Intelligent System for Workforce Classification

Yan Liu Zhenzhen Kou Claudia Perlich Richard Lawrence
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598
{liuya, zkou, perlich, ricklawr}@us.ibm.com

ABSTRACT

We examine several machine learning algorithms for assigning job-categories to a large employee population based on their online directory profiles and reporting structures. This application is an important step in workforce management, and also represents a challenging classification problem which typically contains large-scale unbalanced data, *i.e.*, the distribution of job categories is highly skewed. It also requires combining local information, *e.g.*, the textual description in an employee's webpage, with global structure, *e.g.*, the reporting chain structure, in order to predict job categories. We compare the performance of graphical models to several classification and relational learning algorithms with well-tuned features and show satisfactory accuracies on two datasets of more than 20,000 IBM employees. We also address some challenges that the graphical models face when applied to large-scale data.

1. INTRODUCTION

A common task in corporate workforce management is the assignment of employees to different job-categories including for instance Project Manager, Consultant, or IT Specialist. Employees within a job category are expected to demonstrate some level of expertise in a set of skills specific to the job category.

The identification and classification of employee skills is an increasingly important activity in large companies today for several reasons. More than ever, innovative companies are driving rapidly-changing strategic objectives, and this pace of change requires ongoing identification and prioritization of key skills and job roles. From a tactical perspective, companies that deliver consulting and technology services require clear views into the availability of specific skills required to staff expected near-term engagements. Corporate mergers and acquisitions, as well as large outsourcing agreements, can result in the transfer of a very large number of employees whose skill and job roles must be integrated into the parent company's job-skill taxonomy. Assignment of job category labels can be largely manual, guided by management input and by the broad responsibilities of the employee's immediate organizational group. Given the importance of these classifications, and the potentially large number of employees involved, it is useful to investigate automated approaches to this classification problem.

In this paper, we examine several machine learning models for job-category classification using data obtained from online employee directories. These data include organizational reporting structures as well as information provided by employees about their own job responsibilities. The models are trained using examples with known class labels taken as one of 21 distinct job-category labels. The problem represents a challenging classification task for several reasons:

1. the available textual information is very limited since most people do not provide (or update) a detailed description of their job responsibilities;

2. the population of employees can potentially be very large;

3. relevant information can come from different sources;

4. the distribution over different job categories is skewed - in our application about 80% employees fall into a few dominant classes.

Graphical models are a family of probability distribution defined over graphs, in which the nodes represent the variables and the edges represent the dependency relations between the variables [11]. In recent years, graphical models have been successfully applied to model complex relations in various applications, such as natural language processing, computer vision, and computational biology [10, 2, 6, 16]. With the vast increase in the amount of data available, a natural next step for data mining research is to examine the scalability and effectiveness of graphical models for large domains with complex multi-sourced data.

Given its power to capture complex relations between entities, we explore a graphical model approach for job category prediction by integrating the employee's own information with others via the reporting trees. We propose an empirical strategy to improve the scalability by breaking the dataset into smaller subsets. We also compare the graphical models to other relational learning methods, including stacked models, FOIL, and ACORA, as well as several non-relational machine learning approaches, such as Support Vector Machines, Naive Bayes, and Maximum Entropy, with different feature sets. We observe that graphical models with

DMBA'08, August 24, 2008, Las Vegas, Nevada, USA.

standard features perform competitively with non-relational models trained on well-turned features.

An additional contribution of this paper is the formulation of a graphical model tailored to the analysis of a large, unstructured data set drawn from workforce management. In particular, we strive to answer the following questions: Are there any efficient algorithms that can scale the graphical models for this large-scale application? Compared with other more efficient options, such as simple classifiers and stacking, will graphical models have superior advantages in terms of prediction power?

The remainder of the paper is organized as follows: Section 2 reviews previous related work. In Section 3, we describe the specific job classification task in more detail, including a description of the available data. Sections 4 and 5 describe our methods and experimental results, respectively. We conclude with a summary of the work and suggestions for future work in Section 6.

2. RELATED WORK

Large-scale datasets, *i.e.*, the datasets with a large number of training examples or attributes, are extremely common in nowadays modeling applications. For example, in web mining there are hundreds of millions of webpages available on the internet and they are still increasing dramatically; the Netflix competition involves the movie review records of half-million users [30]; in bioinformatics, we face the challenges to sequence, search and analyze hundreds of thousands proteins from different species and families [31]. One of the major bottlenecks in attacking these large-scale applications is the computational complexity involved with many of the more powerful machine learning algorithms. Previously, there have been several approaches developed or having the nature for large-scale learning tasks, including: (1) on-line learning algorithms [7]; (2) dimension reduction via singular value decomposition or sampling [14, 15]; (3) building effective data structures, such as KD-tree or R-tree, to reduce search space [17]; (4) fast optimization algorithms [4, 9].

Scaling of graphical models usually has multiple dimensions: on one hand, it could refer to the large number of training or testing examples. This issue can be alleviated via reasonable sampling. On the other hand, it can also imply a potentially large number of nodes in the concerned graph. For example, in the Netflix competition [25], the user-movie-review graph will have millions of nodes, which leads to huge complexity if a graphical model is applied. We are currently not aware of serious attempts to scale graphical models and make them more widely applicable.

3. TASK DESCRIPTION

Like most large companies today, IBM maintains an online directory consisting of profiles for all employees. Known internally as "Blue Pages", this directory contains the usual contact information, as well as other sources of information relevant to the task of inferring the job-classification label for each employee. These data include

1. the name of the employee's department (*e.g.* "Predictive Modeling"),
2. a relatively short, employee-entered description of the employee's job responsibilities (*e.g.* "Research in machine learning and data mining"),

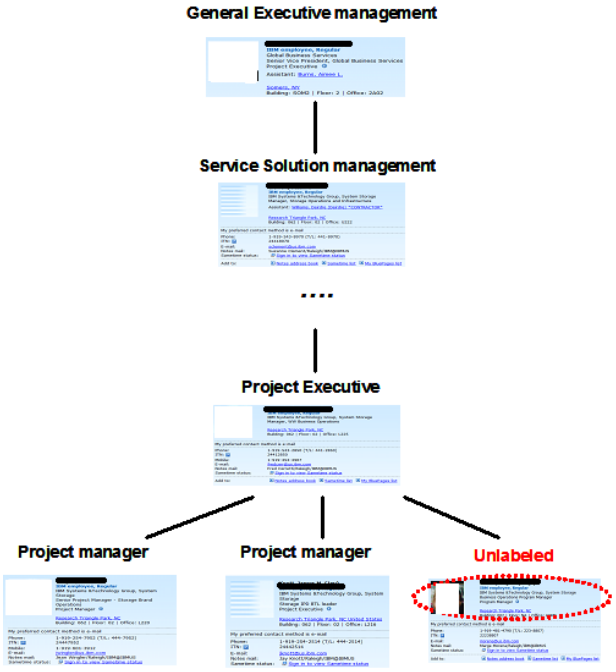

Figure 1: The reporting structure

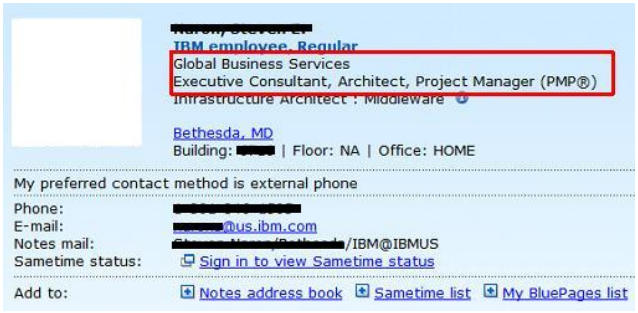

Figure 2: A challenging example

3. a separate section where an employee can provide a longer description of their experience, however this section is typically completed by only a small fraction of employees,
4. the organizational reporting tree, including links to the profile for each manager above the employee in the organization tree and also a list of peers of this employee.

Hence, we have profiles for each employee, for all managers in the reporting structure above them, and for each of their peers. Figure 1 shows an example of a Blue Pages reporting structure.

In order to achieve effective people management, each employee in IBM is assigned to a job category by human resources – in our data, there are 21 unique job-category labels. Figure 3 shows the distribution of these labels for one of our datasets described in detail in Section 5. Note that the 3 most common labels, "Consultant", "IT Specialist", and "Project Management" account for over 80% of the popula-

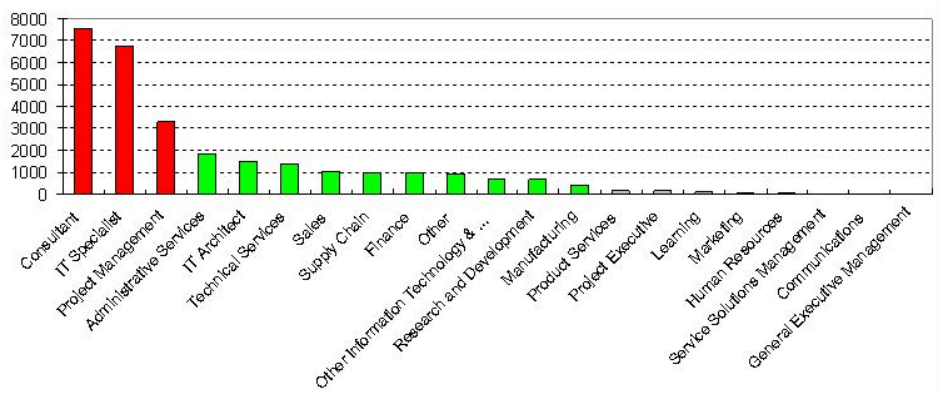

Figure 3: The distribution of job categories

tion[1]. Our goal is to automatically predict the job category label for each employee using the employee profile information.

The job category prediction problem is more challenging than a simple classification task. For example, Figure 2 illustrates a practical obstacle encountered in this specific application. The correct label for this employee is "IT Specialist", yet the description provided by the employee includes the terms "Consultant", "Architect", and "Project Manager" which are very similar to other labels. One explanation may be that the job-category label is out of date, and does not reflect the employee's current responsibilities. Another possibility is that this employee has a different view of his responsibilities than implied by the label. In any event, this issue introduces some uncertain labels into the process.

4. METHODS

From the data mining perspective, job category prediction is a classification problem with relational descriptors between examples. It is therefore structurally similar to many other domains, such as webpage classification with hyperlinks, gene function prediction with regulatory networks, and expert finder with social networks. Numerous approaches have been proposed for such setting:

1. We can treat it as a simple classification problem. Similar to text classification, we can convert the textual features into vectors using the "bag-of-words" representation and feed them into classifiers;

2. We can cast the task as a multi-view learning problem: for one employee, there are two disjoint sub-sets of features (or views), *i.e.* the employees' profiles as well as their managers' profiles. Therefore the multi-view learning algorithm can be applied if we assume each feature is sufficient to learn the job category;

3. We can also approach the task as a stacking problem: the reporting tree provides rich information about the relationships between employees, including their job categories. A computational efficient way to make use of these relations is stacking, *i.e.*, generating additional "local" features, such as the predicted labels from their managers or peers, and incorporating them into the models;

4. We can use general relational modeling approaches that explore the relationships between employees automatically and incorporate all available information including the textual information of both employees and managers/colleagues. There are only few relational learning algorithms that are both suitable for text classification and are sufficiently scalable for this domain including FOIL[23] and ACORA[22];

5. Finally, we can solve the problem using graphical models: we construct a graph with the nodes representing the employees and the edges denoting their reporting structures so that the "global" relations between employees can be captured. There are two variables associated with each node: the job category, either observed or to be inferred, and the employees profile features. Using the inferencing algorithms, we can predict the job category for unlabeled employees.

As we can see, these five methods progress from simple classifiers without modeling any dependencies between employees to graphical models that can capture the information within whole reporting trees.

4.1 Simple Classifiers

We are using a bag-of-word representation for the simple classification models. The basic bag-of-word feature set contains only the textual information available in the employee profile. An enriched bag-of-word feature set contains textual information from both the employee's and his manager's profiles. Here we consider different vocabularies for employees and managers, *i.e.*, if the word "business" appears both in the employee's and the manager's webpage, it is considered as different words.

We consider Support Vector Machines (SVMs), Naive Bayes, Maximum Entropy, and boosted Decision Trees. Support vector machines are discriminative classifiers that simultaneously minimize the empirical classification error and maximize the geometric margin [27]. SVMs have been very successful in many applications and are chosen as one of the

[1]This distribution is for a specific sample of IBM employees and is not indicative of the complete IBM employee population.

baselines in our approach. Naive Bayes (NB) is a probabilistic classifier based on Bayesian theorem with naive independence assumptions [18]. It often works well in many real-world applications and is particularly suited for text data. Maximum Entropy (MaxEnt) models have been successfully applied to many fields, *e.g.*, computer vision and natural language processing [24]. We apply MaxEnt to our job category prediction task since it can be seen as a counterpart of undirected graphical models without modeling the dependencies between examples. A Decision Tree is another popular classification algorithm in data mining, which uses information gain to find predictive input attributes and can generate concise and meaningful models. As is often done, we also apply AdaBoost algorithm to Decision Tree to improve the performance. Some classifiers, *e.g.*, SVMs, are defined as binary classifiers. We use the one-vs-all strategy to build a multi-class classifier.

4.2 Multi-view Learning

In addition to the employee profiles, the manager profiles are often informative about the job categories. For example, a manager whose job descriptions include "research on data mining" is most likely to manage people labeled as "research and development" instead of "sales" and "supply chain". Thus we can make use of two feature sets, *i.e.* employee profiles and manager profiles, and apply multi-view learning algorithms.

From labeling perspective, it is costly and time-consuming to assign the job labels to every new employee, especially when the company acquires a new sub-company and different terminologies are used. How to effectively use the labeled and unlabeled data is therefore of great interest for our task. We apply the multi-view learning (also known as co-training learning) algorithm as a way to explore both the multi-view learning and semi-supervised learning [3]. In particular, we are interested in answering the following questions: Does multi-view learning work for job category prediction? How many labeled training data are required to achieve a reasonable performance?

There are multiple versions of multi-view learning. We use the transductive version. The algorithm is described in Figure 4. Notice that our task is a multi-class classification problem. We can either convert it into several binary classification problems and run the multi-view learning iterations for each subproblem, or put the iteration outside [8]. We choose the latter approach and select the examples with highest prediction confidence into the active labeled set.

Input:labeled set $L = \{\mathbf{x}^L, y\}$, unlabeled set $U = \{\mathbf{x}^U\}$
Output: labeled set $L' = L \cup \{\mathbf{x}^U, \hat{y}^U\}$
Loop until $U = \emptyset$:

1. Train classifier C_1 on set L with employees' profile.
2. Train classifier C_2 on set L with managers profile.
3. Allow C_1 to label p examples with the highest confidence scores from U.
4. Allow C_2 to label p examples with the highest confidence scores from U.
5. Add these self-labeled examples to L.

Figure 4: Multi-View Learning Algorithm

4.3 Stacked Models

Besides the information from the manager, there is also rich information among peer workers. Considering the job category distribution among peers, it is natural to assume that a given employee is very likely to have a similar job category as his peers. Hence, the job category distribution among peers would be a useful feature. To estimate this distribution, we need to estimate the missing job categories first.

Though learning methods usually produce reasonably well-calibrated probability estimates on unseen test data, their estimates on training data are biased. Thus, to obtain the "predictions" for each training example $\mathbf{x}$, we apply *stacking* [29, 12, 28, 1], a meta-learning scheme based on cross-validation. In other words, we first run the simple classification model and use cross-validation to obtain the predicted job categories, as show in Figure 5. Then given the predicted job category for each example, the job category distribution of peer workers $r(y')$ is estimated and added to enrich the feature vector for each example, i.e., $\mathbf{x}' = (\mathbf{x}, r(y'))$. Finally we train a model with the expanded feature set. The process is similar in the testing phase. For the testing set T, we apply the model trained with $\{\mathbf{x}, y\}$ to obtain intermediate predictions over T, expand T to T', and apply the model trained with $\{\mathbf{x}', y\}$ to T' for the final prediction.

Given a training set $D = \{\mathbf{x}, y\}$ and a base learner A, construct cross-validated predictions $\hat{y}$ for $\mathbf{x} \in D$ as follows:

1. Split D into J equal-sized disjoint subsets $D_1 ... D_J$.
2. For $j = 1...J$, let $f_j = A(D - D_j)$, *i.e.*, train a classifier f_j for $j = 1...J$, based all the data from D except the subset D_j.
3. For $\mathbf{x} \in D_j$, $\hat{y} = f_j(\mathbf{x})$, *i.e.*, for data in D_j, apply the classifier f_j to obtain its prediction.

Figure 5: A cross-validation-like technique to obtain predictions in stacked models

The stacking performances are often similar to the ones of relational graphical models, especially when the estimated $r(y')$ can provide enough information about the dependencies among data. Compared with other relational learning approaches, stacking is easy to implement, efficient to compute, and flexible in feature construction since they can be built upon any classification algorithms.

4.4 Relational Models

Relational learning aims generally at modeling complex domains with interdependencies between entities of potentially different types. The job-classification domain can be represented in this framework by 4 relations: Job(Person, Label), Department(Person, Word), ManagerOf(Person, Person), Description(Person, Word).

Motivated by the original work on inductive inference in first order logic, FOIL[23] became one of the most widely used, scalable, and successful relational learner. FOIL is an extension of sequential-covering that learns a set of first order rules where each rule is a Horn clause that can contain negated literals but no function symbols. It has been applied successfully to relational text classification on a number of

Consultant(A) :- Manager(A,B), Description(B,'Supply')
Consultant(A) :- Manager(A,B), Description(B,'Consultant'), Description(A,'Consultant'), Manager(D,B), Description(D,'Consultant'), A<>D.
Consultant(A) :- Manager(A,B), Description(B,'Partner'), Description(A,'Partner'), Department(A,'Business'), Description(B,'Business').

Figure 6: A set of FOIL generated rules to predict Consultant based on managerial relationships and job descriptions.

domains [5, 32]. An example of a set of FOIL rules to predict the Consultant category is shown in Figure 6.

One of FOIL's limitations is the binary classification setting. While it is possible to setup a multi-class learning problem, it is impossible to enforce that every test example is assigned to exactly one class. We therefore only include results for FOIL on a single binary comparison against ACORA, a statistical relational learning algorithm which construct propositional features from the originally relational representation. The idea is fundamentally similar to the stacked approach, except that the feature construction is automated and does not rely on additional domain knowledge beyond the data schema. We chose ACORA[22] as an example of such an algorithm for its scalability and effectiveness in dealing with high-dimensional feature spaces. ACORA uses breadth-first-search to explore the relational domain structure and uses a combination of vector distances and naive Bayes to compress the relational information into propositional features. It has been successfully applied on a number of large and high-dimensional domains [22] including winning the ILP challenge 2005 [21].

4.5 Graphical Models

Graphical models are a powerful computational tool to represent and analyze complex statistical dependencies with graphs. In graphical models, nodes represent random variables, and edges represent conditional dependencies. Graphical models define a joint probability distribution over all the variables as a product of the local functions at the nodes of the graph and inference queries are answered by marginalization.

The job category classification can be formulated as an inference problem over a graphical model, in which the nodes denote individual employees and the edges represent the "reporting to" relationships. We build an undirected graphical model, *i.e.*, a Markov network, with each node associated with observed bag-of-word features based on the employee's profiles and hidden labels for the job categories (see Figure 7(A)). The potential function Φ, is defined to include the bag-of-word features in connected nodes. Following the idea of conditional random fields [13] and discriminative relational Markov networks [26], we have the conditional probability of the labels given the observation as follows:

$$\begin{aligned} &P(Y_1, \ldots, Y_n | \mathbf{x}_1, \ldots, \mathbf{x}_n) \\ &= \frac{1}{Z} \prod_{i=1}^{n} \Phi(\mathbf{x_1}, \ldots, \mathbf{x_n}, y_i) \Phi(\mathbf{x_1}, \ldots, \mathbf{x_n}, y_i, y_i^{\uparrow}) \\ &= \frac{1}{Z} \exp(\sum_{i=1}^{n} \sum_{k=1}^{K} \lambda_k f_k(\mathbf{x}_1, \ldots, \mathbf{x}_n, y_i, y_i^{\uparrow})), \end{aligned}$$

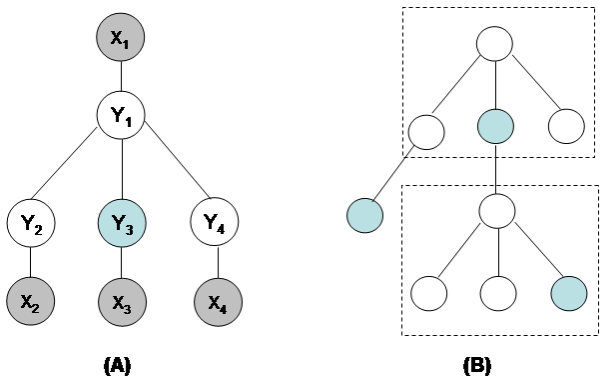

Figure 7: (A) Graphical model representation of our model for job category prediction: X and some of Y are observed variables (shaded nodes). (B) Example of subgraphs with 3 nodes

in which $y_i^{\uparrow}$ represent the neighbor of y_i on the upper depth of the reporting tree, *i.e.* the manager of y_i; f_k are the features defined over the pair of labels and the observations. We further factorize f_k as

$$f_k(\mathbf{x}_1, \ldots, \mathbf{x}_n, y_i, y_i^{\uparrow}) = f'_k(\mathbf{x}_i, \mathbf{x}_i^{\uparrow}) \delta(\langle y_i, y_i^{\uparrow} \rangle, \langle y, y' \rangle),$$

where δ is the indicator function and $\langle y, y' \rangle$ is the pair of label values. λ_k is the feature weight, which is usually learned by maximizing the likelihood of the training data. We do not have a fully labeled set on the whole graph to learn the parameters. Therefore we use an alternative estimation that trains one simple logistic regression using the labeled data by assuming each pair of employee and manager as independent and then use the corresponding learned weights in the graphical models.

To predict the labels for each node in the graph, we use the marginal probability, *i.e.*

$$\tilde{y_i} = \arg\max_{y} \; P(Y_i = y | \mathbf{x}_1, \ldots, \mathbf{x}_n).$$

The marginal probability can be estimated using Belief Propagation (BP), given that our graph topology is simply a tree [20]. The algorithm maintains a message $m_{j,i}(y_i)$ from node y_j to node y_i. The update from y_j to y_i is given by:

$$m_{j,i}(y_i) \leftarrow \sum_{y_j} \Phi(\mathbf{x}, y_j) \Phi(\mathbf{x}, y_i, y_j) \prod_{k \in \mathcal{N}(j)/i} m_{k,j}(y_j),$$

where $\mathcal{N}(j)$ is the neighbor of node y_j. Given the message vector m, the marginal probability can be computed as

$$p(y_i) \leftarrow \frac{1}{Z'_1} \Phi(\mathbf{x}, y_i) \prod_{j \in \mathcal{N}(i)} m_{j,i}(x_i),$$

where Z'_1 is the local normalization term over node y_i.

One practical difficulty of applying graphical models is the computational expense and the scale of the graph that can be handled. In our practice, the implemented Markov networks can process a graph with about 300 nodes at a time. To handle the computational constraints, we split the original reporting tree into sub-trees empirically, *i.e.*, we only allow to cut the edge from a non-leaf node to its manager, so as to keep the peers in one sub-graph (see Figure

7(B)). Borrowing the idea of generalized belief propagation [33], we first run belief propagation on each subtree, then update the messages between the nodes where an edge was cut, and run BP again to pass the new information to other nodes in the subgraph.

5. EMPIRICAL RESULTS

Given the scaling concerns for some of the algorithms, we conduct the experiments on three different subsets of the IBM employee population. The main comparison of all approaches discussed in Section 4 will be demonstrated on a data set with all employees of a single IBM *Subdivision*, which contains 20,320 US employees labeled with one of the 21 categories. It includes both the employee profiles of the whole population in that division and label information as well as the reporting chain, *i.e.*, for each employee there are labels of the employee's manager and peers.

To gain insights about the effect of data properties on the performance of different approaches, *i.e.*, sample size, sample population and features, we prepared two additional random subsets: a *Small Sample*, which contains 979 examples randomly selected from whole US employee population, maintaining the original ratio of the 21 different job categories; and a *Large Sample*, which includes 27,626 examples that were selected based on the employment year, i.e before a particular year cut-off. Since the samples are drawn independent of the hierarchy, the job categories for managers and peers may not available in general.

The distribution of the job categories are skewed in all three sets, For example, in the *Subdivision* set, there are 21 job categories yet about 80% employees fall into the 3 dominant classes. Figure 3 shows the distribution of the job categories in the large sample. All the experiment results are reported in prediction accuracy across all 21 categories on 10-fold cross-validation.

5.1 Models and Feature sets

Knowing that the domain is inherently relational, it would be a rather unfair comparison if we limit the simple classification model only to the immediately available local profile of an employee. It is always within the liberty of the modeler to construct additional features. We therefore consider the different approaches under slight variations of the feature set, which we group as:

F_1: employee profile;
F_2: profile of employee's manager;
F_3: explicit managerial chain (names of managers);
F_4: profiles of linked peers;
F_5: job categories of linked peers.

For **simple classification models**, two feature sets are examined: one is a base feature set with "bag-of-word" representation containing only information from the employee profiles F_1; the other is an enriched feature set with bag-of-word features from both the employee and the manager profiles $F_{1,2}$. We focus initially on the best algorithm Maximum Entropy and consider the performances of other simple classifiers in Section 5.3.

In the **multi-view learning**, we use the bag-of-word features F_1 from the employee's profiles and the manager profiles F_2 as the two independent views.

For **stacking** we use the features F_1 and F_2. The difference between the two variants of stacking is whether to use the known training labels F_5 whenever available or always use the predicted labels, whether known or not.

Finally, the **relational learning** approach ACORA utilizes the specific position of an employee in the organization via the manager name F_3 in addition to the features F_1, F_2. Provided the search depth is sufficient, ACORA can even include F_4 and F_5. However, given the significant increase in runtime and the limited improvement in initial experiments we limit ACORA to F_1, F_2, F_3. In [22] we provide an explanation that identifiers, such as manager names, can act as a Markov boundary and once accounted for, decrease the additional value of additional features. Since FOIL is limited to non-recursive clauses it will only consider F_1, F_2, F_3, F_4.

Beyond the feature space, another worthwhile distinction between the approaches is the scope of inference: stacked models and relational learning models are able to capture only the local dependencies while graphical models can use belief propagation to capture global dependencies. We therefore consider that the **graphical models** have access to the complete feature set F_1, F_2, F_4, F_5.

5.2 Main Comparison: IBM Subdivision

The *Subdivision* data provides a complete relational set that can be represented via a tree, considering the employee as nodes and reporting chains as edges. This data set serves as a good representative for our motivating applications, since corporate mergers and acquisitions usually result in the transfer of employees into existing divisions (where the labels of most population is available) and the labels for the transferred employees are desired. We examine all the approaches discussed in previous section on this dataset.

Table 1 shows the performance achieved by the different approaches with varying feature sets. We observe:

1. With well-tuned features, simple classification models can achieve very satisfactory accuracies. In particular, once we include information about the manager profile F_2, the best performance of simple classifiers is close to other more sophisticated approaches.

2. Multi-view learning is remarkably bad. It is even worse than the simple classifier on the local features. We investigate this artifact further in Section 5.3. The main reason for the comparatively poor performance is the relatively larger size of the data. Multi-view learning is helpful only when there are few labeled examples, which does not hold in our task. For the same reason,

Table 1: Comparison of performance and feature sets for different approaches. $\star$, $\triangle$ and $\diamond$ represent different performance levels based on significance tests: *i.e.* $\star$ for the best performer, $\triangle$ for the second performance level, and $\diamond$ for the third.

Algorithms	Accuracy	Features
Simple classifier $^{\diamond}$	76.5	F_1
Simple classifiers, enriched $^{\triangle}$	82.6	$F_{1,2}$
Multi-view learning $^{\diamond}$	73.9	$F_{1,2}$
Stacking $^{\triangle}$	82.9	$F_{1,2}$
Stacking with Labels *	84.5	$F_{1,2,5}$
Relational learning (ACORA) $^{\triangle}$	82.1	$F_{1,2,3}$
Graphical Models $^{\triangle}$	83.1	$F_{1,2,4}$

we will not run multi-view learner on other feature sets.

3. The stacked models on F_1, F_2 are comparable to the simple classifiers with enriched features. However, once we use the observed training labels F_5 whenever available instead of the prediction the performance increases substantially.

4. The relational learning approach ACORA performs slightly worse than even the simple model with extended features. A possible explanation for the poor performance is: despite its ability to utilize the relational information, the underlying components of ACORA are mostly based on Naive Bayes. As is shown in Table 3, Naive Bayes is suboptimal for our task compared with other classifiers. A related question is how FOIL would have performed on this task. We are unable to include the results of FOIL due to its limitation to binary classes and its huge computational costs (it ran on one of the 10 folds for one of the 21 classification tasks for more than 12 hours without completion). However, we do compare the performance of ACORA and FOIL on the binary task of classifying employees as Consultant and the results show that FOIL is not competitive in this domain.[2]

5. The graphical model achieves a better performance than some of the other approaches, but is inferior to stacking. There are three potential reasons for less competitive performance of graphical models: first, the tree representing the whole dataset with 20,320 nodes can not be processed via the inference algorithm. Therefore we have to split the graph into sub-graphs by cutting the edges between a non-leaf node and its parent, as described in Section 4.5. This approximation might result in the loss of important relational information; second, the local dependencies may dominate the job category assignment. This is consistent with the good performances of the simpler approaches.

In summary, the stacked models perform the best with additional features generated from the true labels. Given the same set of features, graphical models, ACORA and stacked models perform similar, with the accuracy by graphical models slight better (but not statistically significant) than the other two.

In addition to the prediction accuracy, we also comment on the computational complexity of these approaches. Based on the running time, simple classifiers, especially Naive Bayes and Decision Trees, is the most efficient ($5 \sim 30$ mins); stacking and multi-view learning are comparable to simple classification ($0.5 \sim 5$ hours) since these two algorithms are built on top of the base classifiers; graphical model is much slower (~ 20 hours) while relational learning algorithms is the least efficient, which takes over 24 hours. The running time may not be an accurate estimator of complexity due to different implementations, but at least it provides us a guidance when selecting efficient solutions.

5.3 Effect of Sample Size

[2]ACORA accuracy: 90.5% ; FOIL accuracy 83%; Prior: 72%

Table 2: Performance of different classifiers on data sets of varied size (*Small Sample* with 979 examples, *Large Sample* with 27,626 examples and *Subdivision* with 20,320 examples) using different feature set (limited to F_1 and extended using F_1, F_2)

Algorithm	Small Sample		Large Sample		Subdivision	
Features	F_1	F_1, F_2	F_1	F_1, F_2	F_1	F_1, F_2
Naive Bayes	55.5	61.6	67.3	69.5	72.7	76.0
SVMs	52.2	58.5	67.3	74.6	72.2	80.5
Decision Tree	**61.4**	59.2	65.8	71.4	70.2	77.2
Boosted Tree	57.6	**64.3**	70.7	76.6	75.6	82.2
MaxEnt	60.1	64.2	**71.2**	**78.2**	**76.5**	**82.6**

In addition to the performance on the *Subdivision*, we are also interested in answering questions about the generality of the results across data sets. In particular, we explore the effect of sample size on the performance. In our previous results, we can conclude that simple classifiers are able to provide reasonable results with much less running time. Therefore we applied simple models on a large and a small random sample to evaluate the impact of the size of training data and feature sets. The results are summarized in Table 2. As we can see, the extended features with the managers' profiles F_2 consistently help to improve the performance; a larger training set usually results in better performance; the performance on examples from one division are better than those from different divisions in the company. Among the simple models, Maximum Entropy and Boosted Decision Tree yields the best accuracy in general.

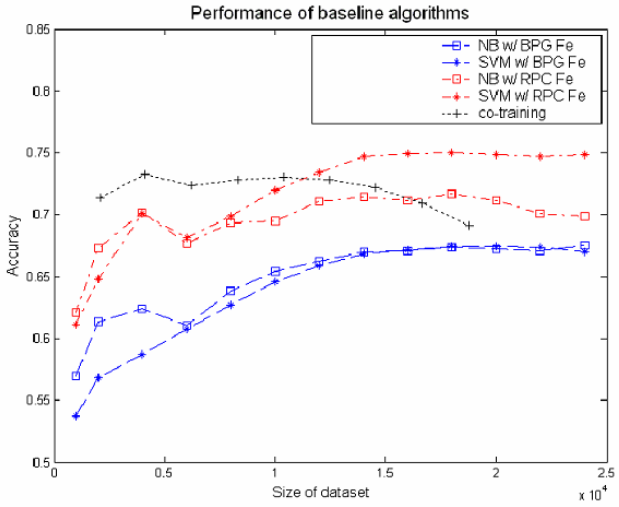

Figure 8: The multi-view learning curve on the *Large Sample*. Acronyms in the legend: NB (Naive Bayes), SVM (support vector machines), BPG Fe (features built from employee's profile), RPC Fe (features built from the employee's and manager's profiles)

In addition to simple classifiers, we also examine the data saturation of multi-view learning to find out how many labeled examples are needed. Since more labeled training data can improve the accuracy (as suggested by Table 2), it would

be interesting to find out how the semi-supervised learning can help improve the prediction. We first examine the multi-view learning algorithm on the small random set, and obtain an accuracy of 73.6%, which gives an improvement of about 14% compared to the performance of simple classifiers with the same feature set (64.3% → 73.6%). Next we apply multi-view learning on increasingly larger training set to answer two questions:

1. Does multi-view learning help when there is a relatively large set of labeled data available?
2. How many labeled training data are required to achieve a reasonable performance?

As shown in Figure 8, when the number of labeled examples are limited, *i.e.* less than 1,000, multi-view learning improves the performance significantly over the simple classifiers. With more and more labeled data available, *i.e.* from 2,000 to 5,000, the improvement becomes less significant and even negative. Similar behaviors have been observed in other applications of multi-view learning, in part because of the instability of the algorithms.

6. CONCLUSION

In this paper, we examine several approaches for large-scale workforce classification. We not only aim at solving an important task in workforce management, but also address the challenges of graphical models for large-scale applications by developing an empirical strategy for efficient inferences. The experiment results show that the graphical model is able to achieve comparable results with other classification or relational learning methods using well-engineered features as input.

The contributions of our work include: (1) we provide comprehensive solutions to the workforce classification task and achieve a competitive accuracy of around 84%, which enable us to deploy our solutions for human resource uses. Currently we have initiated our modeling framework in project management and expert finder systems with promising results; (2) we use the job category classification as a study case for comparative analysis on classification and relational learning and include an unprecedented number of different algorithms. On the one hand, our work confirms that relational learning is able to achieve better performance; on the other hand, most relational learning algorithms or graphical models with approximate inference algorithms can only successfully capture the local information, which can be achieved using well-engineered features. Therefore designing algorithms that capture the global information is crucial for future research.

7. ACKNOWLEDGEMENT

We would like to thank Ching-Yung Lin for general discussion and his help on preparing the data; Chid Apte for discussion on the task; and William Cohen for his insights on relation learning.

8. REFERENCES

[1] K. P. Bennett, A. Demiriz, and R. Maclin. *Exploiting unlabeled data in ensemble methods*. In Proceedings of the 8th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'02), 2002.

[2] J. Bilmes. *Graphical models and automatic speech recognition*. The Journal of the Acoustical Society of America, 5(2202), pp. 2278-2279.

[3] A. Blum and T. Mitchell. *Combining Labeled and Unlabeled Data with Multi-view learning*. In Proceedings of the Workshop on Computational Learning Theory (COLT), 1998.

[4] J. Boyan, W. Buntine, and A. Jagota (Eds.). *Statistical Machine Learning for Large Scale Optimization*, Neural Computing Surveys, 3, pp. 1-58, 2000.

[5] W. W. Cohen. *Text categorization and relational learning*. In Proceedings of the 12th International Conference on Machine Learning (ICML), 1995.

[6] N. Friedman. *Inferring cellular networks using probabilistic graphical models*, Science, 303(2004), pp. 799-805.

[7] Y. Freund and R. E. Schapire. *A decision-theoretic generalization of on-line learning and an application to boosting*. In Proceedings of EuroCOLT, 1995.

[8] R. Ghani. *Combining Labeled and Unlabeled Data for MultiClass Text Categorization*. In Proceedings of International Conference on Machine Learning (ICML 2002), 2002.

[9] T. Joachims. *Training Linear SVMs in Linear Time*. In Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD), ACM, 2006.

[10] M. Johnson, S. P. Khudanpur, M. Ostendorf, and R. Rosenfeld. *Mathematical Foundations of Speech and Language Processing Series: The IMA Volumes in Mathematics and its Applications, Vol. 138*, 2004, Springer.

[11] M. I. Jordan. *Graphical models*, Statistical Science (Special Issue on Bayesian Statistics), 19(2004), pp. 140-155.

[12] Z. Kou and W. W. Cohen. *Stacked Graphical Learning for Efficient Inference in Markov Random Fields*. In Proceedings of 2007 SIAM International Conference on Data Mining (SDM 07), Minneapolis, MN, 2007.

[13] J. Lafferty, A. McCallum, and F. Pereira. *Conditional random fields: Probabilistic models for segmenting and labeling sequence data*. In Proceeding of International Conf. on Machine Learning (ICML;01), 2001.

[14] P. Li. *Very Sparse Stable Random Projections For Dimension Reduction in Lp* $(0 < p <= 2)$ *Norm*. In Proceedings of ACM International Conference on Knowledge Discovery and Data Mining (KDD'07), 2007.

[15] P. Li, K. Church, and T. Hastie. *Conditional Random Sampling: A Sketch-based Sampling Technique for Sparse Data*. In Proceedings of Neural Information Processing Systems (NIPS), 2007.

[16] Y. Liu, J. Carbonell, P. Weigele, and V. Gopalakrishnan. *Protein Fold Recognition Using Segmentation Conditional Random Fields (SCRFs)*, Journal of Computational Biology, 13(2006), pp. 394-406.

[17] T. Liu, A. W. Moore, A. Gray, and K. Yang. *An investigation of practical approximate nearest neighbor algorithms*. In Proceedings of Neural Information Processing Systems (NIPS'04), 2004.

[18] A. McCallum and K. Nigam. *A comparison of event models for Naive Bayes text classification*. In Proceedings of AAAI-98 Workshop on Learning for Text Categorization, 1998.

[19] I. Muslea, S. Minton and C. A. Knoblock. *Active + Semi-supervised Learning = Robust Multi-View Learning*. In Proceedings of the Nineteenth International Conference on Machine Learning (ICML'02), 2002.

[20] J. Pearl. *Reverend Bayes on inference engines: A distributed hierarchical approach*. In Proceedings of AAAI, 1982.

[21] C. Perlich. *Approaching the ILP Challenge: Class-conditional Bayesian Propositionalization for Genetic Classification"*. In Proceedings of the 15th International Conference on Inductive Logic Programming (IL): Late-breaking papers. 2005.

[22] C. Perlich and F. Provost. *Distribution-Based Aggregation for Relational Learning from Identifier Attributes*. Machine Learning, vol 62, pp65–105, 2006.

[23] J.R. Quinlan and R.M. Cameron-Jones. *FOIL: A Midterm Report*. In Proceedings of the 6th European Conference on Machine Learning (ECML), 1993.

[24] A. Ratnaparkhi. *A simple introduction to maximum entropy models for natural language processing*. Technical Report 97-08, Institute for Research in Cognitive Science, University of Pennsylvania, 1997.

[25] R. Salakhutdinov, A. Mnih, and G. Hinton. *Restricted Boltzmann machines for collaborative filtering*. In Proceedings of the 24th International Conference on Machine Learning (ICML07), Corvalis, Oregon, 2007.

[26] B. Taskar, P. Abbeel and D. Koller. *Discriminative Probabilistic Models for Relational Data*. In Proceedings of Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI'02), 2002.

[27] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.

[28] J. Vittaut, M. Amini, and P. Gallinari. *Learning Classification with Both Labeled and Unlabeled Data*. In Proceedings of the 13th European Conference on Machine Learning (ECML'02), Helsinki, Finland, 2002.

[29] D. H. Wolpert. *Stacked generalization*, Neural Networks, vol. 5, pp241–259, 1992.

[30] http://www.netflixprize.com/.

[31] http://www.genome.gov/10001772.

[32] Y. Yang, S. Slattery and R. Ghani. *A study of approaches to hypertext categorization*. Journal of Intelligent Information Systems, vol. 18, 2002.

[33] J.S. Yedidia, W.T. Freeman and Y. Weiss. *Generalized Belief Propagation*. In Proceedings of Neural Information Processing Systems, 2000.