Cross-Domain Semi-Supervised Learning Using Feature Formulation

Xingquan Zhu, *Member, IEEE*

***Abstract*—Semi-Supervised Learning (SSL) traditionally makes use of unlabeled samples[1] by including them into the training set through an automated labeling process. Such a primitive Semi-Supervised Learning (pSSL) approach suffers from a number of disadvantages including false labeling and incapable of utilizing out-of-domain samples. In this paper, we propose a formative Semi-Supervised Learning (fSSL) framework which explores hidden features between labeled and unlabeled samples to achieve semi-supervised learning. fSSL regards that both labeled and unlabeled samples are generated from some hidden concepts with labeling information partially observable for some samples. The key of the fSSL is to recover the hidden concepts, and take them as new features to link labeled and unlabeled samples for semi-supervised learning. Because unlabeled samples are only used to generate new features, but not to be explicitly included in the training set like pSSL does, fSSL overcomes the inherent disadvantages of the traditional pSSL methods, especially for samples not within the same domain as the labeled instances. Experimental results and comparisons demonstrate that fSSL significantly outperforms pSSL-based methods for both within-domain and cross-domain semi-supervised learning.**

***Index Terms*—Cross domain learning, machine learning, semi-supervised learning, transfer learning.**

I. Introduction

SEMI-SUPERVISED Learning (SSL) [1], [2] represents a class of machine learning techniques that utilize unlabeled samples to boost the learning on a labeled set (referred to as the "target set" in this paper). Intuitively, as labeling training samples is often subject to a significant amount of human labor or costs, whereas cheap unlabeled samples can be easily collected with trivial efforts, it makes sense to utilize unlabeled samples in the learning process to boost the model performance. Traditionally, SSL problems are solved through an iterative labeling and learning process, where some automated "labeling agents" trained from labeled samples are used to generate class labels for unlabeled samples, with aggregated training set (containing both genuinely and automatically labeled samples) being used to further improve the labeling agents. The iterative labeling and learning process normally repeats a number of times until some stopping criteria are satisfied. In some situations, all unlabeled samples are included into the training set with a pseudo-class label assigned to each unlabeled sample. The final models trained from the aggregated training set are used for predictions.

Many methods [1], [2] exist for semi-supervised learning by using mechanisms, such as expectation maximization (EM) principles, graph-based label propagation [6], or orthogonal neighborhood-preserving projection [22], to determine proper class labels for unlabeled samples. All these methods, in a narrow sense, share a striking similarity in their design: including unlabeled samples into the training set by assigning a class label to each of them. As a result, unlabeled samples can be directly integrated into the training process in the original feature space. In this paper, we call this approach "primitive semi-supervised learning" (pSSL) mainly because unlabeled samples are included into the training set in a primitive instance form (i.e., original feature space). Alternatively, if an unlabeled samples is linked to the training process through some transformed feature space, we call such approaches formative semi-supervised learning (fSSL). Under this definition, most existing algorithms, such as Co-Training [5], [16], common component [28], ASSEMBLE [30], SemiBoost [29], and Transductive SVM [40] all belong to pSSL because they explore the connection between labeled and unlabeled samples in the original feature space.

Traditional pSSL approaches, in practice, suffer from at least two disadvantages. First, since the class labels of the unlabeled samples are automatically determined without verification, the labeling process may introduce a certain amount of mislabeled samples into the training set. Secondly, pSSL intuitively assumes that all unlabeled samples are from the same domain as the target set. So the class labels of the target set are able to represent unlabeled samples. This assumption, in practice, is too strong for applications involving samples from relevant, but non-identical, domains. For example, assume the learning objective of a labeled training set is to differentiate images of three types of animals including mammal, bird, and reptile (i.e., a three-class classification problem), it is possible that some unlabeled images may only contain natural scenery, fish, or amphibian, so neither of the existing class labels (mammal, bird, and reptile) can be used to properly label them. As a result, the labels of the target set are insufficient for labeling samples in the auxiliary set.[2] Including an unlabeled sample, in its primitive form, into the training set, like pSSL does, is inappropriate for this type of applications, because any class

Manuscript received August 26, 2010; revised January 6, 2011 and April 20, 2011; accepted May 5, 2011. Date of publication June 27, 2011; date of current version November 18, 2011. This work was supported by the Australian Research Council (ARC) Future Fellowship under Grant No. FT100100971. This paper was recommended by Associate Editor S.-F. Su.

The author is with the Faculty of Engineering and Information Technology, University of Technology, Sydney, NSW 2007, Australia (e-mail: xqzhu@it.uts.edu.au).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCB.2011.2157999

[1] In this paper, sample and instance are interchangeable terms.

[2] In this paper, a target set denotes a labeled sample set whereas an auxiliary set denotes an unlabeled set.

label assigned to the unlabeled image would be incorrect and therefore introduce error/noise [20] to the training data.

The inherent disadvantages of the existing pSSL approaches motivate the proposed fSSL design. fSSL considers that both labeled and unlabeled samples are generated from some hidden concepts with labeling information partially observed (i.e., labeling information is available for partial samples). The key of the fSSL is to discover the hidden concepts, and use them as new features to link unlabeled samples to the target set to support semi-supervised learning. Because this approach formulates new features to capture correlations between labeled and unlabeled samples, we name it formative SSL. The main technical challenge of fSSL is summarized as follows.

1) **Data Models:** Because unlabeled samples (possibly from different domains of the target set) may enrich the target set, proper data models are needed to combine labeled and unlabeled samples for learning.
2) **Conceptual Correlations:** As the volumes of the labeled and unlabeled samples can be arbitrarily large, proper designs are needed to summarize instance-level correlations and generate high-level conceptual correlations, which can be used as the key to link labeled and unlabeled samples.
3) **Feature Formulation:** As including unlabeled samples in their primitive forms into the training set (like pSSL does) is not an option for us, new features capable of describing unlabeled samples are needed to boost the learning on the target set.

In this paper, we propose an fSSL framework to address the above challenges. fSSL uses an affinity matrix as the data model to combine labeled and unlabeled samples for analysis (Challenge 1). Probabilistic latent semantic [3] is further applied to the affinity matrix to derive hidden concepts (conceptual correlations) behind instance-level correlations (Challenge 2). At the final stage, the posterior probabilities of the hidden concepts with respect to each labeled sample are used as new features to integrate unlabeled samples into the target set for learning (Challenge 3). The inherent advantage of fSSL, in comparison with pSSL, is twofold.

- **False Labeling Immunization:** fSSL uses unlabeled samples to formulate new features for semi-supervised learning. As a result, there is no risk of including mislabeled samples into the training set. Although new features may also contain erroneous values (or some non-informative values from the learning task perspective), since attribute errors are essentially less harmful than class label errors [20], fSSL is much less vulnerable to data errors and has a much lower risk of performance loss than pSSL.
- **Cross-Domain Learning:** fSSL does not need to assign a class label for each unlabeled sample (whereas most pSSL methods do). As a result, even if unlabeled samples are from different domains of the labeled samples, fSSL is still able to integrate them into the training process to boost the learning on the target set.

The remainder of the paper is structured as follows. Section II briefly reviews related work in the area. The data model and hidden feature formulation are introduced in Section III, followed by Section IV which presents the technical details of the fSSL framework. Experimental results are reported in Section V, and we conclude the paper in Section VI.

II. Related Work

By considering labeled and unlabeled samples in within-domain or cross-domain learning environments, the proposed research is closely related to three research areas, including: 1) semi-supervised learning; 2) transfer learning; and 3) self-taught learning.

Semi-Supervised Learning: Because labeling is a labor intensive and possibly costly process [37], whereas unlabeled samples are readily available for many applications, semi-supervised learning [1], [2] has been popularly used in many applications, including text classification [33] and content-based video/image retrieval [22], [23], [35]. To utilize unlabeled samples for semi-supervised learning, common practices are to: 1) explicitly label unlabeled samples and include them into the training set to train a classification model; or 2) use unlabeled samples to help tune parameters for generative models. For example, Co-Training [5] is one of the most classical semi-supervised learning methods which builds classifiers from different views of the labeled samples to label unlabeled instances according to the predictions from trained classifiers. Blum and Chawla [6] proposed to build graph structures based on the pairwise relationships between labeled and unlabeled samples, and employed graph mincuts to partition the graph in a way that minimizes the number of similar pairs of samples sharing different labels. Similar graph-based semi-supervised learning method has also been investigated [32] through a generative model which uses graph propagation to estimate conditional probabilities and uses linear regression to estimate class priori probabilities. Tang *et al.* [27] proposed a graph-based semi-supervised learning model which considers correlations between labeling concepts to improve annotation performance for video databases (whereas most existing approaches regard labeling concepts to be independent of each other). In many model-based parametric learning systems, semi-supervised learning is used to help estimate the parameters of the underlying model. For example, Dong and Bhanu [23] proposed a user directed semi-supervised expectation-maximization algorithm for mixture parameter estimation, and they have demonstrated that semi-supervised model parameter estimation can handle two basic aspects of a content-based image retrieval system: the changing (image insertion or removal) nature of a database and user queries. Transductive SVM [40] is another type of method in this category, which includes unlabeled samples into the objective function in searching for an optimal solution for classification. While methods in this category do not require an explicit assignment of the class labels for unlabeled samples, the parameter searching process does the labeling in an implicit way such that the estimated parameters are supposed to generate the "best" models. For traditional semi-supervised learning, one assumption commonly made is that both labeled and unlabeled samples share the same (or similar) distributions, whereas in many cases we may encounter situations where labeled and unlabeled samples are collected from different domains. For

example, in document classification, it is possible that labeled samples are collected from domains such as "artificial intelligence," whereas unlabeled samples are collected from different (but relevant) domains such as "computational biology." On the other hand, in image retrieval systems, it is also possible that labeled samples belong to two categories such as "building" versus "scenery," whereas unlabeled samples are actually collected from relevant categories, such as "animal," which belongs to neither categories of the labeled images. In fact, the above issues have been partially observed in a number of semi-supervised learning works, where the main focus is to regard the cross-domain learning as a sample selection bias problem. For example, Chawla and Karakoulas [7] empirically studied the selection bias problem for semi-supervised learning and provided empirical results to answer questions such as the effect of the size of the labeled and unlabeled sets, the data imbalance and noise issues. Similarly, Attenberg and Provost [25] recently studied the effect of data imbalance for label acquisition and suggested alternative solutions for applying human resources to build classification models under extreme class imbalance. In summary, although a large number of works exist for utilizing unlabeled samples to support semi-supervised learning, to the best of our knowledge, none of the existing SSL methods offers solutions to explicitly handle situations where labeled and unlabeled samples are from different domains.

Transfer Learning [9], [10], [13], [21]: Addresses the problem of using knowledge (or information) gained from one domain (namely the auxiliary domain) to improve the learning on a target domain on which labeled training samples are expensive or difficult to acquire. Common solutions for transfer learning are to adjust sample weights [9] or to employ feature selection to adjust sample distributions or feature values so data from auxiliary domain can be used to assist the learning for the target domain. Other approaches use locally weighted ensemble framework [21] to combine multiple models for transfer learning, with the weight values dynamically adjusted according to each local model's prediction power on the test data. Zhu *et al.* [36] uses sample weighting to solve transfer learning in an incremental learning setting. In addition to the sample weighting or feature transferring, other cross domain learning methods build consensus regularization model [24] to train classifiers, by considering local data in one domain and the prediction consensus with classifiers trained from other domains. Multi-task learning [26] addresses the problem of learning a task together with other related tasks simultaneously, where all tasks normally share the same domain information. For traditional transfer learning, cross-domain learning, or multi-task learning, samples from both auxiliary and target domains are assumed to be suitably labeled and are subject to different sample distributions or feature representations. In comparison, fSSL enables the integration of unlabeled samples from different domains to advance the transfer learning. It can essentially solve the transfer learning problem by extracting hidden concepts between target and auxiliary sets as new features.

Self-Taught Learning [11], [15]: Represents a special type of learning task whose purpose is to transfer knowledge from unlabeled samples (presumably from different domains of the target set) through unsupervised feature construction. In other words, self-taught learning does not have the assumption that unlabeled data follows the same class labels or generative distribution as the labeled data (which is a general assumption that most SSL methods employ). One prerequisite of the self-taught learning is that both labeled and unlabeled samples should be decomposed into some *base vectors* [12], to represent high-level features of the data. For example, in self-taught learning [15], the sparse coding [12] was used to decompose each image patch (i.e., a small region of the image) into a sparse weighted combination of a set of base vectors (which are shared across different patches). This is similar to Fourier transformation or Wavelet transformation where input signal is decomposed into some weighted combinations of the base functions (e.g., sine or cosine functions for Fourier transformation). Although such base vectors are available for image data, there is, unfortunately, no base vector available for generic datasets. Due to such limitations, the solutions proposed in [15] are only applicable for image data, but cannot be applied to generic data, which normally do not have shared coding bases. In comparison, the proposed fSSL framework can easily handle any type of data for self-taught learning.

III. Data Model and Hidden Feature Formulation

Assume $L = \{l_1, l_2, \ldots, l_{|L|}\}$ and $U = \{u_1, u_2, \ldots, u_{|U|}\}$ each denotes a set of $|L|$ labeled and $|U|$ unlabeled instances, where each labeled instance $l_i = \{x_i, y_i\}$ contains a set of feature values x_i and a class label y_i, and each unlabeled instance $u_j = \{x_j\}$ only contains feature values but has no class label. In this paper, we also refer to L and U as a "target set" and an "auxiliary set," respectively. The **aim** of semi-supervised learning is to build prediction models from labeled samples in the target set L, by leveraging information from auxiliary set U, to predict samples in a test set, which are assumed to be sampled from the same domain as the target set L. Different from generic semi-supervised learning which assumes that target and auxiliary sets have the same domain, in our problem definition, we do not restrict target set L and auxiliary set U to have the same domain. If L and U are from different domains, we refer to the problem as cross-domain semi-supervised learning.

To explicitly characterize the relationships between labeled and unlabeled samples, we consider that both labeled (L) and unlabeled samples (U) are generated from a generative model showing in Fig. 1. More specifically, the generative model assumes that behind the observable (labeled and unlabeled) samples, some hidden concepts $z_1, z_2, \ldots, z_k$ exist to determine the whole sample generation process, with instance generation following a three-step procedure,

- Pick a hidden concept z_k with probability $P(z_k)$.
- Generate a labeled instance l_i, given concept z_k, with probability $P(l_i|z_k)$.
- Generate an unlabeled instance u_j, given concept z_k, with probability $P(u_j|z_k)$.

As the final outcome of the above instance generation process, one obtains observable instance pairs $(l_i, u_j), i = 1, \ldots, |L|,\ j = 1, \ldots, |U|$ whereas hidden concept variables $z_k, k = 1, \ldots, K$ are discarded. Notice that the generation of the instances l_i and u_j is conditionally independent, given a

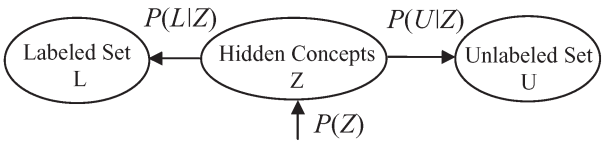

Fig. 1. Graphical generative model for labeled and unlabeled samples.

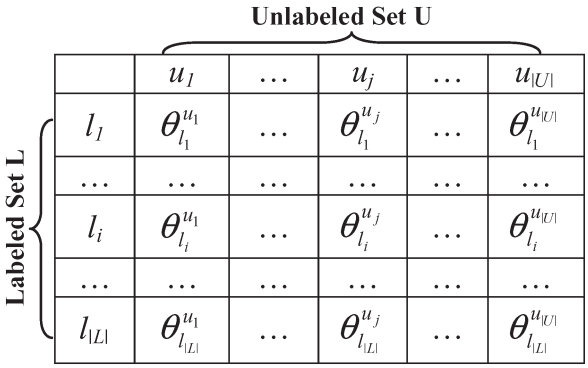

Fig. 2. Affinity matrix recording instance-level correlations between labeled (row) and unlabeled (column) samples.

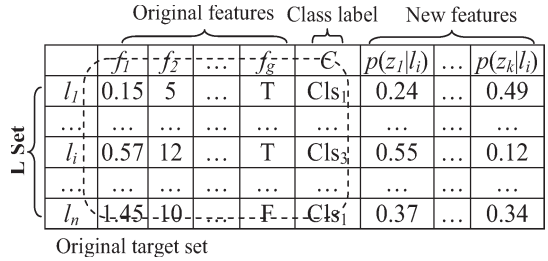

Fig. 3. Conceptual view of using hidden features to link unlabeled samples to labeled instance for semi-supervised learning.

hidden concept z_k. It is easy to derive conditional independence formula as shown in

$$P(l_i, u_j|z_k) = P(l_i|z_k)P(u_j|z_k). \tag{1}$$

To explicitly capture the above data generation process, we employ an instance–instance affinity matrix, as shown in Fig. 2, to denote instance-level correlations between observed, including labeled and unlabeled, samples. In Fig. 2, each row l_i and each column u_j denotes a labeled and an unlabeled sample, respectively, and the value $\theta_{l_i}^{u_j}$ represents a measurable correlation, such as Euclidean distance, between samples in the corresponding column and row.

Assume a suitable method exists to discover hidden concepts behind labeled and unlabeled samples, the posterior probability of a concept z_k with respect to each labeled sample l_i, $p(z_k|l_i)$, can can formulate a new feature for l_i. Consequently, one can transfer the original target set into the one shown in Fig. 3, where a set of new features are added to the target set to bridge the connection between each labeled instance l_i and all unlabeled samples U for learning. As a result, the key issue remaining is the calculation of posterior probabilities $p(z_k|l_i), k = 1, \ldots, K$ for each labeled sample l_i. This problem can be formulated as the probabilistic latent semantic [3] extraction, which has been popularly used in natural language processing.

A. Probabilistic Latent Sematic for Hidden Concept Extraction

Assume the existence of an affinity matrix $\Theta \in \mathbb{R}^{|L|\times|U|}$, the joint probability of observing a labeled/unlabeled instance pair (l_i, u_j) is given by (2), where K denotes the number of hidden hidden concepts z_k underneath the data

$$P(l_i, u_j) = P(l_i)P(u_j|l_i) = P(l_i)\sum_{k=1}^{K} P(u_j|z_k)P(z_k|l_i). \tag{2}$$

Applying Bayes' rule to the conditional probability $P(z_k|l_i)$, we have

$$P(z_k|l_i) = \frac{P(l_i|z_k)P(z_k)}{P(l_i)}. \tag{3}$$

Accordingly, one can formulate a new form for (2) as

$$P(l_i, u_j) = \sum_{k=1}^{K} P(z_k)P(l_i|z_k)P(u_j|z_k). \tag{4}$$

In fact, according to the conditional independence given in (1), (4) can be interpreted as the marginal probability of the joint probability $P(l_i, u_j, z_k)$ over all concept $z_k, k = 1, \ldots, K$ as given in

$$P(l_i, u_j) = \sum_{k=1}^{K} P(z_k)P(l_i, u_j|z_k) = \sum_{k=1}^{K} P(l_i, u_j, z_k). \tag{5}$$

Assume the instance–instance affinity matrix $\Theta = \theta_{l_i}^{u_j}$, $i = 1, \ldots, |L|$; $j = 1, \ldots, |U|$ denotes the normalized number of times that a labeled instance l_i is observed with an unlabeled instance u_j (i.e., instance pairs with a higher correlation value have a better chance of being observed simultaneously), the total log likelihood over all instances pairs is denoted by

$$\mathcal{L} = \log P(L, U, Z) = \sum_{k=1}^{K}\sum_{i=1}^{|L|}\sum_{j=1}^{|U|} \theta_{l_i}^{u_j} \log P(l_i, u_j, z_k). \tag{6}$$

According to the Maximum Likelihood principle, one can determine posterior probabilities, such as $P(z_k|l_i)$, by maximizing the log-likelihood function given in (6). To solve the problem, one can separate major variables in (6) into two groups: 1) hidden variable Z; and 2) observable variables L and U, and employ the EM process below to iteratively update the variables to approach to the maximum.

- **E-Step:** Using observable variables L and U to estimate the hidden variable as $P(Z|L, U)$.
- **M-Step:** Using $P(Z|L, U)$ to calculate conditional probabilities $P(L|Z)$ and $P(U|Z)$, such that the expected total log-likelihood defined in (6) can be maximized.

Following the above EM principle and using normalization constraint that $\sum_{k=1}^{K} P(z_k|l_i, u_j) = 1$, rearrange (6) using conditional independence given in (1) we have

$$\mathcal{L} = \log[P(Z, L, U)] = \log[P(Z|L, U)P(L, U)] = \sum_{k=1}^{K}\sum_{i=1}^{|L|}\sum_{j=1}^{|U|}\theta_{l_i}^{u_j}\log[P(z_k|l_i, u_j)P(l_i, u_j)]. \quad (7)$$

Using joint probability $P(l_i, u_j)$ as given in (4), we have

$$\mathcal{L} = \log[P(Z, L, U)] = \sum_{k=1}^{K}\sum_{i=1}^{|L|}\sum_{j=1}^{|U|}\theta_{l_i}^{u_j}\{\log[P(l_I|z_k)P(u_j|z_k)] + \log[P(z_k)]\}P(z_k|l_i, u_j). \quad (8)$$

From (8), one can devise an iterative EM procedure between: 1) $P(z_k|l_i, u_j)$; and 2) $P(z_k)$, $P(l_i|z_k)$, and $P(u_j|z_k)$, to maximize (8). According to Bayes' rule, the conditional probability $P(z_k|l_i, u_j)$ is given in

$$P(z_k|l_i, u_j) = \frac{P(l_i, u_j|z_k)P(z_k)}{\sum_{t=1}^{K} P(l_i, u_j|z_t)P(z_t)}. \quad (9)$$

Rearrange (9) using (1), we have

$$P(z_k|l_i, u_j) = \frac{P(l_i|z_k)P(u_j|z_k)P(z_k)}{\sum_{t=1}^{K} P(l_i|z_t)P(u_j|z_t)P(z_t)}. \quad (10)$$

Given conditional probabilities $P(z_k|l_i, u_j)$, one can determine $P(z_k)$, $P(l_i|z_k)$, by maximizing (8) using Lagrange multipliers with respect to the three constraints given in (11), as shown in (12) where λ_k, μ_i, and ν_j are defined Lagrange multipliers

$$\sum_{z=1}^{K} P(z_k) = 1; \sum_{i=1}^{|L|} P(l_i|z_k) = 1; \sum_{j=1}^{|U|} P(u_j|z_k) = 1 \quad (11)$$

$$\Lambda(\mathcal{L}, \lambda, \mu, \nu) = \mathcal{L} + \sum_{k=1}^{K}\lambda_k(1 - P(z_k)) + \sum_{i=1}^{|L|}\mu_i(1 - P(l_i|z_k)) + \sum_{j=1}^{|U|}\nu_j(1 - P(u_j|z_k)) \quad (12)$$

The critical values of (12) occur when its gradient is zero. One can calculate the partial derivatives of (12) with respect to $P(z_k)$, $P(l_i|z_k)$, and $P(u_j|z_k)$, respectively, and set them equal to zero. The results will lead to solutions for $P(z_k)$, $P(l_i|z_k)$, and $P(u_j|z_k)$ given as follows:

$$P(z_k) = \frac{\sum_{i=1}^{|L|}\sum_{j=1}^{|U|}\theta_{l_i}^{u_j}P(z_k|l_i, u_j)}{\sum_{i=1}^{|L|}\sum_{j=1}^{|U|}\theta_{l_i}^{u_j}} \quad (13)$$

$$P(l_i|z_k) = \frac{\sum_{j=1}^{|U|}\theta_{l_i}^{u_j}P(z_k|l_i, u_j)}{\sum_{i=1}^{|L|}\sum_{j=1}^{|U|}\theta_{l_i}^{u_j}P(z_k|l_i, u_j)} \quad (14)$$

$$P(u_j|z_k) = \frac{\sum_{i=1}^{|L|}\theta_{l_i}^{u_j}P(z_k|l_i, u_j)}{\sum_{i=1}^{|L|}\sum_{j=1}^{|U|}\theta_{l_i}^{u_j}P(z_k|l_i, u_j)}. \quad (15)$$

B. Hidden Feature Formulation

Following probability functions given in (10) and (13)–(15), an EM process can be carried out to repetitively update (10) and (13)–(15) for a number of iterations or until the algorithm converges. After that, one can apply Bayes' rule to the conditional probabilities $P(l_i|z_k), i = 1, \ldots, |L|$, $k = 1, \ldots, K$ and determine the posterior probability of z_k with respect to each labeled sample l_i as shown in

$$P(z_k|l_i) = \frac{P(l_i|z_k)P(z_k)}{\sum_{t=1}^{K} P(l_i|z_t)P(z_t)}. \quad (16)$$

The posterior probabilities $P(z_k|l_i), k = 1, \ldots, K$ thus form K-dimensional hidden features for each labeled instance l_i.

IV. Formative Semi-Supervised Learning

Following the data model and the hidden feature formulation, Algorithm 1 lists the main procedure of the proposed fSSL learning framework. Given a labeled target set L and an unlabeled auxiliary set U, fSSL first calculates the instance–instance affinity matrix $\Theta \in \mathbb{R}^{|L|\times|U|}$ by using measures such as Euclidean distance or cosine distance. In addition, because hidden concept extraction only requires the affinity matrix Θ, fSSL can also support semi-supervised learning for heterogeneous sample sets (i.e., labeled and unlabeled samples have different feature representations) as long as one can properly determine their (dis)similarities.

After the calculation of the Θ, another important step is to determine the number of hidden concepts K. According to the generative model in Fig. 1, the number of hidden concepts is closely related to the number of categories of the labeled and unlabeled samples. Assume instances in L and U are *i.i.d* with their labels randomly discarded during the sample generation process, it is sufficient to set K to any number greater than the total number of classes of the target set. On the other hand, since fSSL eventually uses the posterior probabilities of each hidden concept z_k, $P(z_k|l_i)$, $k = 1, \ldots, K$, as new features for l_i, it is necessary to limit the number of hidden concepts and control the total number of new feature dimensions (because learning from high-dimensional data is an identified machine learning challenge). By taking the above two factors into consideration, we empirically set hidden concept number K equal to two times the number of classes of the target set. In Section V-B, we will report the performance of fSSL with respect to different number of hidden concepts and justify the selection of the K value for generic datasets.

After the determination of the hidden concept number K, the solutions proposed in Section III are employed to extract hidden concepts and calculate posterior probabilities, as given in (16), for each labeled sample l_i. A new training set L' is

formed, through which a classifier $\hbar(L')$ is trained to classify a test instance x.

Algorithm 1 fSSL: formative Semi-Supervised Learning

Require: (1) A labeled set $L = \{l_1, \ldots, l_i, \ldots, l_{|L|}\}$ (i.e., target set) where $l_i = \{x_i, y_i\}$; and
(2) An unlabeled set $U = \{u_1, \ldots, u_j, \ldots, u_{|U|}\}$ (i.e., auxiliary set) where $u_j = \{x_j\}$.
$\Theta \in \mathbb{R}^{|L|\times|U|} \leftarrow$ Instance–Instance Affinity Matrix(L, U)
$ClassNum \leftarrow$ Target-Set-Class-Number(L)
$K \leftarrow 2 \times ClassNum$
$P(l_i|z_1), \ldots, P(l_i|z_K) \leftarrow$ Hidden-Concept-Extraction (Θ, K) // EM process between (10) and (13)–(15).
for each labeled instance $l_i \in L$ **do**
  for each hidden concept $z_k k = 1, \ldots, K$ **do**
    $P(z_k|l_i) \leftarrow$ Instance-Concept Posterior Probability.// (16)
  end for
end for
$F \leftarrow P(z_1|l_i), \ldots, P(z_K|l_i)$ // Hidden feature formulation
$L' \leftarrow$ Form new training set using L and F
$\hbar(L') \leftarrow$ train classifier from L'
========== TEST PHASE ==========
for each test instance x **do**
  $\Delta_x \leftarrow$ find x's nearest neighborhood in L
  $F'_x \leftarrow$ estimate hidden features using (17)
  $x' \leftarrow x \cup F'_x$: form new instance
  $\hbar(L', x')$: classify
end for
return prediction accuracy

A. Test Phase

Notice that although classifier $\hbar(L')$ is trained from the transferred target set L' with newly formulated features, a test sample x, however, does not have corresponding feature values. This is because a test sample was not included in the affinity matrix Θ, the posterior probabilities $P(z_k|x), k = 1, \ldots, K$ of x are therefore not immediately available. On the other hand, including each test instance x into the affinity matrix as $\Theta' \leftarrow \Theta \cup x$ and solving Θ' for each test instance x are heavily time consuming, because it involves an iterative EM process (we obviously cannot afford to repeat the whole process for each single test instance). Alternatively, we can use a less expensive procedure to estimate $P(z_k|x)$ by using x's nearest neighborhood Δ_x in L.

According to the Bayes' Rule, the posterior probability $P(z_k|x)$ is the product between the priori probability $P(z_k)$ and the conditional probability (or likelihood) $P(x|z_k)$, divided by the priori probability $P(x)$. Because $P(x)$ is a constant for any concepts z_k, $P(z_k|x), k = 1, \ldots, K$ are only determined by $P(x|z_k)P(z_k)$, as shown in the graphical model in Fig. 4(a). Because $P(x|z_k)$ denotes the likelihood of instance x given a hidden concept z_k, we can use x's nearest neighborhood Δ_x in the original target set L to estimate $P(x|z_k)$, as shown in Fig. 4(b). Consequently, assume x's nearest neighborhood Δ_x contains $|\Delta_x|$ instances, the calculation of $P(z_k|x)$ can be estimated by using the average of all samples in Δ_x as $\sum_{i,l_i\in\Delta_x}^{|\Delta_x|}(P(l_i|z_k)P(l_i|x)P(z_k)/|\Delta_x|)$. As a result, the posterior probability $P(z_k|x), k = 1, \ldots, K$ is given by (17), where

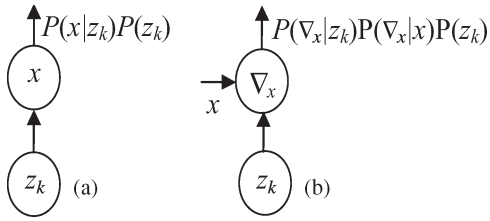

Fig. 4. Graphical models for estimating posterior probability $P(z_k|x)$. (a) Calculate $P(z_k|x)$ using conditional probability $P(x|z_k)$, which is unavailable. (b) Estimate $P(z_k|x)$ using x's nearest neighborhood Δ_x in L, whose conditional probabilities $P(\Delta_x|z_k)$ are available.

$P(x|l_i)$ denotes the likelihood of the test instance x given a labeled sample l_i, which can be replaced by the correlation or Euclidean distance between x and l_i

$$P(z_k|x) = \frac{P(x|z_k)P(z_k)}{\sum_t^K P(x|z_t)P(z_t)} = \frac{\sum_{i,l_i\in\Delta_x}^{|\Delta_x|} P(l_i|z_k)P(x|l_i)P(z_k)}{|\Delta_x|\sum_t^K \sum_{i,l_i\in\Delta_x}^{|\Delta_x|} P(l_i|z_t)P(x|l_i)P(z_t)} \quad (17)$$

After the calculation of the posterior probabilities $P(z_k|x), k = 1, \ldots, K$, x is transferred into a new form x' fully compatible with the classifier $\hbar(L')$ for prediction.

V. Experimental Results

A. Experimental Settings

We implement the fSSL framework using Java platform and WEKA machine learning tool [18]. The source code of the whole framework, including benchmark datasets, can be downloaded from our web site[3]. Since fSSL is independent of the learning algorithms, we collect and report results using four learning algorithms, including C4.5 [14], Naive Bayes, SVM [8], and Transductive SVM [40]. All experimental results reported in the paper are based on 5 times 10-fold cross-validation.

Benchmark Data: For evaluation purposes, we use 11 benchmark datasets from UCI machine learning repository [4], and a high-dimensional gene expression dataset as our test bed. The domain information of the 12 benchmark datasets, including data characteristics, are listed in Table I. To generate labeled L and unlabeled U subsets for within-domain or cross-domain learning, we randomly split each dataset into two subsets, with the size of the first (L) and the second (U) subsets being $\alpha \times 100\%$ and $(1-\alpha) \times 100\%$ of the size of the original dataset, respectively. For *within-domain learning*, L and U are randomly split across all class labels, whereas for cross-*domain learning*, L and U are split based on different sets of class labels, with the test set having the same domain as the labeled set L. The similar approach has also been popularly used in assessing the algorithm performance for transfer learning across different domains [9] (Detailed procedures on cross-domain benchmark data generation are elaborated in Section V-B). For unlabeled set U, we simply discard class labels of all instances in U, and treat them as unlabeled samples (in other words, we know genuine class labels of each unlabeled sample,

[3] fSSL source code: http://www-staff.it.uts.edu.au/ xqzhu/fssl/index.html

TABLE I
BENCHMARK DATA SETS AND DOMAIN INFORMATION

Data set & Domain Information	Shorthand	Sizes	Dimensions	Classes
Car: Car evaluation data set classifying a car into four categories using 6 features including buying price, number of doors etc.	Car	1728	6	4
Credit: Credit screening for credit card applications. The purpose is to distinguish credit-worthy applications from non credit-worthy customers using 15 features.	Credit	690	15	2
Digits: Handwritten digit number (0-9) recognition using stretched images, each representing a 16x16 square box (the 256 attributes)	Digits	3689	256	10
Krvskp: Chess data set classifying King-Rook *vs*. King-Pawn (*i.e.*, White-can-win *vs*. White-cannot-win) using 36 categorical attributes describing the board.	Krvskp	3196	36	2
Letter-7: Identifying a black-and-white displaying rectangular region as one of the 7 English capital letters, including A, B, E, F, H, K, R	Letter	5329	16	7
Lymphoma: Gene expression data classifying 96 tissue samples into Germinal Cell Like (GCL) *vs*. Angiocentric lymphoma (ACL), using information from 4026 genes.	Lymph	96	4026	2
Monks-1: Representing the basis of a first international comparison of learning algorithms, the target concepts are: Class =2 if (a1 = a2) or (a5 = 1); otherwise Class=1	Monks	432	6	2
Segment: Classifying each 3x3 image region into seven categories, including sky, grass, etc., using 19 image features such as intensity-mean, color hue-mean etc.	Segment	2310	19	7
Tictactoe: Board configurations at the end of tic-tac-toe games. The target concept is Class= true if "x" has one of 8 possible ways to create a "three-in-a-row".	TTT	958	9	2
Vowel: Multidimensional time series data classifying nine male speakers' pronunciation for vowels, including hid, hId, hEd, hAd, hYd, had, hOd, hod, hUd, hud, and hed.	Vowel	990	13	11
Wine: Chemical analysis of wines grown in the same region in Italy but derived from three different cultivars (class labels). Attributes include Malic acid, Color intensity etc.	Wine	178	13	3
Zoo: Classification of animals (into 7 categories) using boolean attributes such as hair, feathers, eggs, milk etc., and a numeric attribute which accounts for number of legs.	Zoo	101	16	7

and can restore the labeling information of the sample, if necessary, to assess the algorithm performance).

Benchmark Methods and Settings: For comparison purposes, we implement two benchmark semi-supervised learning algorithms including Co-Training [5] and Transductive SVM [40] (we use SVM^{light}[39] with default parameter settings, except the kernel types, for Transductive SVM). Because Transductive SVM is based on support vector machines which cannot accommodate any other learning algorithms, in following sections, we first report results using Co-Training-based pSSL for generic learning algorithms (including including C4.5 [14], Naive Bayes, and generic SVM [8]). In the last section of the experiments, we comparatively study fSSL and Transductive SVM for both within- and cross-domain semi-supervised learning. For Co-Training-based pSSL method, we exactly follow the design in [5] and randomly partition the original features into to two non-overlapping views with nearly the same number of features for each view. We train one classifier from each view, and determine the label of an unlabeled sample according to its predicted class labels from the two classifiers. In our implementation, unless specified otherwise, Co-Training labels 15% of unlabeled samples in 50 iterations and includes labeled samples into the training set. Our results in Fig. 7 will show that due to the inherent false labeling risk, labeling more samples does not necessarily provides better results for Co-Training, and most of times, we actually observe performance loss if more samples are included into the training set.

In addition to the fSSL and pSSL methods, we also report the results of two baseline approaches, ***lower-bound*** (denoted by **L-Bound**) and ***upper-bound*** (denoted by **U-Bound**). For lower-bound, a classifier is trained from the labeled set L only, and for upper-bound, a classifier is trained from the aggregation of both labeled L and unlabeled U sets, by assigning genuine class label back to each unlabeled samples (recall that we know genuine class labels of all unlabeled samples). Clearly, lower-bound provides a baseline to indicate the minimum accuracy a classifier can achieve, whereas upper-bound provides an ideal situation to demonstrate pSSL's optimal performance under assumption that all unlabeled samples are perfectly labeled.

In our experiments, the instance–instance affinity matrix Θ is calculated by using Euclidean distance as the similarity measure. The maximum number of EM repetitions is set to be 1000, with a possible early termination if differences between two consecutive rounds are sufficiently small. The detailed parameter settings for fSSL is studied in the next subsection.

B. Parameter Settings for FSSL

Key parameters of the proposed fSSL framework include: 1) K: the number of hidden concepts; and 2) $|\Delta_x|$: the size of the nearest neighborhood used to estimate the posterior probabilities of each test instance x, as defined in (17). In Algorithm 1, we empirically set the number of hidden concepts as twice the number of classes of a dataset, whereas an important issue is to study the selection of these parameters and investigate the sensitivity of the algorithm performance with respect to different parameters, such that users can determine suitable parameter values for new datasets.

In Fig. 5(a)–(c), we report the performance (i.e., the prediction accuracies) of the fSSL with respect to different number of hidden concepts K and different sizes of nearest neighborhood, where each figure shows results for one dataset and all three figures represent results for datasets with two, four, and 11 classes, respectively.

Overall, the results in Fig. 5(a)–(c) suggest the following three major observations: 1) The system performance is moderately sensitive to the number of hidden concepts Z, and is relatively stable to the sizes of nearest neighborhood (except for Vowel dataset). For example, for Car dataset as shown in Fig. 5(b), the best algorithm performance (88.62%) is achieved

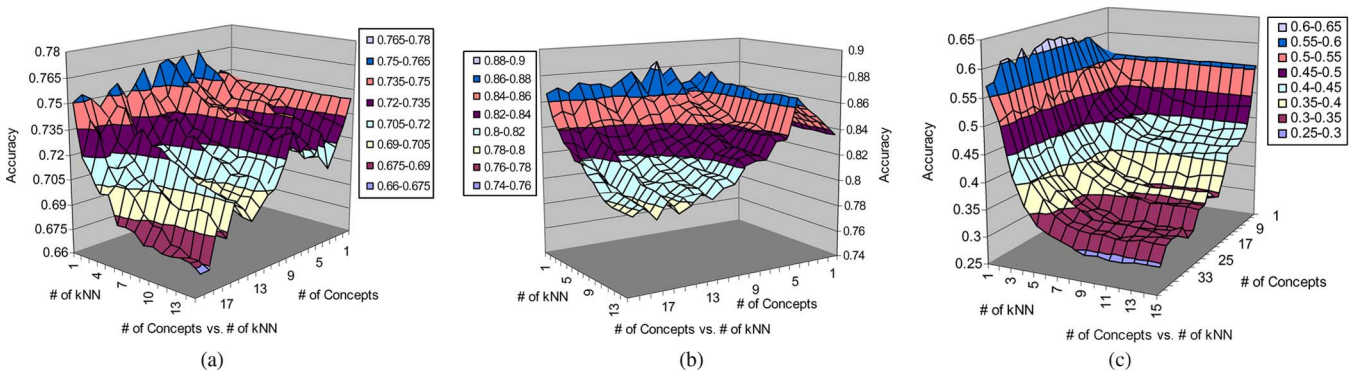

Fig. 5. Prediction accuracies (showing in the z-axis) with respect to the numbers of hidden concepts $(z_k, k = 1, \ldots, K)$ and the sizes of the nearest neighborhood (using C4.5 as the learning algorithm). The number of concepts K varies from 1 to 20 for TTT and Car datasets, and from 1 to 40 for Vowel dataset. The size of nearest neighborhood varies from 1 to 15 for all datasets. (a) TTT: 2 classes. (b) Car: 4 classes. (c). Vowel: 11 classes.

TABLE II
WITHIN-DOMAIN SSL RESULTS. A † INDICATES A t-TEST SIGNIFICANT, COMPARED TO $pSSL$, AT THE 0.05 LEVEL. THE SIZES OF THE LABELED AND UNLABELED SETS ARE 0.25% × 100% AND 0.75 × 100% OF THE ORIGINAL DATAS ET. pSSL (CO-TRAINING) LABELS 0.15% × 100% OF UNLABELED SAMPLES IN 50 ROUNDS. THE ACCURACY OF THE METHOD (AMONG LOWER-BOUND, pSSL, AND fSSL) WITH THE HIGHEST VALUE IS BOLDFACED

Learner	Method	Car	Credit	Digits	Krvskp	Letter	Lymph	Monks	Segment	TTT	Vowel	Wine	Zoo
C4.5	L-Bound	83.46	83.16	78.80	**98.01**	82.71	72.22	76.05	92.03	73.17	54.24	83.29	78.77
	pSSL	76.62	84.06	**79.81**	97.40	85.65	71.11	75.23	93.16	71.01	55.00	82.41	**81.78**
	fSSL	**87.89**†	**85.65**	79.44	97.23	**89.28**†	**74.56**†	**89.65**†	**93.98**	**75.49**†	**60.31**†	**90.08**†	80.27
	U-Bound	92.72	85.27	81.75	99.37	91.54	78.44	97.23	97.01	85.10	81.71	93.33	92.64
NB	L-Bound	**80.13**	78.04	83.46	86.38	71.06	77.33	69.31	78.22	69.48	50.24	93.22	84.72
	pSSL	74.84	78.98	82.33	80.63	72.38	74.0	70.24	80.41	69.73	47.06	**94.51**	85.79
	fSSL	79.92†	**84.13**†	**84.78**	**91.27**†	**75.56**	77.33	**74.54**†	**86.49**†	**70.39**	**53.78**†	93.81	**88.72**
	U-Bound	85.47	77.68	85.59	88.09	74.78	81.78	75.01	81.89	69.02	62.12	97.63	93.09
SVM	L-Bound	86.39	**53.91**	89.67	92.22	93.24	75.33	77.15	30.30	73.01	45.56	35.94	49.68
	pSSL	80.46	53.33	**91.03**	91.36	94.73	75.33	**79.44**	31.90	73.59	47.52	34.56	52.18
	fSSL	**90.33**†	53.84	90.48	**92.62**	**94.97**	75.33	78.95	**38.83**†	**76.04**†	**47.89**	**37.32**†	**52.47**
	U-Bound	95.50	56.88	95.38	94.08	97.74	88.22	91.67	62.12	87.29	86.87	45.94	89.09

when the number of concepts is 10 and the number of nearest neighbors is 2; 2) As the size of the nearest neighborhood grows, a general trend is that the algorithm performance will deteriorate, especially for the setting with a large number of concepts; and 3) Assume the size of the nearest neighborhood is suitably selected for a dataset, the best algorithm performance is achieved when the number of hidden concepts is about twice the number of classes. Indeed, while fSSL directly uses hidden concepts to formulate new features, for a small number of hidden concepts (say one or two concepts), the new features do not provide much information to differentiate samples from different classes. In an extreme case, if only one hidden concept is specified, this feature along provides almost no useful information at all, because all samples are assigned to the same concept. In Fig. 5 the results show that the accuracies remain the same for one hidden concept regardless of the size of the nearest neighborhood, which asserts that the new feature, for one hidden concept, does not provide much useful information for learning. On the other hand, when the number of hidden concepts is relatively large, the new features become very sparse which makes them unsuitable for differentiating samples different classes. In an extreme case, assume the number of hidden concepts is the same as the number of labeled samples, each labeled sample is most likely going to be assigned to one individual concept. Such sparse features, in practice, provide no value for learning at all, not to mention that the increase feature dimensions may actually bring negative impact to the learning algorithm.

In summary, the above experimental results suggest that for generic datasets, one can select the number of concepts to be one or twice the number of classes. Meanwhile, one can select a relatively small size of nearest neighborhood for posterior probability estimation for test instances. For experiments in the remaining sections, the size of the nearest neighborhood is set as one, because $1 - k$NN has been proven to have its asymptotic error rate at most twice of the Bayes error [34]. In addition, the number of hidden concepts is set as twice the number of classes for each dataset.

C. Within-Domain Learning Results Comparison

To compare the performance of fSSL and pSSL for semi-supervised learning on samples within the same domain, we randomly split labeled and unlabeled sets with different size ratios, across all classes. The mean accuracies over 5 time 10-fold cross validations are reported in Table II and Fig. 6. The former shows detailed accuracy comparisons of 0.25:0.75 splitting for labeled and unlabeled sets (i.e., L and U sets are 0.25 × 100% and 0.75 × 100% of the original dataset, respectively) and the latter reports the accuracies with respect to different sizes of L sets on the Vowel and Car datasets.

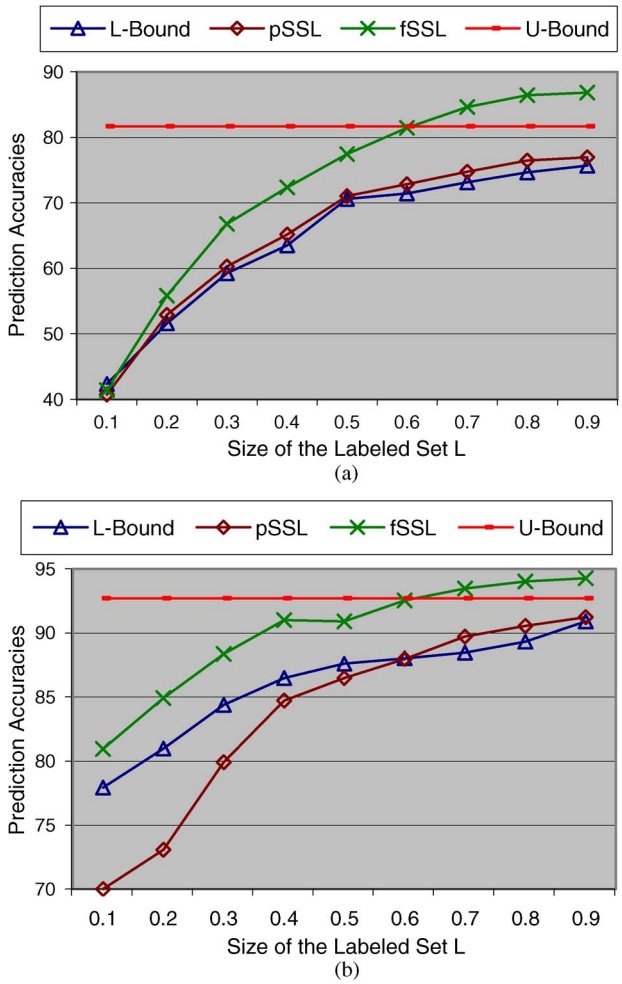

Fig. 6. Prediction accuracies with respect to different sizes (α) of labeled set L. The x-axis denotes the size of the labeled set α, and the y-axis shows the prediction accuracies (using C4.5 as the learning algorithm). (a) Vowel: 11 classes. (b) Car: 4 classes.

The results in Table II and Fig. 6 show that comparing to pSSL and lower-bound, fSSL can receive significant performance gain on most datasets, where the accuracy gain can be as much as 10% or higher. Such improvement can be observed across all three types of learning algorithms and different sizes of labeled sets. For high-dimensional data (e.g., Lymph dataset with 4026 dimensions) when using C4.5 as the learning algorithm, fSSL can still outperform pSSL and L-Bound, whereas for NB and SVM, the performance of fSSL remains the same as the L-Bound. This observation asserts that the size of the original feature dimension is not a barrier for fSSL to claim benefits for high-dimensional data. Despite of the existing high-dimensional features, new features formulated by using fSSL can still provide useful information to help improve the learner performance, especially for unstable learners, such as decision tree methods like C4.5.

Interestingly, by including new features into the target set, we frequently observe that fSSL can even outperform the upper-bound which denotes the best accuracy a learner can achieve on the whole dataset (assuming all unlabeled samples are perfectly labeled). The moral seems to be that new features provide some "fresh" knowledge the original feature space incapable

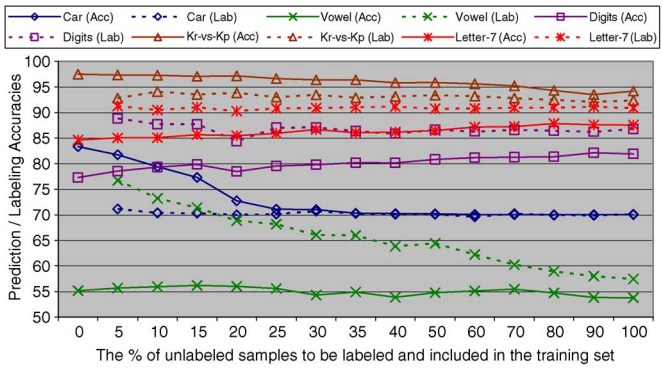

Fig. 7. Prediction (ACC) and labeling (Lab) accuracies of the Co-Training with respect to the percentage of samples to be labeled and included in the training set. The x-axis denotes the labeling percentage and the y-axis shows the prediction (solid lines) and labeling (dashed lines) accuracies. Each dataset corresponding to two lines with the same color and same style ($\alpha = 0.25$ and using C4.5).

of conveying, even if all samples are included into the training set. We believe that the reason behind is twofold,

- Although training instance number plays an important role for a learner to derive correct decision concepts, the separability of the underlying data may be improved if some new features are included to transfer data into some higher (compared to the original) dimensional spaces, through which the same learner can achieve a better accuracy. Just like kernel machines [38] which transfer data into high-dimensional space and claim better separability for learning, fSSL also transfers data into a higher-dimensional space with new features containing additional knowledge gained from unlabeled samples.
- As many datasets contain redundant instances, increasing training set size, although potentially helpful, may deteriorate the learner performance. This can be easily verified by the results in Table II, where models trained from random subsets even outperform the upper-bound (e.g., The Credit dataset with NB). The separability of the new features, in practice, can be more powerful than simply adding new training samples, not to mention that pSSL may actually include mislabeled samples.

For most datasets we observed, the accuracies of pSSL are only marginally better than the lower-bound. This suggests that pSSL has very limited capability of utilizing unlabeled samples. The main disadvantage of the traditional pSSL methods, as we discussed in Section I, stems from the mislabeling risks and mishandling of the out-of-domain samples. Notice that class labels play vital roles for learners to derive correct decision concepts, including mislabeled samples into the training set may severely deteriorate the learner performance [20]. To validate the hypothesis, in Fig. 7 we report the the prediction and labeling accuracies of Co-Training with respect to different percentages of samples to be labeled and included in the training set. As we can see, pSSL's average labeling accuracy is between 50% to 90%, which implies that a large portion of unlabeled samples are falsely labeled and included in the training set as noise. As a result, we can observe significant accuracy drop for datasets, such as Car, Kr-vs-Kp, and Vowel, when instances are continuously labeled and included in the training set.

TABLE III
CROSS-DOMAIN SSL RESULTS. A † INDICATES A t-TEST SIGNIFICANT, COMPARED TO $pSSL$, AT THE 0.05 LEVEL. pSSL (CO-TRAINING) LABELS 0.15% × 100% OF UNLABELED SAMPLES IN 50 ROUNDS. THE CLASSES USED TO SPLITTING TARGET VERSUS AUXILIARY SETS ARE **CAR:** {ACC, GOOD, VGOOD VERSUSUNACC}, **DIGITS:** {0, 2, 4, 6, 8 VERSUS 1, 3, 5, 7, 9}, **LETTER:** {A, E, H, R VERSUS B, F, K}, **SEGMENT:** {BRICKFACE, FOLIAGE, WINDOW VERSUSSKY, CEMENT, PATH, GRASS}, AND **VOWEL:** {hid, hEd, hAd, hYd, hOd, hUd VERSUS hId, had, hod, hud, hed}

Learner	Method	Car	Digits	Letter	Segment	Vowel
C4.5	L-Bound	92.27	92.09	94.62	94.79	87.07
	pSSL	92.13	90.42	92.41	93.65	87.36
	fSSL	**93.42**	**93.78**	**95.52**	**96.68**	**93.83**†
NB	L-Bound	80.32	91.88	87.19	62.43	80.85
	pSSL	83.60	89.44	86.80	58.57	80.42
	fSSL	**85.24**	**94.49**†	**89.23**	**67.62**†	**85.17**†
SVM	L-Bound	93.74	**98.91**	89.52	87.23	83.61
	pSSL	94.29	94.77	87.24	85.59	82.82
	fSSL	**94.98**	97.97	**91.33**	**89.83**	**86.47**†

Although fSSL may also introduce incorrect values to the hidden features, errors in the attributes are generally much less harmful than class label errors [20]. In the worst scenario, assume the newly formulated features contain nothing but random values, the impact of the random features can be easily reduced by most learning algorithms (such as C4.5) because many of them have an internal procedure to select informative attributes for learning.

D. Cross-Domain Learning Results Comparison

In the second set of experiments, we compare algorithm performance for cross-domain semi-supervised learning. To simulate cross-domain learning tasks, we separate class labels of the original dataset into two subsets, Set_A and Set_B, with each of them treated as the classes of the target set and the auxiliary set, respectively. For each instance in the original set, if its class label belongs to Set_A we will assign it to the target set, otherwise, it is assigned to the auxiliary set with its class label discarded.

In Table III, we report the results on five selected datasets (since a target set must contain at least two classes and an auxiliary set should have at least one class, the cross-domain learning can only be carried out on datasets with at least three classes). Overall, it is observed that fSSL is effective for cross-domain semi-supervised learning on majority datasets.

Comparing to the within-domain learning, we observe that the performance gain between each method and the baseline (lower-bound) actually decreases. This suggests that instances from one domain has relatively limited value for another domain, assume two domains are relatively isolated. Indeed, imagining a target task in distinguishing "Tiger" from "Bird," it might be helpful to provide some unlabeled images with grass, forest (background information for tiger images) and ocean, and sky (background information for bird images). Such auxiliary information can help determine that a "Tiger" is more closely related to one type of auxiliary images than others, and vice versa. On the other hand, providing unlabeled samples, such as mathematical formula or handwriting pictures, or the extreme case some pure black/white images, adds very little information to help distinguish "Tiger" and "Bird," mainly because they are essentially not related to the underlying learning tasks. Determining to which degree that one domain can be beneficial to another domain [17] or when unlabeled data can be helpful for supervised learning [19], [31] is vitally important for cross-domain semi-supervised learning. Research on this issue is, however, beyond the coverage of this paper.

In summary, the results reported in this section conclude that fSSL provides an effective platform to utilize unlabeled samples for semi-supervised learning. Significant performance gains can be observed, across different learning algorithms, for within-domain or cross-domain semi-supervised learning.

E. Comparisons Between FSSL and Transductive SVM

While the above results have demonstrated the performance of fSSL for both within-domain and cross-domain semi-supervised learning, the comparisons are made with Co-Training, which does not necessarily represent the state of the art semi-supervised learning algorithms. In this subsection, we compare the performance of fSSL with Transductive SVM (denoted by TrSVM) in the setting that transductive support vector machines [40] are used as the underlying learning algorithm.

Transductive SVM can be viewed as standardized SVM with an additional regularization term to account for unlabeled training data [2]. In short, assume a training set contains $|L|$ labeled examples $\{(x_i, y_i)\}_{i=1}^{|L|}, y_i \in \{1, -1\}$ and $|U|$ unlabeled samples $\{(x_i)\}_{i=|L|+1}^{|L|+|U|}$. The decision function of generic SVM has the following form:

$$f_\theta(x) = w \cdot \Phi(x) + b \tag{18}$$

where $\theta = (w, b)$ are the parameters of the learning models, and $\Phi(\cdot)$ denotes the kernel function in the feature space. The objective of the Transductive SVM is to solve optimization problem as defined by

$$\min \frac{1}{2}\|w\|^2 + \lambda_1 \sum_{i}^{|L|} Loss\left(y_i \times f_\theta(x_i)\right) + \lambda_2 \sum_{i=|L|+1}^{|L|+|U|} Loss\left(|f_\theta(x_i)|\right) \tag{19}$$

where $Loss(x) = \max(0, 1 - x)$ defines the classical hinge loss function for labeled samples, and $Loss(|x|) = \max(0, 1 - |x|)$ is the symmetric hinge loss function for unlabeled sample. λ_1 and λ_2 are parameters specified by the users. Different from Co-Training-based SSL methods which directly assign class labels to unlabeled samples (with potential mislabeling risks), Transductive SVM, on the other hand, directly includes unlabeled samples into the objective function without requiring class labels for them [as defined in the third term of (19)].

In our experiments, we first apply TrSVM to the training set L and train a classifier from labeled samples, with the accuracy of this classifier denoted by "TrSVM_L." In the second method, we apply TrSVM to both labeled and unlabeled training samples, with the accuracy of the classifier denoted by TrSVM_{L+U}. In the third method, we use fSSL to formulate new features, and apply TrSVM to the transferred training samples (i.e., TrSVM is used as the learning algorithm) with the accuracy of the classifier denoted by "TrSVM_{fSSL}." In the four method, we restore class labels of all unlabeled samples (recall that the class

TABLE IV
EXPERIMENTAL COMPARISONS BETWEEN fSSL AND TRANSDUCTIVE SVM FOR WITHIN-DOMAIN SEMI-SUPERVISED LEARNING. TrSVM$_L$, TrSVM$_{L+U}$, AND TrSVM$_{fSSL}$, EACH DENOTES THE ACCURACYOF APPLYING TRANSDUCTIVE SVM LEARNING ALGORITHM TO LABELED SAMPLES (L), LABELED PLUS UNLABELED (L + U) SAMPLES, AND TRANSFERRED LABELED SAMPLES USING fSSL, RESPECTIVELY. A † INDICATES A t-TEST SIGNIFICANT, COMPARED TO TrSVM$_{L+U}$, AT THE 0.05 LEVEL. BECAUSE TRANSDUCTIVE SVM CAN ONLY HANDLE BINARY CLASSIFICATION PROBLEMS, WE ONLY REPORT RESULTS ON FIVE BINARY CLASSIFICATION DATAS ETS (LINEAR AND POLYNOMIAL DENOTE LINEAR KERNEL AND POLYNOMIAL KERNEL, RESPECTIVELY)

Kernel	Method	Credit	Krvskp	Lymph	Monks	TTT
Linear	TrSVM$_L$	64.59	91.37	83.11	66.36	65.31
	TrSVM$_{L+U}$	**69.22**	90.58	80.46	67.44	60.35
	TrSVM$_{fSSL}$	64.59	**91.71**	**84.15**†	**71.90**†	**67.37**†
	TrSVM$_{All}$	65.32	93.83	92.45	68.52	71.44
Polynomial	TrSVM$_L$	55.07	93.21	73.24	70.13	71.82
	TrSVM$_{L+U}$	**57.74**	91.08	73.55	71.04	68.82
	TrSVM$_{fSSL}$	55.38	**94.64**†	**76.22**	**74.32**†	**74.52**†
	TrSVM$_{All}$	57.18	94.89	77.33	82.66	81.19

label of each instance was discarded to generate an unlabeled instance, so we know the genuine class label of each unlabeled sample) and aggregated all samples to form a training set. We apply TrSVM to the aggregated training set, with the accuracy of the classifier dented by "TrSVM$_{All}$," which simulates the best accuracy a TrSVM can possibly achieve, if all unlabeled samples are suitably labeled.

In Table IV, we report within-domain semi-supervised learning results on five binary classification problems (please note Transductive SVM, as implemented in SVM^{light}[39], can only handle binary classification problems). In all experiments, the size of the labeled and unlabeled sets are 25% and 75% of the original dataset. To observe algorithm performance with respect to different types of kernel functions, we employ two types of kernels, linear kernel and polynomial kernel, in our experiments. When employing TrSVM to unlabeled samples only, linear kernel outperforms polynomial kernel for 2 out of 5 datasets. This observation asserts that the accuracies of SVM classifiers are sensitive to the type of kernels selected in the objective function. A carefully selected kernel can improve the performance gain for as much as 10%. Meanwhile, the effectiveness of fSSL can be observed for both linear and polynomial kernels, and the results are consistent to the within-domain learning results as we reported in Table II.

Because Transductive SVM directly includes unlabeled samples into the objective function without explicitly assigning a class label to each unlabeled sample, we expect that a TrSVM classifier trained from labeled and unlabeled samples (i.e., TrSVM$_{L+U}$) should outperform a TrSVM classifier trained from labeled samples only (i.e., TrSVM$_L$). In practice, this is not always the case. Although we do observe TrSVM$_{L+U}$ to have significant performance gain from one dataset (Credit), its improvement on the rest of the datasets is rather marginal. In fact, when comparing TrSVM$_L$ versus TrSVM$_{L+U}$ one can observe that TrSVM$_{L+U}$ appears to be highly unstable and may result in severe performance loss for some datasets, whereas for TrSVM$_{fSSL}$ there is normally no noticeable performance loss, even if its results is not better than TrSVM$_L$. In three out of five datasets, we observed TrSVM$_{fSSL}$ to be statistically significantly better than TrSVM$_{L+U}$.

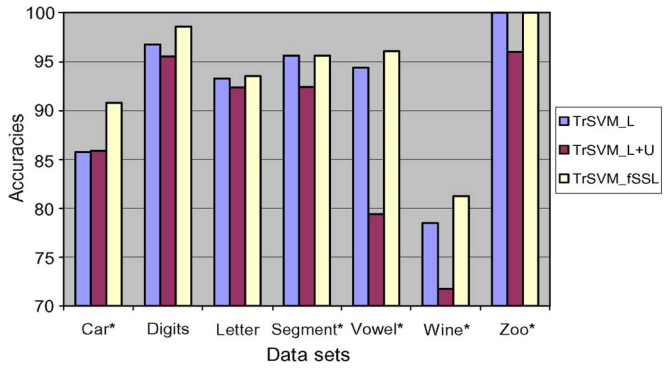

Fig. 8. Experimental comparisons between fSSL and Transductive SVM for cross-domain semi-supervised learning. TrSVM$_L$, TrSVM$_{L+U}$, TrSVM$_{fSSL}$, each denotes accuracy of applying Transductive SVM learning algorithm to labeled samples (L), labeled plus unlabeled (L + U) samples, and transferred labeled samples using fSSL, respectively. A * indicates a t-test significant, comparing TrSVM$_{fSSL}$ to TrSVM$_{L+U}$, at the 0.05 level. Because Transductive SVM can only handle binary classification problems, classes used to splitting target and auxiliary sets are **Car**: {good, vgood versus unacc, acc}, **Digits**: {0, 1 versus 2, 3, 4, 5, 6, 7, 8, 9}, **Letter**: {B, R versus A, E, F, H, K}, **Segment**: {bickface, cement versus sky, foliage, window, path, grass}, **Vowel**: {hid, hId versus hEd, hAd, hYd, had, hOd, hod, hUd, hud, hed}, **Wine**: {1, 2 versus 0}, and **Zoo**: {bird, invertebrate versus mammal, reptile, fish, amphibian, insect}. *For comparison purposes, the actual accuracies of Voweland Wine* data *sets are the accuracies showing in the figure minus 20.*

For cross-domain semi-supervised learning, as shown in Fig. 8, TrSVM$_{fSSL}$ is a clear winner, where the performance of the Transductive SVM can deteriorate significantly when unlabeled samples are directly included into the learning process (like TrSVM$_{L+U}$ does). Indeed, while TrSVM$_{L+U}$ does not directly assign class labels to unlabeled instances, the third term defined in (19), which accounts for the loss incurred by unlabeled samples, actually employs an implicit labeling process to calculate the loss of each unlabeled samples. In other words, the loss value as defined by $\max(0, 1 - |w \cdot \Phi(x_i) + b|$ for each unlabeled instance $x_i, i = |L| + 1, \ldots, |L| + |U|$, implicitly labels each unlabeled samples through the arithmetical $\max(0, \cdot)$, depending on the loss of each unlabeled sample with respect to the current model $\theta = (w, b)$. So the optimization process is to find the best model θ with minimum objective value as defined by (19). As a result, TrSVM$_{L+U}$, in our definition, still falls into the primitive SSL category, and would inherently bear the same disadvantages as pSSL. For some datasets, such as Vowel as shown in Fig. 8, the performance loss of the TrSVM$_{L+U}$ can be quite dramatic. By using new feature formulation, fSSL provides effective way to support Transductive SVM for cross-domain semi-supervised learning, where the performance of fSSL (denoted by TrSVM$_{fSSL}$) mostly outperforms Transductive SVM classifiers trained from labeled samples, even for data with very high dimensionality.

VI. CONCLUSION

In this paper, we proposed a semi-supervised learning framework which uses feature formulation to combine labeled and unlabeled samples for learning. We argued that the major disadvantage of the traditional semi-supervised learning is twofold, including: 1) false labeling of the unlabeled samples may introduce a significant amount of noise into the training set; and 2) cross-domain samples cannot be properly labeled for

learning. Alternatively, we seek to formulate hidden features, using labeled and unlabeled samples, to boost the learning on the target set. To achieve the goal, we regard that labeled and unlabeled samples share some hidden concepts during the instance generation process. Our goal is to discover shared hidden concepts, based on the observed instance pairs, and use them as new features to link labeled and unlabeled samples for learning. The main contribution of the paper, compared to other existing work, is threefold: 1) a new view of semi-supervised learning via hidden concept-based instance generation process; 2) a new model for integrating labeled and unlabeled samples via hidden feature formulation; and 3) a framework for supporting cross-domain semi-supervised learning.

REFERENCES

[1] O. Chapelle, B. Scholkopf, and A. Zien, *Semi-Supervised Learning*. Cambridge, MA: MIT Press, 2006.

[2] X. Zhu, "Semi-Supervised Learning Literature Survey," Univ. Wisconsin-Madison, Madison, WI, Tech. Rep. Comp. Sci. TR 1530, 2007.

[3] T. Hofmann, "Unsupervised learning by probabilistic latent semantic analysis," *Mach. Learn.*, vol. 42, no. 1/2, pp. 177–196, Jan. 2001.

[4] A. Asuncion and D. Newman, UCI Machine Learning Repository, 2007.

[5] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proc. COLT*, 1998, pp. 92–100.

[6] A. Blum and S. Chawla, "Learning from labeled and unlabeled data using graph mincuts," in *Proc. ICML*, 2001, pp. 19–26.

[7] N. Chawla and G. Karakoulas, "Learning from labeled and unlabeled data: An empirical study across techniques and domains," *J. Artif. Intell. Res.*, vol. 23, no. 1, pp. 331–366, Jan. 2005.

[8] C. Chang and C. Lin. (2008). LIBSVM—A Library for Support Vector Machines. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm/

[9] W. Dai, Q. Yang, G. Xue, and Y. Yu, "Boosting for transfer learning," in *Proc. ICML*, 2007, pp. 193–200.

[10] C. Do and A. Ng, "Transfer learning for text classification," in *Proc. NIPS*, Vancouver, BC, Canada, 2005, pp. 299–306.

[11] K. Huang, Z. Xu, I. King, M. Lyu, and C. Campbell, "Supervised self-taught learning: Actively transferring knowledge from unlabeled data," in *Proc. IJCNN*, 2009, pp. 481–486.

[12] H. Lee, R. Raina, A. Teichman, and A. Ng, "Exponential family sparse coding with application to self-taught learning," in *Proc. IJCAI*, 2009, pp. 1113–1119.

[13] S. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2009.

[14] J. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann, 1993.

[15] R. Raina, A. Battle, H. Lee, B. Packer, and A. Ng, "Self-taught learning: Transfer learning from unlabeled data," in *Proc. ICML*, Corvallis, OR, 2007, pp. 759–766.

[16] S. Goldman and Y. Zhou, "Enhancing supervised learning with unlabeled data," in *Proc. ICML*, 2000, pp. 327–334.

[17] M. Rosenstein, Z. Marx, L. Kaelbling, and T. Dietterich, "To transfer or not to transfer," in *Proc. NIPS Workshop Inductive Transf.*, 2005.

[18] I. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*. San Mateo, CA: Morgan Kaufmann, 2005.

[19] A. Singh, R. Nowak, and X. Zhu, "Unlabeled data: Now it helps, now it does't," in *Proc. NIPS*, 2008, pp. 1–8.

[20] X. Zhu and X. Wu, "Class noise vs. attribute noise: A quantitative study of their impacts," *Artif. Intell. Rev.*, vol. 22, no. 3/4, pp. 177–210, Nov. 2004.

[21] J. Gao, W. Fan, J. Jiang, and J. Han, "Knowledge transfer via multiple model local structure mapping," in *Proc. KDD*, 2008, pp. 283–291.

[22] T. Zhang, K. Huang, X. Li, J. Yang, and D. Tao, "Discriminative orthogonal neighborhood-preserving projections for classification," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 40, no. 1, pp. 253–263, Feb. 2010.

[23] A. Dong and B. Bhanu, "Active concept learning in image databases," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 35, no. 3, pp. 450–466, Jun. 2005.

[24] F. Zhuang, P. Luo, H. Xiong, Y. Xiong, Q. He, and Z. Shi, "Cross-domain learning from multiple sources: A consensus regularization perspective," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 12, pp. 1664–1678, Dec. 2010.

[25] J. Attenberg and F. Provost, "Why label when you can search?: Alternatives to active learning for applying human resources to build classification models under extreme class imbalance," in *Proc. KDD*, 2010, pp. 423–432.

[26] R. Caruana, "Multitask learning," *Mach. Learn.*, vol. 28, no. 1, pp. 41–75, Jul. 1997.

[27] J. Tang, X. Hua, M. Wang, Z. Gu, G. Qi, and X. Wu, "Correlative linear neighbourhood propagation for video annotation," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 2, pp. 409–416, Apr. 2009.

[28] D. Miller and S. Uyar, "A mixture of experts classifier with learning based on both labeled and unlabeled data," in *Proc. NIPS*, 1997, pp. 571–578.

[29] P. Mallapragada, R. Jin, A. Jan, and Y. Liu, "SemiBoost: Boosting for semi-supervised learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 11, pp. 2000–2014, Nov. 2009.

[30] K. Bennett, A. Demiriz, and R. Maclin, "Exploiting unlabeled data in ensemble methods," in *Proc. KDD*, 2002, pp. 289–296.

[31] T. Zhang and F. Oles, "A probability analysis on the value of unlabeled data for classification problem," in *Proc. ICML*, Stanford, CA, 2000, pp. 1191–1198.

[32] J. He, J. Carbonell, and Y. Liu, "Graph-based semi-supervised learning as a generative model," in *Proc. IJCAI*, 2007, pp. 2492–2497.

[33] K. Nigam, A. Mccallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using EM," *Mach. Learn*, vol. 39, no. 2/3, pp. 103–134, May/Jun. 2000.

[34] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. IT-13, no. 1, pp. 21–27, Jan. 1967.

[35] X. He, "Incremental semi-supervised subspace learning for image retrieval," in *Proc. ACM SIGMM*, 2004, pp. 2–8.

[36] Z. Zhu, X. Zhu, Y. Guo, and X. Xue, "Transfer incremental learning for pattern classification," in *Proc. ACM CIKM*, 2010, pp. 1709–1712.

[37] X. Zhu, P. Zhang, X. Lin, and Y. Shi, "Active learning from stream data using optimal weight classifier ensemble," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 40, no. 6, pp. 1607–1621, Dec. 2010.

[38] B. Schölkopf and A. Smola, *Learning With Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA: MIT Press, 2001.

[39] T. Joachims, "Making large-scale SVM learning practical," in *Advances in Kernel Methods—Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, Eds. Cambridge, MA: MIT Press, 1999.

[40] T. Joachims, "Transductive inference for text classification using support vector machines," in *Proc. ICML*, 1999, pp. 200–209.

Xingquan Zhu (M'04) received the Ph.D. degree in computer science from Fudan University, Shanghai, China, in 2001.

He is a recipient of the Australia ARC Future Fellowship and a Professor of the Centre for Quantum Computation & Intelligent Systems, Faculty of Engineering and Information Technology, University of Technology, Sydney (UTS), Sydney, NSW, Australia. Before joining the UTS, he was a tenure track Assistant Professor in the Department of Computer Science & Engineering, Florida Atlantic University, Boca Raton, (2006–2009), a Research Assistant Professor in the Department of Computer Science, University of Vermont, Burlington, (2002–2006), and a Postdoctoral Associate in the Department of Computer Science, Purdue University, West Lafayette, IN (2001–2002). His research mainly focuses on data mining, machine learning, and multimedia systems. Since 2000, he has published more than 110 referred journal and conference proceedings papers in these areas. He is an Associate Editor of the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING (2009 to present), and a Program Committee Co-Chair for the 23rd IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2011) and the 9th International Conference on Machine Learning and Applications (ICMLA 2010).