with an existing source domain model MLN_s. This existing model MLN_s could be designed by domain experts, or it could be automatically learned using a standard structure learning method and weight learning method if the source domain labeled data is available. Another characteristic of our framework is that we considered only unlabeled target domain data to revise the source domain model for the target domain. We aim at analyzing the differences and the similarities between the two domains using evidential predicates from the target domain.

 # Our Adaptation Framework
INPUT: $MLN_s = \langle F_s, W_s \rangle$: An MLN for source domain D_s;
Λ_t: A set of unlabeled data in target domain D_t
OUTPUT: MLN_t: An MLN for D_t
ALGORITHM:
1: $W_t \leftarrow$ Perform weight adaptation on $\langle Fs, W_s \rangle$
2: $\langle F_t, W'_t \rangle \leftarrow$ Perform Formula Refinement on $\langle F_s, W_t \rangle$
3: $W''_t \leftarrow$ Perform weight adaptation on $\langle F_t, W'_t \rangle$
4: $MLN_t = \langle F_t, W''_t \rangle$

Figure 1: An outline of our adaptation framework.

The rationale of our proposed framework is that though the source domain and the target domain are different, they are related in certain aspects and share certain similarities. Therefore, we develop a new algorithm for weight adaptation. First, similar to standard MLN learning, we attempt to obtain a set of weights which maximize the likelihood of the target domain data. Second, since both domains solve the same task, the source domain MLN_s and the target domain MLN_t should share certain similarities. Hence, by jointly maximizing the likelihood of the target domain data and modeling the extent of difference between the two domains, we can learn a new set of weights, W_t, suitable for the target domain.

The second major component in our framework is to tackle the problem of inadequate formulae. Some relations described by the source domain formulae may no longer be beneficial to the target domain even with the adapted weights as they may lead to ambiguous decisions. We aim at refining those ambiguous formulae to discover additional relational constraints for the target domain. The challenge of refining the formulae lies in that with only unlabeled data of the target domain where the truth values of the query predicates are unknown, it is not trivial to establish the relations between the evidential predicates and the query predicates. Therefore, we develop a new algorithm to identify those ambiguous formulae by analyzing the adapted weights, W_t. The rationale of our formulae refinement algorithm follows common human decision making. If the current information is ambiguous, we need to find further references to support our decision making. Similarly, we are seeking closely related

ISBN: 978-988-17012-9-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

relations to refine the ambiguous formulae.

The overview of our framework is depicted in Figure 1. First, the weights of the source domain MLN_s is revised by our weight adaptation component. Second, by analyzing the adapted weights W_t, we refine the set of source domain formulae F_s. Finally, the refined set of formulae F_t together with updated weights W'_t is adapted again to obtain the final target domain MLN_t.

4.1 Weight Adaptation

Let MLN_s be the MLN for D_s with the set of weights W_s. Given a set of M training data $\Lambda_s = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(M)}, y^{(M)})\}$ in the source domain D_s, where $x^{(i)}$ and $y^{(i)}$ refer to the truth value of the ground evidential predicates and the truth value of the ground query predicates respectively, the pseudo-log-likelihood function of the training examples can be expressed as:

$$Ł_{W_s}(\Lambda_s) = \sum_{k=1}^{M} \sum_{l} P_{W_s}(Y_l = y_l^{(k)} | MB_{x^{(k)}}(Y_l)) \quad (4.3)$$

where Y_l and $y_l^{(k)}$ refer to the l-th atom of any query predicate and the truth value of the l-th ground query predicate in the k-th training example respectively. $MB_{x^{(k)}}(Y_l)$ refers to the state of the Y_l's Markov blanket in the k-th training example. The objective of learning of MLN_s is to find the set of W_s such that the pseudo-log-likelihood function is maximized. Since in the target domain D_t, the truth value of the ground query predicate is unknown, the set of N training data in D_t can be represented by $\Lambda_t = \{(x^{(1)}), \ldots, (x^{(N)})\}$. The objective of MLN learning in the target domain D_t is to find a set of weights, namely, W_t, which is different from W_s in principle, such that the learned MLN, denoted as MLN_t can accurately predict the truth value of the ground query predicates in D_t. MLN learning in the target domain becomes nontrivial and Equation 4.3 is not adequate.

Our weight adaptation approach is designed based on two objectives. The first objective is that we aim at learning the set W_t such that MLN_t should be tailored to Λ_t. On the other hand, we observe that the source domain D_s and the target domain D_t should share certain similarity. Our second objective is to ensure that MLN_t will not deviate too far away from MLN_s. Hence, we aim at maximizing the following objective function in our MLN adaptation approach:

$$\begin{aligned} &Ł'_{W_t}(\Lambda_t, W_s) \\ &= \sum_{k=1}^{N} \sum_{l} \sum_{X_h \in MB(Y_l)} \log \{ \sum_{y'=0,1} P_{W_t}(X_h = x_h^{(k)} | MB_{x^{(k)}}(X_h), Y_l = y') \\ &\quad P_{W_t}(Y_l = y' | MB_{x^{(k)}}(Y_l))\} - \delta Q(\Lambda_t, W_s, W_t) \end{aligned} \quad (4.4)$$

where $X_h \in MB(Y_l)$ and $x_h^{(k)}$ refer to the h-th atom in $MB(Y_l)$ and the truth value of the h-th ground predicate in the k-th unlabeled data in Λ_t respectively, $Q(\Lambda_t, W_s, W_t)$ is a penalty function with respect to Λ_t, W_s, and W_t, and δ is the penalty parameter.

Recall that our first objective is to find a set of W_t such that MLN_t is tailored to Λ_t. As the truth value of the ground query predicates is unknown, this is insufficient for learning MLN_t using the pseudo-log-likelihood expressed in Equation 4.3. Instead, for each ground query predicate Y_l in Λ_t, we aim at maximizing the likelihood of the ground evidential predicates X_h connected to Y_L. The first term of Equation 4.4 refers to the expected pseudo-log-likelihood function on the ground evidential predicates in Λ_t, with respect to the $P_{W_t}(Y_l = y' | MB_{x^{(k)}}(Y_l))$. Our second objective is to prevent MLN_t from deviating too far away from MLN_s. To achieve this, we introduce a penalty function $Q(\Lambda_t, W_s, W_t)$ that is defined as follows:

$$Q(\Lambda_t, W_s, W_t) = \sum_{k=1}^{N} \sum_{l} \chi(y_l|_{W_s}, y_l|_{W_t}) \quad (4.5)$$

where $y_l|_{W_s}$ and $y_l|_{W_t}$ are the predicted truth values for the ground query predicate Y_l in Λ_t using W_s and W_t respectively, and $\chi(x, y)$ is an indicator function which is equal to 1 if $x = y$ and 0 otherwise. It is obvious that $Q(\Lambda_t, W_s, W_t)$ increases as the number of disagreements for predicting the truth value of the ground predicates using MLN_s and MLN_t increases. As a result, by adjusting the penalty parameter δ, we can reduce the disagreement on prediction using MLN_s and MLN_t, and hence prevent MLN_t from deviating too far away from MLN_s in learning.

4.2 Logic Formula Refinement

We develop an algorithm of logic formula refinement to modify the formulae in the source domain MLN_s. It aims to refine the formulae which are not informative to the target domain by discovering additional relations or constraints. Moreover, formulae usually contains constants in the predicate arguments. As an example, formulae are constructed for each token, i.e. constants, for solving tasks of natural language processing. Our algorithm is able to refine such formulae. It is achieved by first analyzing the structure of the source domain formulae and the corresponding adapted weights to identify the ambiguity of the relations described. Formulae sharing similar premises with minor weight difference between the formulae are more likely to be ambiguous to the target domain. Then, we discover additional information for the specification of those formulae in the target domain. We seek for closely related facts, i.e. the truth values of the ground evidential predicates, in the target domain unlabeled data as clues for supporting the formulae. The formulae describing those closely related facts are used as reference to initialize the weights of the refined formulae.

Figure 2 depicts the logic formula refinement algorithm. As shown in Steps 1 to 2, for each subset $\mathcal{F}_i$ of formulae sharing similar structure, we evaluate its informativeness by calculating the difference of the adapted weights across the formulae within a subset of formulae $\mathcal{F}_k$ sharing similar premises using Equation 4.6.

ISBN: 978-988-17012-9-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

 # Our Formula Refinement Algorithm
INPUT: $MLN_s = \langle F_s, W_s \rangle$: An MLN for source domain D_s;
Λ_t: A set of unlabeled data in target domain D_t; θ: A threshold
β: the maximum length of a path
OUTPUT: MLN_t: An MLN for D_t
Notation: $\mathcal{F}_i$: the subset of formulae $\{f_{i1}, ..., f_{im}\}$ having the same premise.
ALGORITHM:
1: for each $\mathcal{F}_i \subset F_s$
2: if $S(\mathcal{F}) > \theta$
3 for each $f_{ij} \in \mathcal{F}_i$
4: for each predicate e_m in the premise of f_{ij} and is not *invariant*
5: for each g_l, the ground predicate of e_m
6: for each true ground predicate g_r connected to g_l in Λ_t
7: $\mathcal{P} \leftarrow$ construct paths with g_l & g_r and have length $< \beta$
8: $\mathcal{P}' : \{p_{mn1}, ..., p_{mnr}\} \leftarrow$ variablize the paths $\mathcal{P}$
9: $e_n \leftarrow$ variablize g_r
10: Select the clause $p_{mnk} \in \mathcal{P}'$ with highest $\rho(e_m, e_n, p_{mn})$ & highest length of p_n
11: $\mathcal{J}_l \leftarrow$ find the formulae matching g_r and with same query predicate of f_{ij}
12: $P_l \leftarrow P_k \cup p'_n$
13: for each P_l
14: $f_a \leftarrow createFormula(P_l, f_{ij})$
15: $w_a \leftarrow$ average value of the weights of $\mathcal{J}_l$
16: $F_a \leftarrow F_a \cup F_a$; $W_a \leftarrow W_a \cup w_a$
17: $F'_t \leftarrow F_s \cup F_a$; $W'_t \leftarrow W_s \cup W_a$

Figure 2: Formula Refinement Algorithm

$$S(\mathcal{F}_k) = 1/n \sum_{i=1}^{n} \{ \sum_{j=i+1}^{k} |w_{ki} - w_{kj}| \} / |w_{ki}| \quad (4.6)$$

where n is the number of formulae in $\mathcal{F}_k$ and w_i is the weight of the formula $f_i \in \mathcal{F}_k$.

Definition 1 *A predicate is **invariant** if it has the same truth value for both the source and the target domains.*

Definition 2 *A ground predicate $g_k(a_1, a_2, ..., a_i, ..., a_m)$ is **connected** to another ground predicate $g_l(b_1, b_2, ..., b_j, ..., b_n)$ if $\exists a_i \; \exists b_j \; : \; (a_i = b_j)$ where g_k, g_l are m-ary and n-ary predicate respectively, and $a_1, ..., a_m, b_1, ..., b_n$ are the constants for the arguments.*

Definition 3 *A **path** p of **length** l is a series of l distinct ground predicates $p = (g_1, g_2, ..., g_k, ..., g_l), \forall g_i \in D$ where D represents the knowledge base, such that for $1 < k \leq l$:*

1. *the kth ground predicate g_k is connected to the $(k-1)$th ground predicate g_{k-1}, and*
2. *for $1 < i \leq k-1$, $g_k \neq g_i$.*

In Steps 3 and 4, for each equation $f_{ij} \in \mathcal{F}_i$, we focus on the evidential predicates which are not invariant as defined in Definition 1. For the citation segmentation task, an example of invariant predicate is $IsDate(t)$ which represent whether a token t is a term for describing date. The atom $IsDate(\text{“}December\text{”})$ is always true for both the source and the target domains. Pairs of connected ground predicates are identified. Paths, which contains the pair of connected ground predicates and have length shorter than or equal to the maximum length value specified, are discovered. A path is a series of connected ground predicates as given in Definitions 2 and 3. For example, the path, $E(a,b) \wedge F(a,c)$, where the two ground predicates, $E(a,b)$ and $F(a,c)$, are connected with the constant, a, is of length 2. These candidate paths are then variablized where some constants are replaced with variables to form candidate clause $\mathcal{P}'$ in Step 8. Using the above example, considering the path $E(a,b) \wedge F(a,c)$, since the constants b and c are not the focus of interest, they are replaced by variables ν_1 and ν_2 to form the clause $E(a,\nu_1) \wedge F(a,\nu_2)$. Next, the relatedness measure $\rho(e_m, e_n, p_{mnk})$ is calculated for each candidate clause p_{mnk} by Equation 4.7. If the closely related ground predicate is contained in the grounding of the source domain formulae, it is added to the matching formulae subset $\mathcal{J}_r$ in Step 11. The longest candidate clause with maximum $\rho(e_m, e_n, p_{mnk})$ is used to construct new formulae. Finally, in Steps 13 to 15, new formulae are constructed by the formula f_{ij}, set of clauses $p'_n \in \mathcal{P}'$, where the corresponding weight is initialized by the average weights of the matching formulae. These formulae are added to the rest of the formulae to obtain the target domain MLN_t.

The relatedness measure for a candidate clause p_{mn} is defined as:

$$\rho(e_m, e_n, p_{mn}) = \frac{N_p(p_{mn})[R(p_{mn}) - R(e_m)R(e_n)]}{\sqrt{[R(e_m) - R(e_m)^2][R(e_m) - R(e_m)^2]}} \quad (4.7)$$

where $R(p_{mn})$, $R(e_m)$, and $R(e_n)$ denote the ratio of the number of true groundings over the number of groundings for the clause p_{mn}, predicates e_m and e_n, in the target domain unlabeled data λ_t respectively. p_{mn} represents the candidate clause containing the predicates e_m and e_n. $N_p(x, D)$ and $N_t(x, D)$ denote the number of true groundings for formula or clause x. and the number of grounding for formula or clause x given the dataset D respectively. $R(x)$ is defined as:

$$R(x) = N_p(x, \lambda_t) \; / \; N_t(x, \lambda_t) \quad (4.8)$$

5 Experiments

Our framework is implemented based on the Alchemy system [5], which provides algorithms in statistical relational learning for the MLN. As a baseline for evaluation, the source domain MLN is learned using the standard weight learning approach in the Alchemy system. The source domain MLN is then directly applied on the target domain for testing without adaptation.

We have conducted experiments on the task of segmentation of citation records. The goal of segmentation of citation records is to extract the candidate fields, namely, title, author, and venue, from the citation strings. We employed the segmentation MLN model developed by Singla and Domingos [11] for our experiments. The corresponding query is $InField(i, f, c)$, which is true if and only if the i-th position of the citation c is part of the field f, where $f \in \{Title, Author, Venue\}$. The main evidential predicate is $HasToken(t, i, c)$, which is true if and only if citation c has a token t in the i-th position where t is a the token in the source domain labeled data. Moreover, there are predicates describing the strings, the information on

ISBN: 978-988-17012-9-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Table 1: Experimental Results in F_1 measure for Segmentation of Citation Records

Method	Target Domains											
	constraint satisfaction				automated reasoning				reinforcement learning			
	Total	Author	Title	Venue	Total	Author	Title	Venue	Total	Author	Title	Venue
Baseline Model	76.4%	73.9%	69.0%	81.4%	76.2%	74.7%	67.1%	81.0%	77.3%	75.8%	72.2%	80.7%
Our Proposed Model	78.8%	78.4%	71.0%	83.5%	79.0%	81.2%	68.8%	83.1%	78.9%	81.8%	71.9%	81.8%

punctuation, and positional information regarding the citations. For example, the following formula, describing whether a word t_j is part of the field f_n, is created for every word in the source domain data.

$$HasToken(t_j, i, c) \Rightarrow InField(i, f_n, c)$$

Experimental Setup

We conducted experiments on CiteSeer [6], one of the standard datasets used for information extraction on citations. The CiteSeer dataset has approximately 1,500 citations, and it contains four different topic sections, namely, constraint satisfaction, face recognition, automated reasoning, and reinforcement learning. In the experiments, the face recognition section of the CiteSeer dataset is used as the source domain. Each of the three other sections is used as target domain separately. The penalty parameter δ was set to a value of 1 for weight adaptation. The threshold for formula refinement is set to 0.5. The F-measure metric is used for evaluating the citation segmentation performance. Specifically, the F_1 measure is the equally weighted harmonic mean of Precision and Recall.

Experimental Results

Table 1 shows our adaptation performance for the segmentation task. The performance of our proposed model demonstrates consistent improvement obtained by our framework for different target domains. Our model can successfully captures the difference between the source and the target domain. Weights representing the relative importance of the formulae are revised and the ambiguous formulae are modified for the target domains. An example of a refined formula is:

$$HasToken(knowledge, i, c) \wedge HasToken(between, j, c) \wedge Next(j, i) \Rightarrow InField(i, f, c)$$

where $Next(j, i)$ represents that the token at position j is next to the token at position i. Since, the token "knowledge" appears frequently in both the field of "title" and "venue", our algorithm has refined the formula by a closely related token "between". The token "between" appears mostly in the field of "title". Hence, its appearance together with the token "knowledge" is a good reference for deciding the field.

6 Conclusions

In this paper, we have presented a framework that can adapt an existing MLN to a target domain solving the same task using unlabeled data from the target domain. Our proposed weight adaptation method revises the significance of the existing logic formulae for the target domain using the unlabeled target domain data only. Our formula refinement algorithm discovers useful relational patterns and refines existing ambiguous formulae. The new formulae constructed are useful in modeling the underlying relations in the target domain. The consistent improvements in the experimental results of our framework demonstrate that both the logic formulae refined and the adapted weights can better characterize the target domain.

References

[1] J. Davis and P. Domingos. Deep transfer via second-order markov logic. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*, 2008.

[2] H. L. Guo, H. Zhu, Z. Guo, X. Zhang, X. Wu, and Z. Su. Domain adaptation with latent semantic association for named entity recognition. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2009.

[3] S. Kok and P. Domingos. Learning the structure of markov logic networks. In *Proceedings of the Twenty-Second International Conference on Machine Learning*, 2005.

[4] S. Kok and P. Domingos. Learning markov logic network structure via hypergraph lifting. In *Proceedings of the Twenty-Sixth International Conference on Machine Learning, Canada*, 2009.

[5] S. Kok, P. Singla, M. Richardson, M. Sumner, and H. Poon. The alchemy system for statistical relational ai. In *http://alchemy.cs.washington.edu/*, 2006.

[6] S. Lawrence, K. Bollacker, and C. L. Giles. Autonomous citation matching. In *Proceedings of International Conference on Autonomous Agents*, 1999.

[7] L. Mihalkova and R. J. Mooney. Transfer learning by mapping with minimal target data. In *Proceedings of the AAAI-08 Workshop on Transfer Learning for Complex Tasks*, 2008.

[8] S. Pan, J. Kwok, and Q. Yang. Transfer learning via dimensionality reduction. In *Proceedings of the Twenty-Third AAAI conference on Artification Intelligence*, 2008.

[9] R. Raina, A. Battle, H. Lee, B. Packer, and A. Ng. Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the 24th Annual International Conference on Machine Learning*, 2007.

[10] M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62:107–136, 2006.

[11] P. Singla and P. Domingos. Memory-efficient inference in relational domains. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, 2006.

[12] E. Zhong, W. Fan, J. Peng, and K. Zhang. Cross domain distribution adaptation via kernal mapping. In *Proceedings of 15th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2009.

ISBN: 978-988-17012-9-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)