



Intelligent System for Workforce Classification

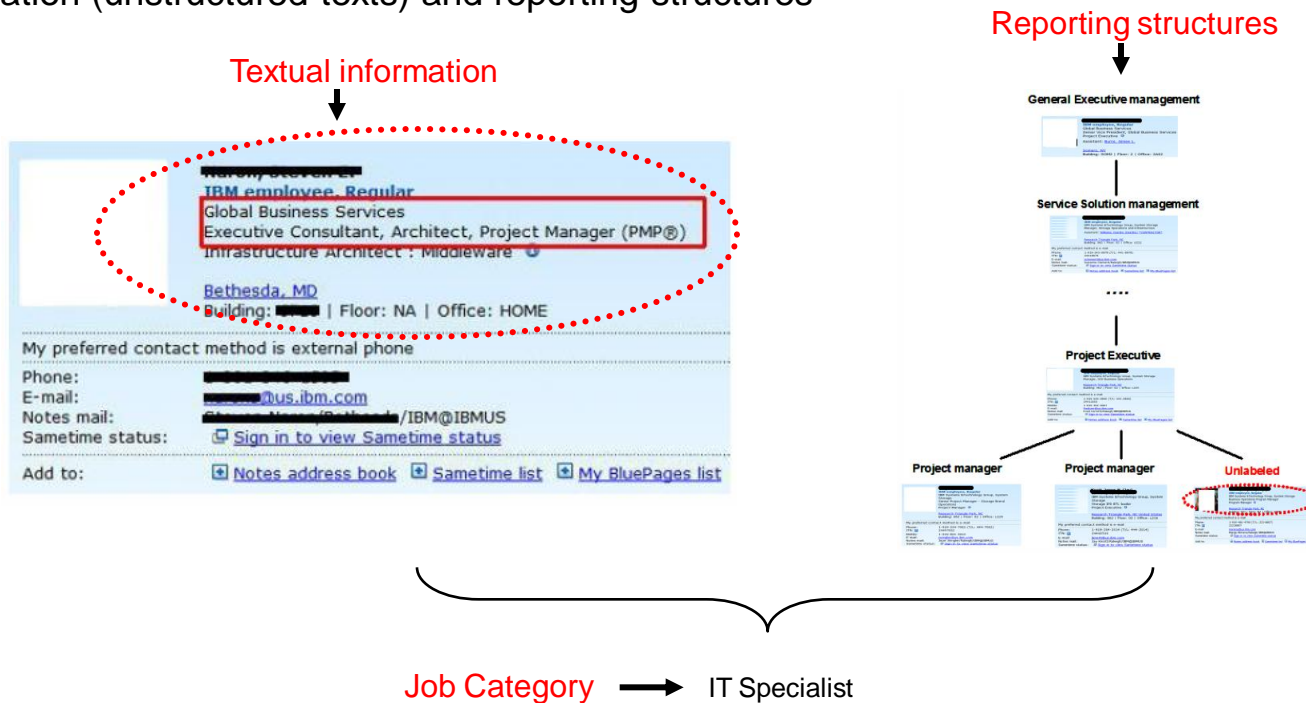
Yan Liu, Zhenzhen Kou (Yahoo!), Claudia Perlich, Rick Lawrence

Predictive Modeling Group
IBM T.J. Watson Research Center

Aug 24, 2008

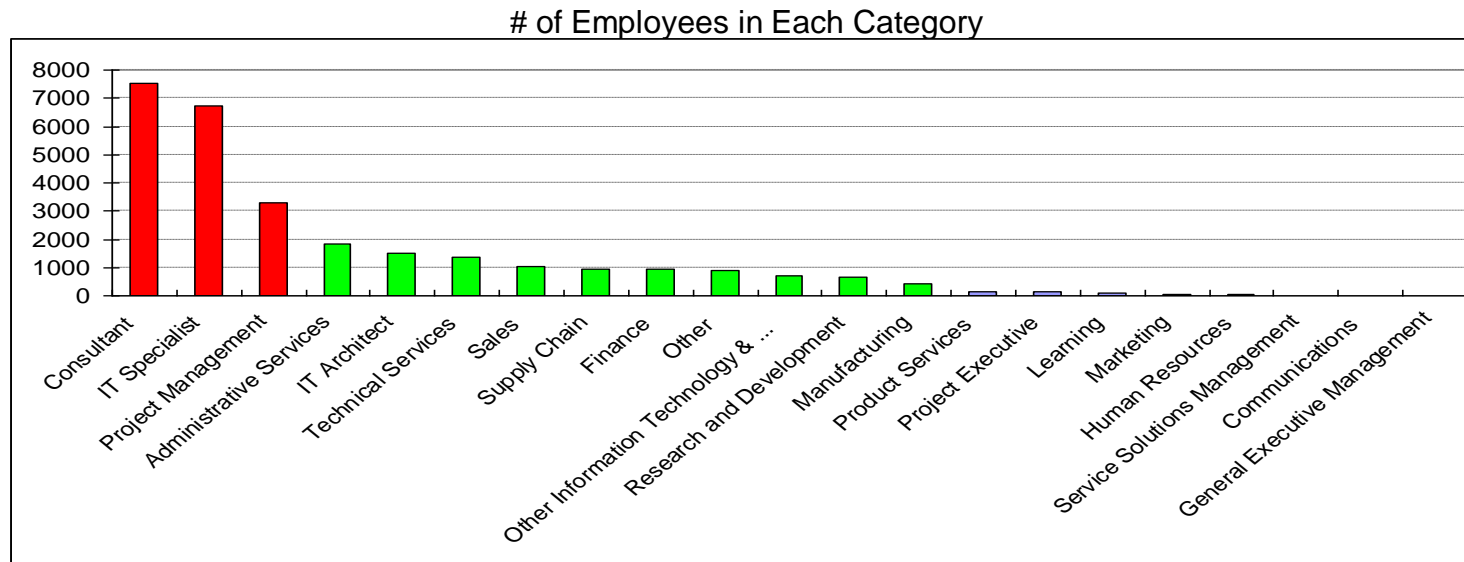
Workforce classification: Job Category Prediction

- Assigning relevant job-categories to a large employee population is an essential step in workforce management
 - Consulting and technology services require clear views into the availability of specific skills required to staff expected near-term engagements
 - Rapidly changing strategic objectives, corporate mergers and acquisitions
- It is particularly challenging if there is no structured information available such as skills assessments etc
- Technical Objective: automatically predict the job category of an employee using his/her profile information (unstructured texts) and reporting structures



Data Mining Task

- Workforce classification task can be cast as a relational learning problem
 - ▶ Input: a set of labeled examples and unlabeled examples, a graph describing the relationship of examples
 - ▶ Output: predicted labels of the unlabeled examples
- Challenges
 - ▶ Effectively combining textual information and graph information in large scale applications
 - ▶ Skewed distribution of classes
 - ▶ Very limited information available in the employee profiles
 - ▶ Labeling uncertainty



Goal: Examine of Possible Solutions for Our Task

1. Simple classification
2. Multi-view learning
3. Stacking algorithms
4. Statistical relational learning models
5. Graphical model approach

Independent

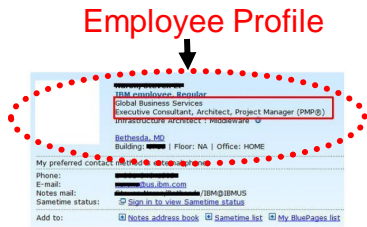
*Locally
dependent*

*Global
dependent*



Solution # 1: Simple Classification

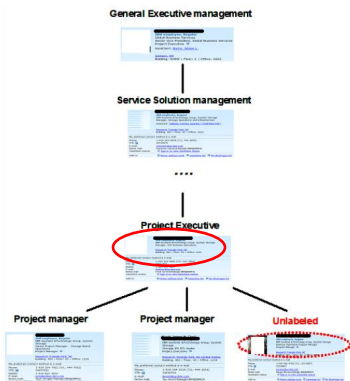
- Use bag-of-words extracted from the employee's Bluepage



TermID	T ₁	T ₂	T ₃	T ₄	...	T _n
Doc Vec	0.33	0.00	0.00	0.47	...	0.53

- Classifiers**
- Naïve Bayes
 - Boosting decision tree
 - Support Vector Machines
 - Max Entropy

- For each employee, we have his profile as well as his managers' profiles



TermID	T ₁	T ₂	T ₃	T ₄	...	T _n
Doc Vec	0.33	0.00	0.00	0.47	...	0.53
TermID	T ₁	T ₂	T ₃	T ₄	...	T _n
Doc Vec	0.33	0.00	0.00	0.47	...	0.53
TermID	T ₁	T ₂	T ₃	T ₄	...	T _n
Doc Vec	0.33	0.00	0.00	0.47	...	0.53

Classifiers

Solution # 2: Multi-view Learning

- Two type features available
 - ▶ Employee's profile
 - ▶ Employee's managers' profile

- We apply the co-training learning algorithm as a way to explore both the multi-view learning and semi-supervised learning
 - ▶ Transductive version
 - ▶ Multi-class classification

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 .
-

Solution #4: Relational Learning

- Relational learning aims generally at modeling complex domains with interdependencies between entities of potentially
- Logic-based relational learning v.s. statistical relational learning

- We define 4 relations

- ▶ Job(Person, Label)
- ▶ Department(Person, Word)
- ▶ ManagerOf(Person, Person)
- ▶ Description(Person, Word)

Examples of Rules by FOIL

```

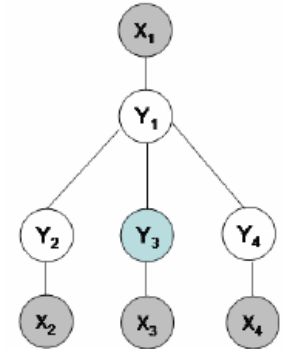
Consultant(A) :- Manager(A,B), Description(B,'Supply')
Consultant(A) :- Manager(A,B), Description(B,'Consultant'), Description(A,'Consultant'), Manager(D,B), Description(D,'Consultant'), A<>D.
Consultant(A) :- Manager(A,B), Description(B,'Partner'), Description(A,'Partner'), Department(A,'Business'),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

```

- Due to the complexity of FOIL, we also use ACORA, a statistical relational learning algorithm which construct propositional features from the originally relational representation
- ACORA 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

Solution #5: Graphical Models

- Graphical models are a powerful computational tool to represent and analyze complex statistical dependencies with graphs
- We build an undirected graphical model, i.e., a Markov network
 - Observed variables: bag-of-word features based on the employee's profiles
 - Hidden labels: labels of the job categories
 - The conditional probability is defined as follows:



$$\begin{aligned}
 P(Y_1, \dots, Y_n | \mathbf{x}_1, \dots, \mathbf{x}_n) &= \frac{1}{Z} \prod_{i=1}^n \Phi(\mathbf{x}_1, \dots, \mathbf{x}_n, y_i) \Phi(\mathbf{x}_1, \dots, \mathbf{x}_n, y_i, y_i^\dagger) \\
 &= \frac{1}{Z} \exp\left(\sum_{i=1}^n \sum_{k=1}^K \lambda_k f_k(\mathbf{x}_1, \dots, \mathbf{x}_n, y_i, y_i^\dagger)\right) \longrightarrow f'_k(\mathbf{x}_i, \mathbf{x}_i^\dagger) \delta(\langle y_i, y_i^\dagger \rangle, \langle y, y' \rangle)
 \end{aligned}$$

- Efficient algorithms Huge computational complexity is major bottleneck
 - Training phase
 - 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
 - Testing phase
 - 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
 - Borrowing the idea of generalized belief propagation, we first run belief propagation on each subtree, then update the messages between the nodes where an edge was cut

Experiment Results

▪ Data sets

▶ Subdivision set

- all employees of a single IBM subdivision, which contains 20,320 US employees labeled with one of the 21 categories

▶ Small Sample set

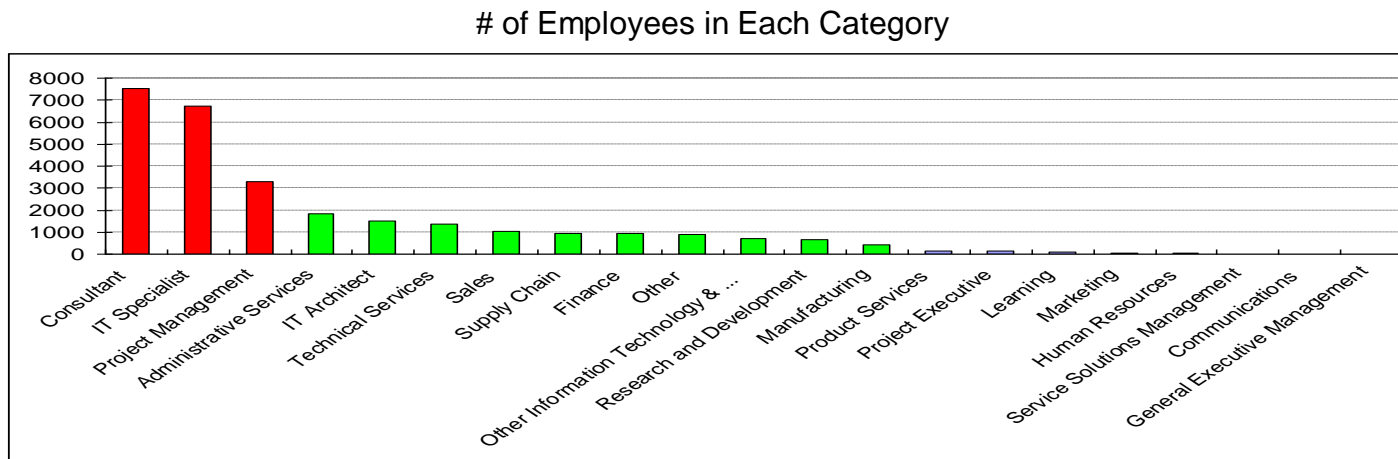
- 979 examples randomly selected from whole US employee population, maintaining the original ratio of the 21 different job categories

▶ Large Sample set (across division)

- 27,626 examples selected based on the employment year (before a particular year cut-off)

▪ Evaluation

▶ Prediction accuracy across all 21 categories on 10-fold cross-validation



Experiment Results

Feature sets:

- ▶ F1: employee profile
- ▶ F2: profile of employee's manager
- ▶ F3: explicit managerial chain (names of managers)
- ▶ F4: profiles of linked peers
- ▶ F5: job categories of linked peers

Results

- ▶ Best performer: Stacking with predicted labels
- ▶ Given the same set of features, graphical models, ACORA and stacked models perform similarly, with the accuracy by graphical models slight better

Running time

- ▶ Simple classifiers: 5~ 30 mins
- ▶ Stacking, multiview learning: 0.5 ~ 5 hour
- ▶ Graphical model: ~ 20 hours
- ▶ ACORA: ~ 24 hours

Baseline Results

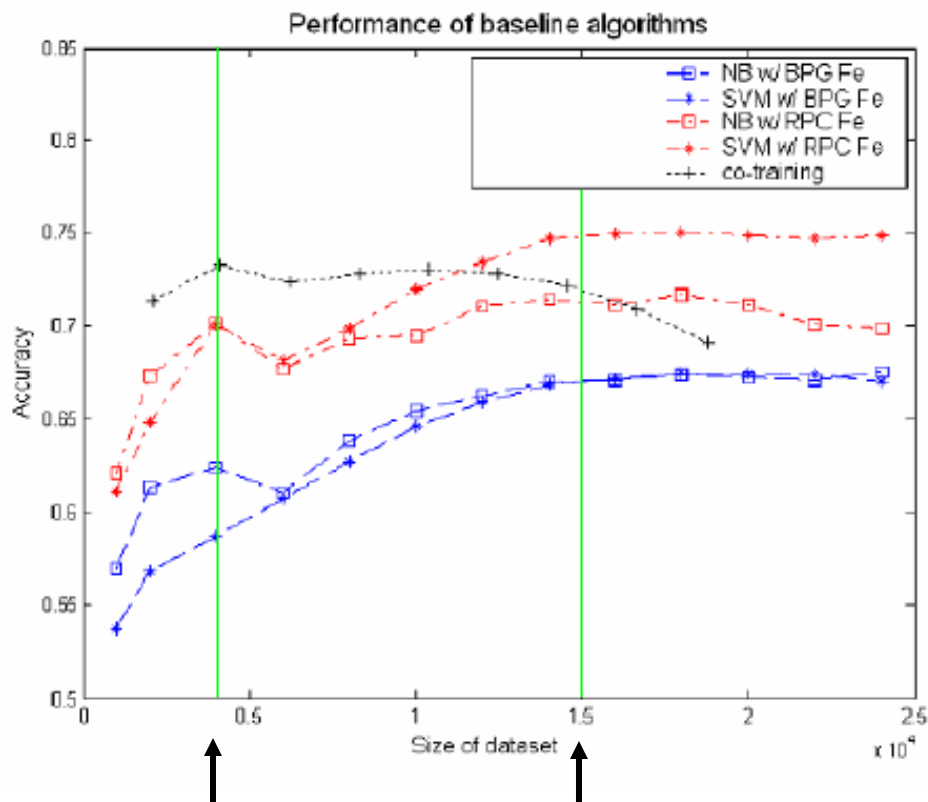
Algorithm Features	Small Sample		Large Sample		Subdivision	
	F_1	F_1, F_2	F_1	F_1, F_2	F_1	F_1, F_2
Naive Bayes	55.5	61.6	67.3	69.5	72.7	76.0
SVMs	52.2	58.5	67.3	74.6	72.2	80.5
Decision Tree	61.4	59.2	65.8	71.4	70.2	77.2
Boosted Tree	57.6	64.3	70.7	76.6	75.6	82.2
MaxEnt	60.1	64.2	71.2	78.2	76.5	82.6

Comparison Results on Division set

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

Effective Sample Size

- We demonstrated that simple classifiers are able to provide reasonable results with much less running time
- Questions: How many labeled training data are required to achieve a reasonable performance?



Conclusion

- We examine several approaches for large-scale workforce classification
 - ▶ Including: simple classification, multi-view learning, stacking algorithms, statistical relational learning models, graphical model approach
 - ▶ Comprehensive solutions to the workforce classification task and achieve a competitive accuracy

- Summary of technical insights
 - ▶ Relational learning is able to achieve better performance
 - ▶ 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

