Efficient Interactive Training Selection for Large-scale Entity Resolution

Qing Wang, Dinusha Vatsalan and Peter Christen
{qing.wang,dinusha.vatsalan,peter.christen}@anu.edu.au

Research School of Computer Science
The Australian National University
Canberra ACT 0200, Australia

This research was partially funded by

the Australian Research Council (ARC), Veda, and Funnelback Pty. Ltd.,
under Linkage Project LP100200079.

Entity Resolution – Introduction

- *Entity resolution* (ER) is to determine whether or not different entity representations (e.g., records) correspond to the same real-world entity.

Entity Resolution – Introduction

- *Entity resolution* (ER) is to determine whether or not different entity representations (e.g., records) correspond to the same real-world entity.
- Consider the following relation AUTHORS:

aid	name	affiliation	email
1	Qing Wang		qw@gmail.com
2	Mike Lee	Curtin University	
3	Qinqin Wang	Curtin University	
4	Jan Smith		jan@gmail.com
5	Q. Wang	University of Otago	qw@gmail.com
6	Jan V. Smith	RMIT	jan@gmail.com
7	Q. Q. Wang		
8	Wang, Qing	University of Otago	

– Are Qing Wang (1) and Q. Wang (5) the same person?
– Are Qinqin Wang (3) and Q. Wang (5) not the same person?
– ...

Entity Resolution – Training Data

- Various techniques, including supervised and unsupervised learning, have been proposed for ER in past years.
- Training data is generally in the form of *true matches* and *true non-matches*, i.e., pairs of records.
- Supervised techniques generally result in much better matching quality; nonetheless, these techniques require training data.
- In most practical applications, training data have to be manually generated, which is known to be difficult both in terms of cost and quality.
- Two challenges stand out:

(1) How can we ensure "good" examples are selected for training?

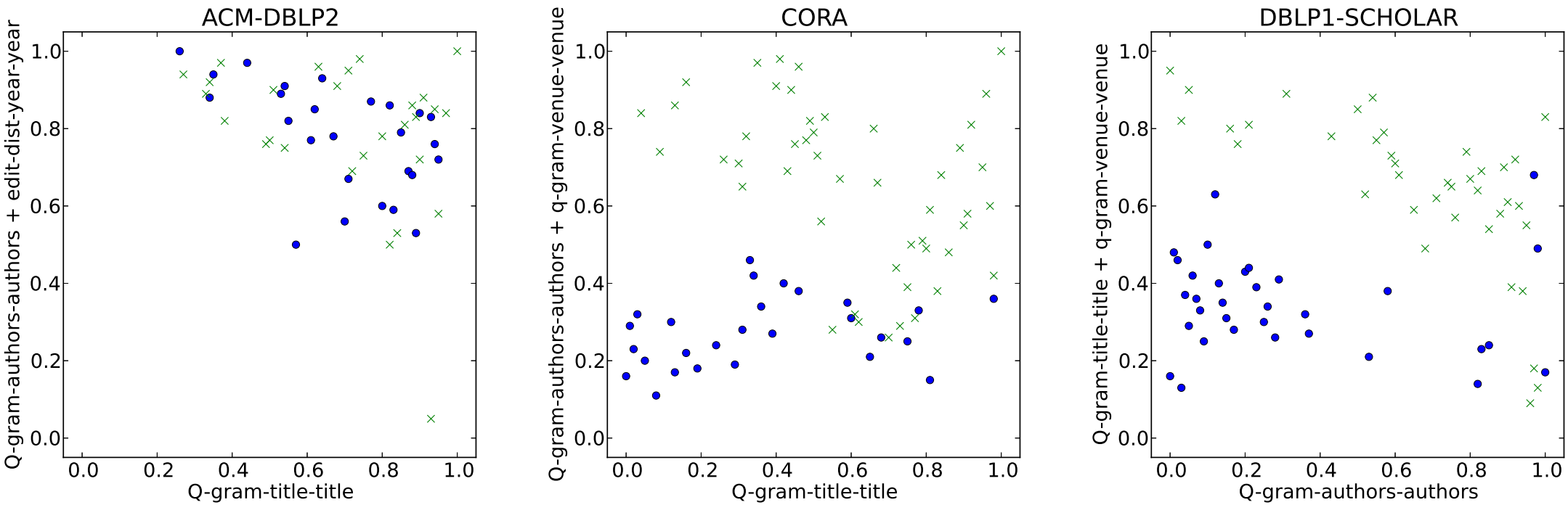
(2) How can we minimize the user's burden of labeling examples?

Active Learning for Entity Resolution

- Active learning is a promising approach for selecting training data.
- The central idea is to reduce labeling efforts through *actively* choosing *informative* or *representative* examples.
- Although successful, most existing active learning methods have some limitations.
  - They are grounded on a **monotonicity assumption** – *a record pair with higher similarity is more likely to represent the same entity than a pair with lower similarity.*
- However:
  - How do we know whether the monotonicity assumption holds on a data set since training data are not available?
  - How can we effectively select training data when the monotonicity assumption does not hold?

Monotonicity Assumption

- The monotonicity assumption is valid in some real-world applications but does not generally hold.
- In the following examples, non-matches with the highest similarity are denoted by **light green crosses**, and matches with the lowest similarity are denoted by **dark blue dots**.



Goal of Our Research

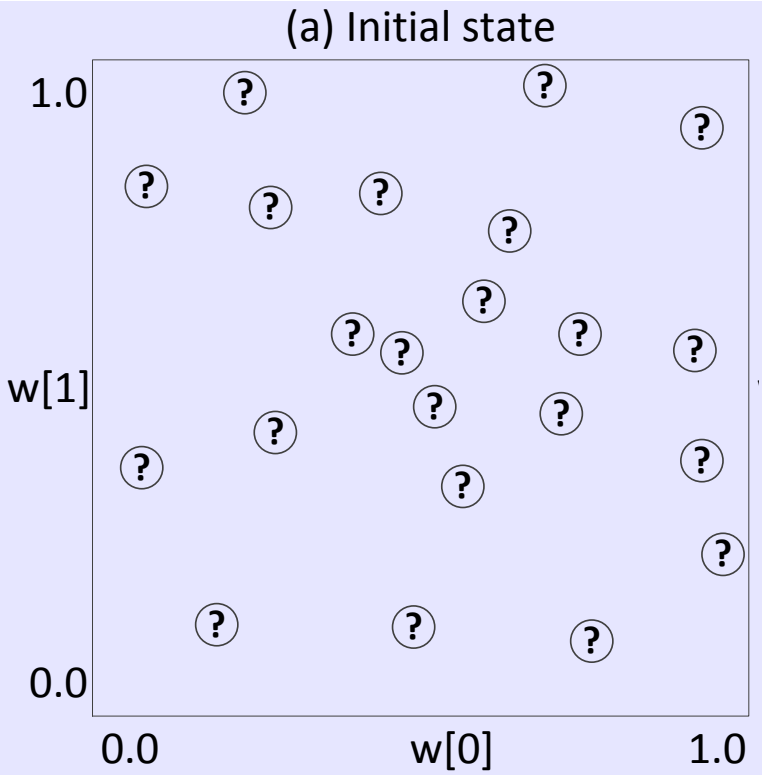
- We develop an interactive training method for efficiently selecting ER training data over large data sets.
  - Can be applied without prior knowledge of the match and non-match distributions of the underlying data sets, i.e., unlike other works, we do not rely on the monotonicity assumption.
  - Incorporates a budget-limited noisy human oracle, which ensures:

    (1) the overall labeling efforts can be controlled at an acceptable level and as specified by the user;

    (2) the accuracy of labeling provided by human experts can be simulated.
- We experimentally evaluate our method on four real-world data sets from different application domains.

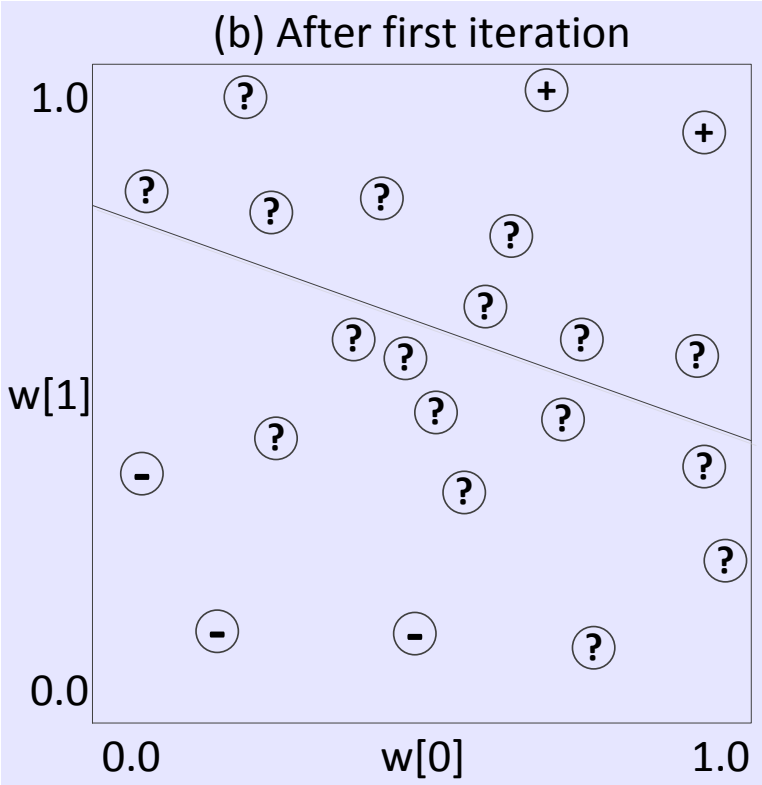
Our Active Learning Method - Main Ideas

- Suppose that we have weight vectors that are generated from pair-wise record comparisons, and the labels of these weight vectors are unknown.



Our Active Learning Method - Main Ideas

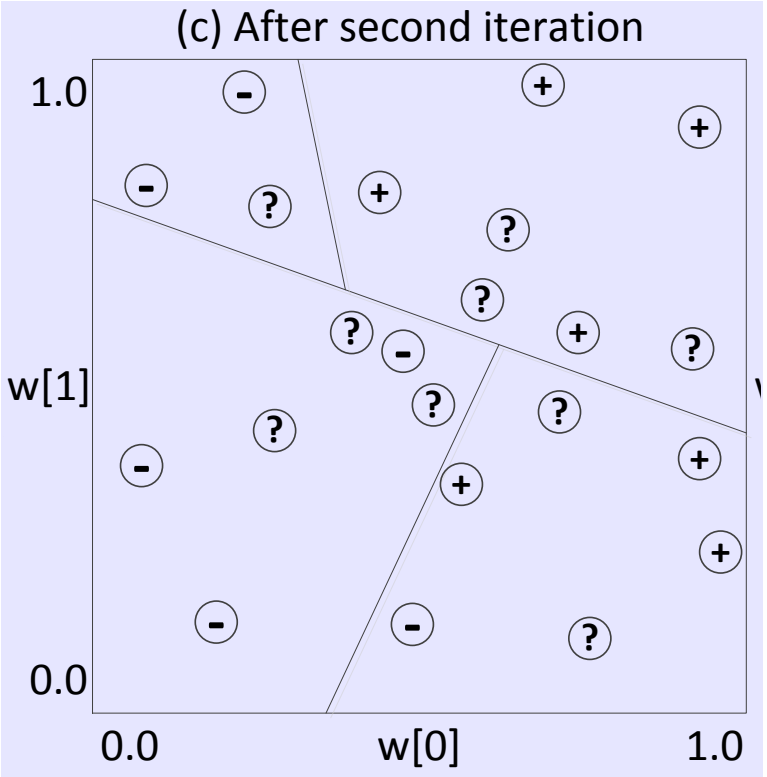
- Some weight vectors are iteratively selected and manually classified, leading to splitting the set of weight vectors into smaller clusters until each cluster is classified as being *pure* or *fuzzy*, i.e., $purity(\mathbf{W}_i) = max\left(\frac{|\mathbf{T}_i^M|}{|\mathbf{T}_i^M \cup \mathbf{T}_i^N|}, \frac{|\mathbf{T}_i^N|}{|\mathbf{T}_i^M \cup \mathbf{T}_i^N|}\right)$.



- $\mathbf{W}_i$ is a set of weight vectors.
- $\mathbf{T}_i^M$ and $\mathbf{T}_i^N$ are the subsets of $\mathbf{W}_i$ which are manually classified by the human oracle into matches and non-matches.

Our Active Learning Method - Main Ideas

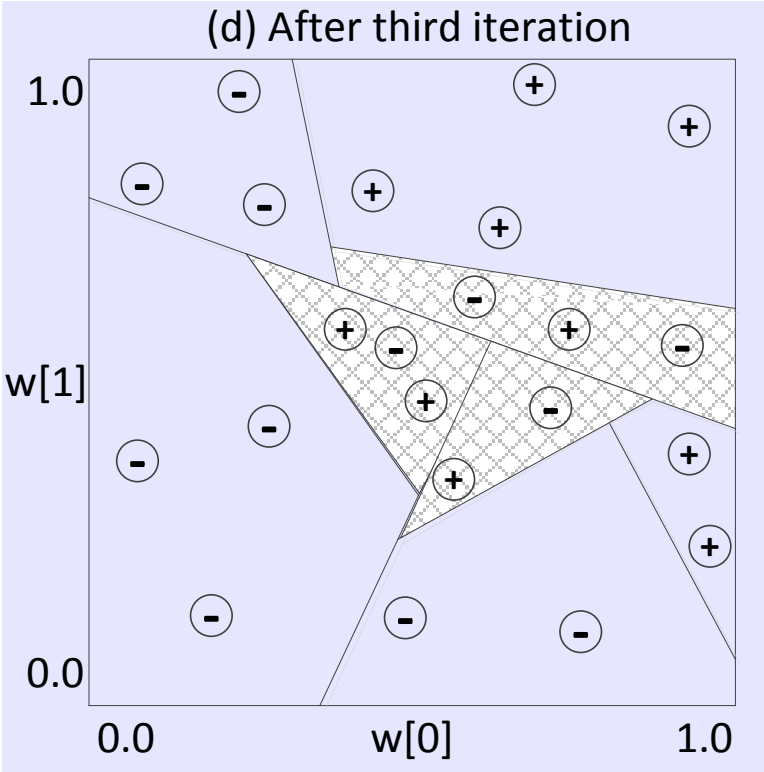
- Some weight vectors are iteratively selected and manually classified, leading to splitting the set of weight vectors into smaller clusters until each cluster is classified as being *pure* or *fuzzy*, i.e., $purity(\mathbf{W}_i) = max\left(\frac{|\mathbf{T}_i^M|}{|\mathbf{T}_i^M \cup \mathbf{T}_i^N|}, \frac{|\mathbf{T}_i^N|}{|\mathbf{T}_i^M \cup \mathbf{T}_i^N|}\right)$.



- $\mathbf{W}_i$ is a set of weight vectors.
- $\mathbf{T}_i^M$ and $\mathbf{T}_i^N$ are the subsets of $\mathbf{W}_i$ which are manually classified by the human oracle into matches and non-matches.

Our Active Learning Method - Main Ideas

- During this process, the training set is interactively constructed by gathering the weight vectors from pure clusters.



Our Active Learning Method - Algorithm

$\mathbf{T}^M = \emptyset, \mathbf{T}^N = \emptyset$ // Initialize training sets as empty
$\mathbf{Q} = [\mathbf{W}]$ // Initialize queue of clusters
$b = 0$ // Initialize number of manually labeled examples
while $\mathbf{Q} \neq \emptyset$ **and** $b \leq b_{tot}$ **do**:
  $\mathbf{W}_i = \mathbf{Q}.pop()$ // Get first cluster from queue
  if $b = 0$ **then**:
    $\mathbf{S}_i = \text{INIT\_SELECT}(\mathbf{W}_i, k)$ // Initial selection of weight vectors
  else:
    $\mathbf{S}_i = \text{MAIN\_SELECT}(\mathbf{W}_i, k)$ // Select informative weight vectors
  $b = b + |\mathbf{S}_i|$ // Update number of manual labeling done so far
  $\mathbf{T}_i^M, \mathbf{T}_i^N, p_i = \text{ORACLE}(\mathbf{S}_i)$ // Manually classify selected weight vectors
  $\mathbf{T}^M = \mathbf{T}^M \cup \mathbf{T}_i^M$; $\mathbf{T}^N = \mathbf{T}^N \cup \mathbf{T}_i^N$; $\mathbf{W}_i = \mathbf{W}_i \setminus (\mathbf{T}_i^M \cup \mathbf{T}_i^N)$
  if $p_i \geq p_{min}$ **then**:
    if $|\mathbf{T}_i^M| > |\mathbf{T}_i^N|$ **then**:
      $\mathbf{T}^M = \mathbf{T}^M \cup \mathbf{W}_i$ // Add whole cluster to match training set
    else:
      $\mathbf{T}^N = \mathbf{T}^N \cup \mathbf{W}_i$ // Add whole to non-match training set
  else if $|\mathbf{W}_i| > c_{min}$ **and** $b \leq b_{tot}$ **then**: // Low purity, split cluster further
    if $\mathbf{T}_i^M \neq \emptyset$ **and** $\mathbf{T}_i^N \neq \emptyset$ **then**:
      $\text{CLASSIFIER}.train(\mathbf{T}_i^M, \mathbf{T}_i^N)$ // Train classifier
      $\mathbf{W}_i^M, \mathbf{W}_i^N = \text{CLASSIFIER}.classify(\mathbf{W}_i)$ // Classify current cluster
      $\mathbf{Q}.append(\mathbf{W}_i^M)$; $\mathbf{Q}.append(\mathbf{W}_i^N)$ // Append new clusters to queue
return $\mathbf{T}^M$ and $\mathbf{T}^N$

Experimental Set-up

- Four data sets:

Data set name(s)	Number of records	Number of unique weight vectors	Class imbalance	Time for pair-wise comparisons
NCVR	224,073 / 224,061	3,495,580	1 : 27	441.6 sec
CORA	1,295	286,141	1 : 16	47.0 sec
DBLP-GS	2,616 / 64,263	8,124,258	1 : 3273	963.1 sec
ACM-DBLP	2,616 / 2,294	687,910	1 : 1785	95.3 sec

- We used the *Febrl* open source record linkage system for the pair-wise linkage step, together with a variety of blocking/indexing and string comparison functions.

- Our proposed active learning approach and the baseline approaches are implemented in Python 2.7.3.

Experimental Tasks

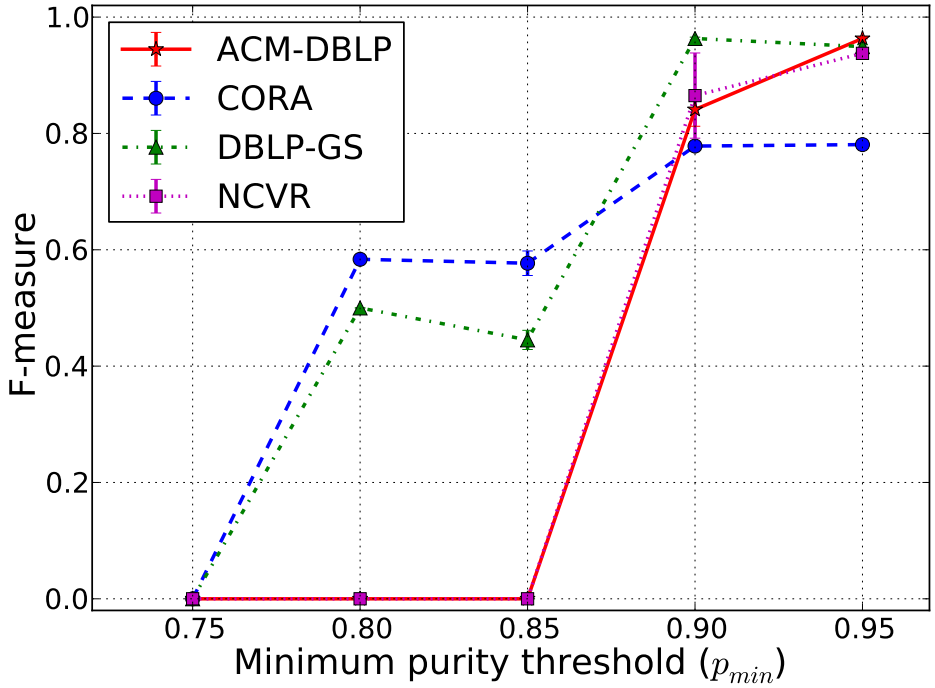
- How do the values for the six main parameters of our approach affect the quality of the classification results?

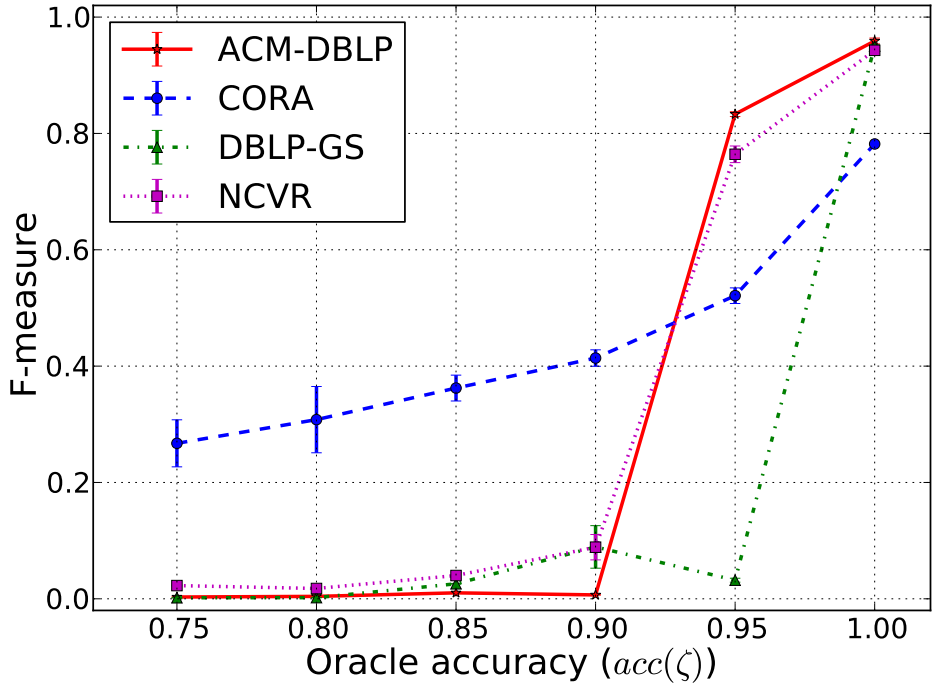
  (1) Minimum purity threshold
  (2) Accuracy of the oracle
  (3) Budget limit
  (4) Number of weight vectors per cluster
  (5) Initial selection function (*Far*, *01* and *Corner*)
  (6) Main selection function (*Ran*, *Far* and *Far-Med*)

- How does our approach perform compared to other classification techniques?
  - Supervised approaches (*decision tree* and *support vector machines with linear and polynomial kernels*)
  - Un-supervised approaches (*automatic k-nearest neighbor clustering*, *k-means clustering*, and *farthest first clustering*)

Experimental Results (1)

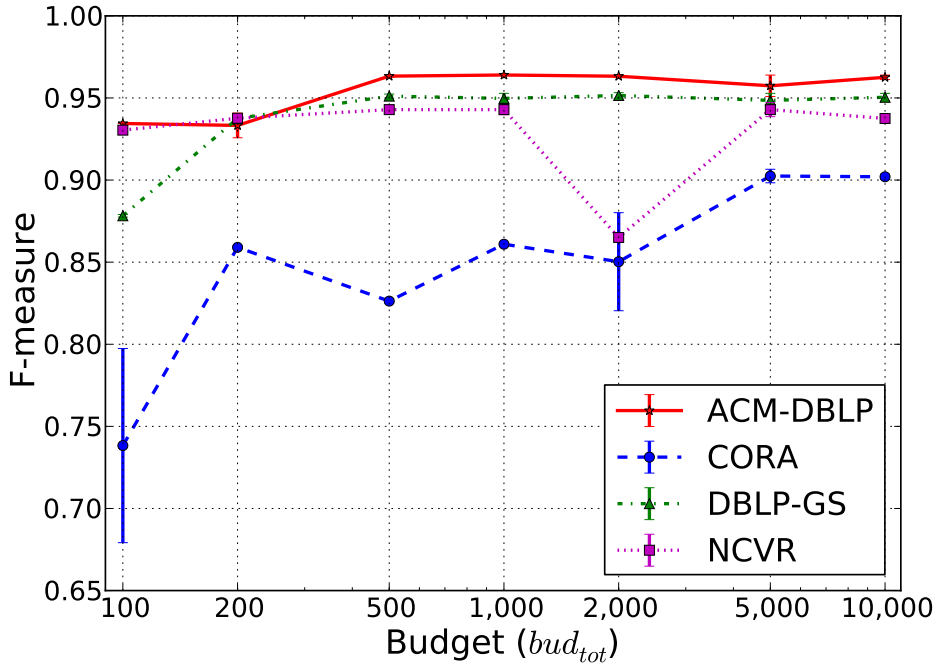
- F-measure increases when the minimum purity threshold increases, since a higher purity of cluster requirement results in more accurately classified clusters.
- F-measure also increases when the accuracy of the oracle increases.

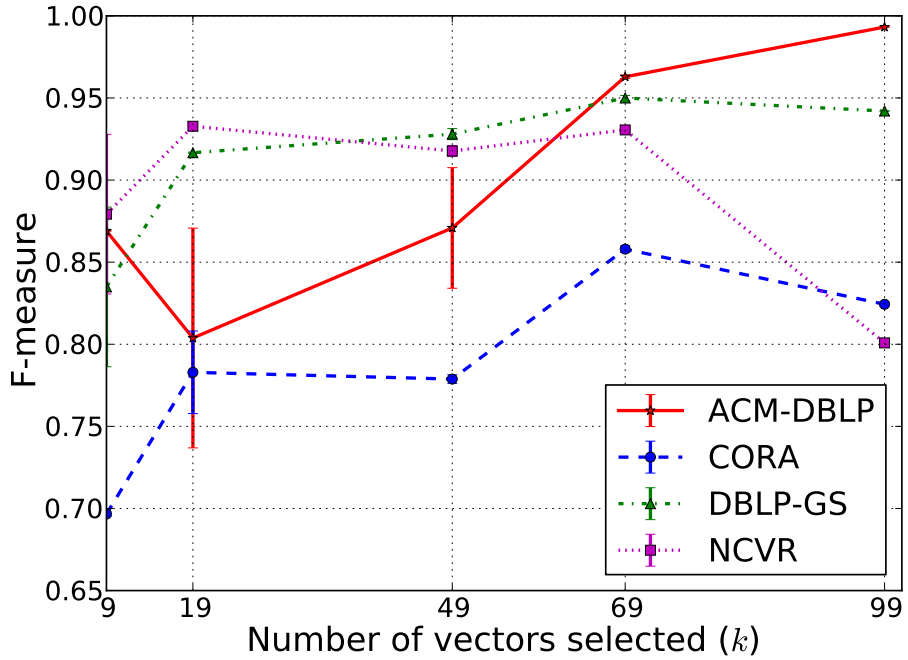




Experimental Results (2)

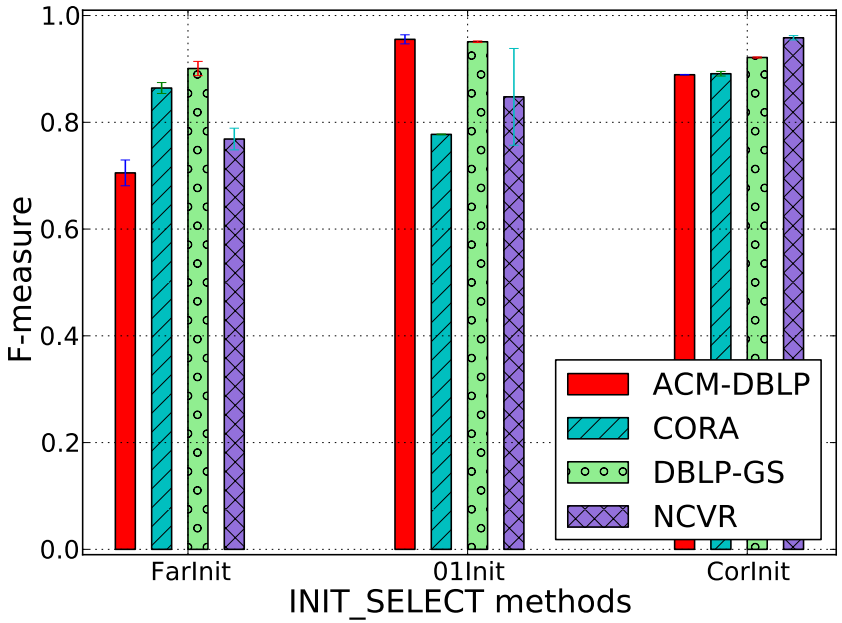
- F-measure increases with larger budgets and more weight vectors selected.
  - Larger budgets allow more vectors to be manually labeled.
  - A larger number of weight vectors selected from each cluster can represent the clusters more effectively.

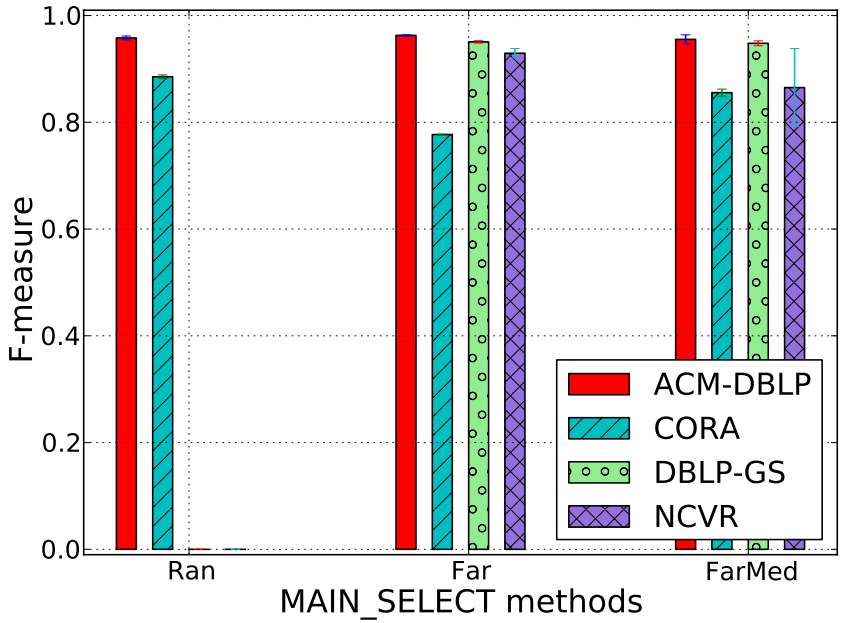




Experimental Results (3)

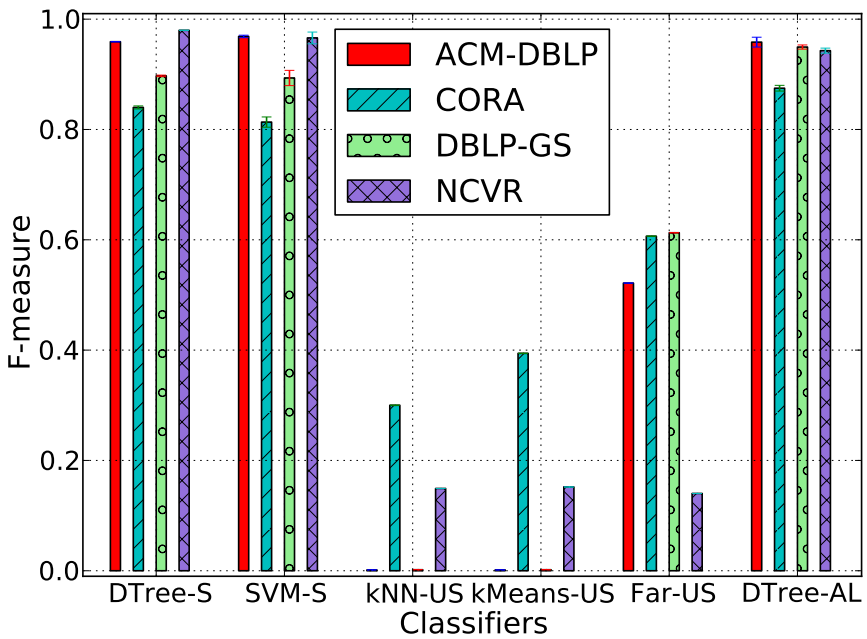
- Selection methods:
  - **Initial selection**: *01* comparatively performs well, though all methods achieve high F-measure on all data sets except *Far* on the ACM-DBLP data set.
  - **Main selection**: *Far* and *Far-Med* perform equally well on all four data sets, while *Ran* does not perform well over two relatively large data sets.

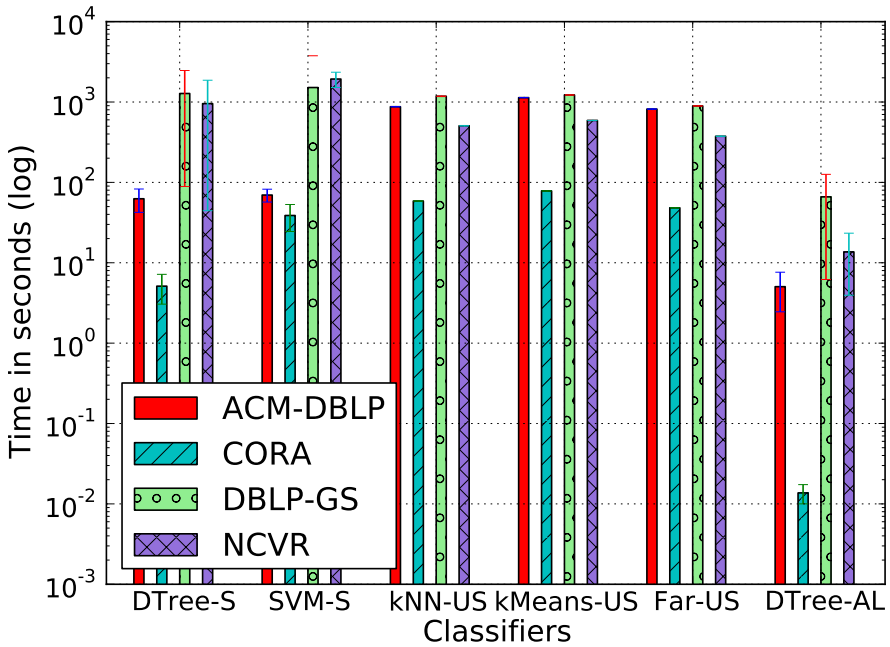




Experimental Results (4)

- Our active learning approach achieves:
  - Significantly higher F-measure results compared to unsupervised approaches, and comparable results to fully supervised approaches;
  - Significantly lower runtime than all other approaches on all four data sets.





Conclusions and Future Work

- We have developed an active learning approach for reducing the labeling costs in ER while achieving high linkage quality results.
- Our experiments validate the efficiency and effectiveness of our approach compared to both existing fully supervised and unsupervised ER classifiers.
- We plan to further study the following issues:
  - How does the ordering of clusters (in the queue) affect the training quality?
  - How can our approach be improved if the accuracy of a human oracle is known?
  - How does the budgeting strategy (the way of using budget) affect the training quality?