Intimate Learning: A Novel Approach for Combining Labelled and Unlabelled Data

Zhongmin Shi and **Anoop Sarkar**
School of Computing Science, Simon Fraser University, Canada
{zshi1,anoop}@cs.sfu.ca

Abstract

This paper introduces a new bootstrapping method closely related to co-training and scoped-learning. The method is tested on a Web information extraction task of learning course names from web pages in which we use very few labelled items as seed data (10 web pages) and combine with an unlabelled set (174 web pages). The overall performance improved the precision/recall from 3.11%/0.31% for a baseline EM-based method to 44.7%/44.1% for intimate learning.

1 Intimate learning

The expensive nature of labelling data for machine learning methods and the lack of success in using purely unsupervised methods have motivated the study of learning methods that combine labelled and unlabelled data. Successful methods in this area include bootstrapping methods like co-training and scoped-learning. In this paper we introduce a novel method called *intimate learning* for bootstrapping that is related to these methods. The task is to learn the target function $h_t(X) \in Y$ where X is the set of all feature sets and Y is the set of class labels. The input to our learning algorithm is a set of n labeled examples of the form $(\mathbf{x_i}, y_i)$. $\mathbf{x_i} \in X$ has p_i features $x_{i1}, x_{i2}, ..., x_{ip_i}$ associated with the ith example. $y_i \in Y$ is the label of the ith example. In *intimate learning*, we find another class label $y'_i \in Y'$ which is relevant to y_i and we assume that a new function $h'(X) \in Y'$ requires fewer labelled items to learn. We say that Y' is relevant to Y if accurately finding $y'_i \in Y'$ can help in identifying $y_i \in Y$ in example i. We then create the new target function $h(X, h'(X)) \in Y$ that performs the same classification as $h_t(X)$ does. We call Y' the **intimate class**, $h'(X)$ the **intimate function**. Intimate learning is related to the co-training algorithm [Blum and Mitchell, 1998], in which for training examples $(\mathbf{x_i}, y_i)$, $\mathbf{x_i}$ is decomposed into a pair $(\mathbf{x_{i_1}}, \mathbf{x_{i_2}})$ corresponding to two different "views", and the target function $h(X) = h_1(X_1) = h_2(X_2)$ predicting a single label class. While in our model, X has only one "view" but labeled into two classes (target and intimate classes), i.e., $h_t(X) \neq h'(X)$[1]. Other related work is scoped-learning [Blei *et al.*, 2002], which uses a classifier trained on global features from the entire training data and classifiers trained on scope-limited features which are more specific to local subsets of the data.

2 The Information Extraction task

We apply intimate learning to the problem of identifying course names from those pages identified as course web pages in WebKB which consists of web pages collected from four universities. [2] For our experiment we used: 10 web pages as seeds, 174 web pages as unlabeled training data and 40 web pages as test data. All web pages are tokenized using space and punctuation symbols. We used an EM-based decision list learning algorithm [Collins and Singer, 1999] as our baseline method for combining labeled and unlabeled data (more details in §2.1). The course name feature model is initialized using the seed data and then trained using EM on the training data. Our observation on the training data indicated that the course number, which does not form part of our target label is likely to co-occur with a course name. Course names have similar characteristics to other named entities like names of people or organizations. Compared with course names, course numbers have far more regular forms, which usually consist of the department abbreviation and the number of the course, e.g., *CMPT 825*. As a result identifying a course number is easier and we take this class to be our intimate class Y'. Our target class Y determines whether a string is a course name. In this paper, we have Y and Y' with one element each (+*course-name* and +*course-number*), rather than a 2-class classification task (+*course-name*, −*course-name*). See §2.1 for details on how test examples are handled. We explain the details of our algorithm using Figure 1 (for now, ignore the dashed arrow). The algorithm is a two-stage process. The left half of Figure 1 illustrates the first stage, in which the intimate class is learned by a classification algorithm that is identical to the EM-based baseline system (see §2.2). The second stage in the right half implements almost the same EM-based algorithm, but with course numbers added from the first stage as an extra feature towards learning of the course name (the darker line with an arrow).

2.1 EM Learning for course number identification

The features used for identifying a course number are:

- The HTML format of the course number, which is the pair of HTML tags before and after the course number

[1] Intimate learning is *not* the same as feature selection: Y' is not directly observed. Furthermore Y' doesn't determine Y; X combined with Y' can improve prediction of Y.

[2] Other applications of intimate learning include finding product names in web pages, which appear together with prices, quantities, locations which are generally much easier to identify.

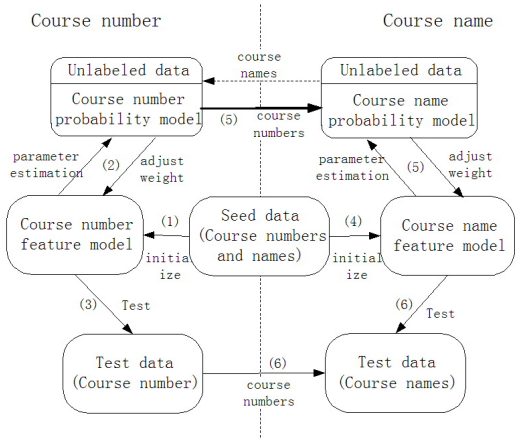

Figure 1: Course name identification system chart

respectively.

- Department name S and length $|S|$.
- Course id: integer I and its digit length $|I|$.
- The separator symbol sequence between S and I.

Features are collected for each string s_j from the seed data. We estimate the conditional probability $F_{num} = P(y' \mid x_{i,j})$ of seeing the label y' given the feature $x_{i,j}$. F_{num} defines a decision list of rules $x_{i,j} \rightarrow y'$ ranked by confidence score $P(y' \mid x_{i,j})$. In training, we use EM to (re)estimate the course number probability model $P_{num}(y' \mid s)$, where s is the input string. In the E-step, P_{num} is re-estimated based on F_{num}. We assume all features do not equally contribute to course number identification and assign different weight c_i to x_i, and each parameter of P_{num} is computed by:

$$P(y' \mid s_j) = \frac{\sum_i c_i \cdot P(y' \mid x_{i,j})}{\sum_i c_i} \quad (1)$$

In the M-step, F_{num} is adjusted with respect to P_{num}:

$$P(y' \mid x_{i,j}) = \frac{Count(x_{i,j}) \cdot P(y' \mid x_{i,j}) + P(y' \mid s_j)}{Count(x_{i,j}) + 1} \quad (2)$$

Since y' is a single label *+course-number*, for test examples we pick those to be labelled as a course number by using 2-means clustering to separate those examples that have high confidence scores (high $P(y' \mid x_{i,j})$ values) from those examples that have low confidence scores. This method avoids hand-picking or using a held-out set to pick a confidence threshold.

2.2 Course name identification

The learning procedure of the course name identification is the same as that for the course number, except that the course number identified using the model defined in §2.1 becomes an important feature in predicting whether an example is labelled as a course name. For each candidate course name, the features used are:

- *Intimate class:* course number preceding candidate
- The pair of HTML tags before and after course name.

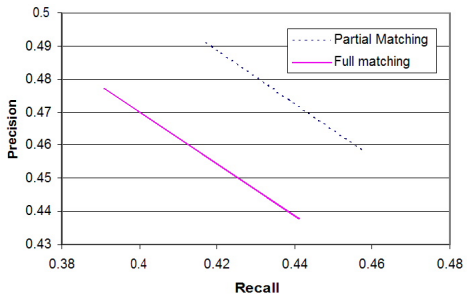

Figure 2: Experimental results of course name identification. The precision and recall curves for partial and full matchings.

- Each word in the course name and the number of words.
- Each separator symbol between the course number and course name and total number of such symbols.

The course name feature model F_{nam} is initialized from the seed data and trained by the EM algorithm defined in §2.1. Applying the trained feature model to the test data generates all course name candidates with their probabilities. We again apply the 2-means clustering algorithm defined in §2.1. Note that the chosen candidates are individual words instead of the full string as a course name. We simply group contiguous words as a single course name. The **baseline system** is identical to the course name feature model F_{nam} except that it does not use the identified course number as an input feature.

3 Experiments and discussion

Two metrics are applied to the performance evaluation of course name identification: *partial matching*, in which the course name is correctly recognized if any of its words is predicted, and *full matching*, in which the course name is correctly predicted only if all words of the course name are predicted and in the correct order. The precision/recall of the baseline system is **3.11%/0.31%** for *full matching*. Figure 2 illustrates the performance of our implementation on course name identification, for full and partial matchings. Our experiments show that the intimate learning algorithm exhibits significant gains in performance over the baseline system obtaining **44.7%/44.1%** for *full matching* and 45.8%/46.5% for *partial matching*. Since the course number and course name are two related classes, in addition to intimate learning, a co-training-based extension can also be applied to training one class by the other, and vice versa as shown by the dashed arrow in Figure 1. For details and full set of references, please refer to *http://natlang.cs.sfu.ca/researchProject.php?s=299*.

References

[Blei *et al.*, 2002] D. Blei, D. Bagnell, and A. McCallum. Learning with scope, with application to information extraction and classification. In *Proc. UAI*, 2002.

[Blum and Mitchell, 1998] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proc. COLT*, 1998.

[Collins and Singer, 1999] M. Collins and Y. Singer. Unsupervised models for named entity classification. In *Proc. EMNLP*, 1999.