Bottom-Up Search and Transfer Learning in SRL

Raymond J. Mooney

University of Texas at Austin

with acknowledgements to

Lily Mihalkova **Tuyen Huynh**

and

Jesse Davis Pedro Domingos Stanley Kok

Complexity of SRL/ILP/MLG

- ILP/SRL/MLG models define very large, complex hypothesis spaces.
- Time complexity is intractable without effective search methods.
- Sample complexity is intractable without effective biases.

Structure Learning

- SRL models consist of two parts:
  - **Structure**: logical formulae, relational model, or graph structure
  - **Parameters**: weights, potentials, or probabilities.
- Parameter learning is easier and much more developed.
- Structure learning is more difficult and less well developed.
  - Structure is frequently specified manually

Bottom-Up Search and Transfer Learning

- Two effective methods for ameliorating time and sample complexity of SRL structure learning:
  - **Bottom-Up Search**: Directly use data to drive the formation of promising hypotheses.
  - **Transfer Learning**: Use knowledge previously acquired in related domains to drive the formation of promising hypotheses.

SRL Approaches

- SLPs (Muggleton, 1996)
- PRMs (Koller, 1999)
- BLPs (Kersting & De Raedt, 2001)
- RMNs (Taskar et al., 2002)
- MLNs (Richardson & Domingos, 2006)

Markov logic networks

Markov Logic Networks (MLNs)

- A logical KB is a set of **hard constraints** on the set of possible worlds
- An MLN is a set of **soft constraints**: When a world violates a formula, it becomes less probable, not impossible
- Give each formula a **weight** (Higher weight $\Rightarrow$ Stronger constraint)

$$P(\text{world}) \propto \exp\left(\sum \text{weights of formulas it satisfies}\right)$$

Sample MLN Clauses

$\text{Parent}(X,Y) \wedge \text{Male}(Y) \Rightarrow \text{Son}(Y,X)$: 10^{10}

$\text{Parent}(X,Y) \wedge \text{Married}(X,Z) \Rightarrow \text{Parent}(Z,Y)$: 10

$\text{LivesWith}(X,Y) \wedge \text{Male}(X) \wedge \text{Female}(Y) \Rightarrow \text{Married}(X,Y)$: 1

MLN Probabilistic Model

- MLN is a **template** for constructing a Markov net
  - Ground literals correspond to nodes
  - Ground clauses correspond to cliques connecting the ground literals in the clause
- Probability of a world x:

$$P(x) = \frac{1}{Z} \exp\left(\sum_i w_i n_i(x) \right)$$

Weight of formula *i*

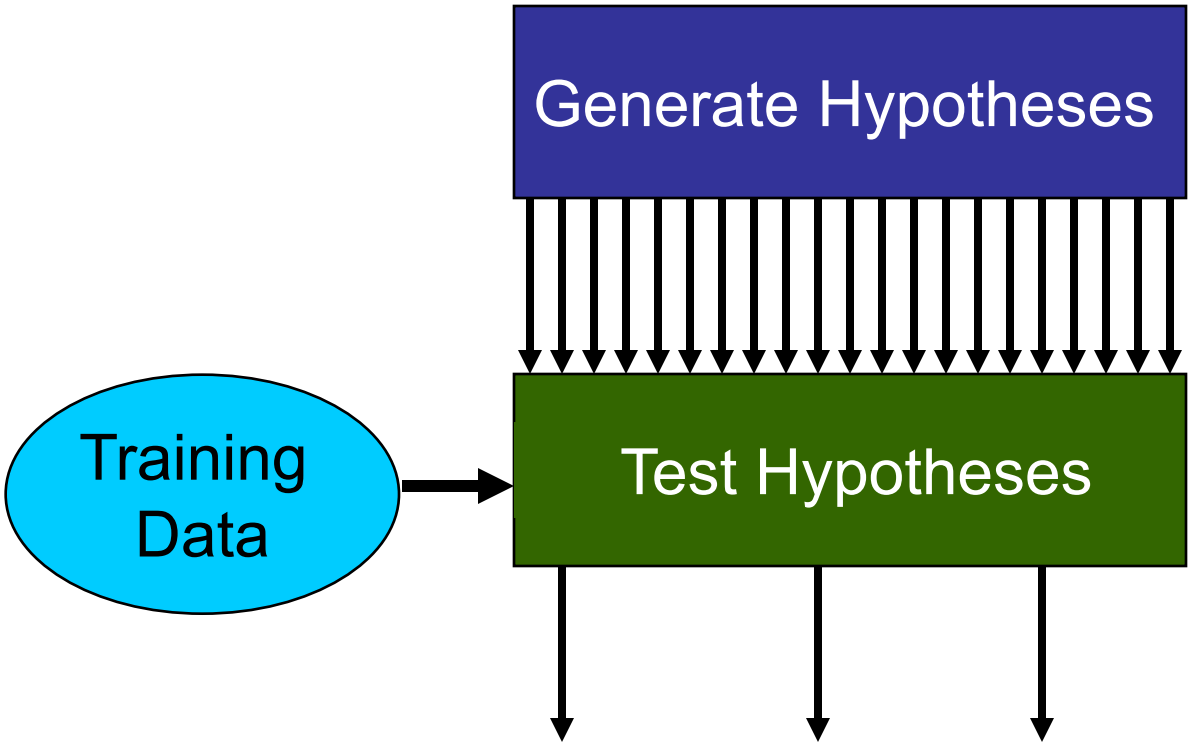
No. of true groundings of formula *i* in *x*

Alchemy

- Open-source package of MLN software provided by UW that includes:
  - Inference algorithms
  - Weight learning algorithms
  - Structure learning algorithm
  - Sample data sets
- All our software uses and extends Alchemy.

BOTTOM-UP SEARCH

Top-Down Search



Top-Down Search in SRL

- SRL typically uses top-down search:

  Start with an empty theory.

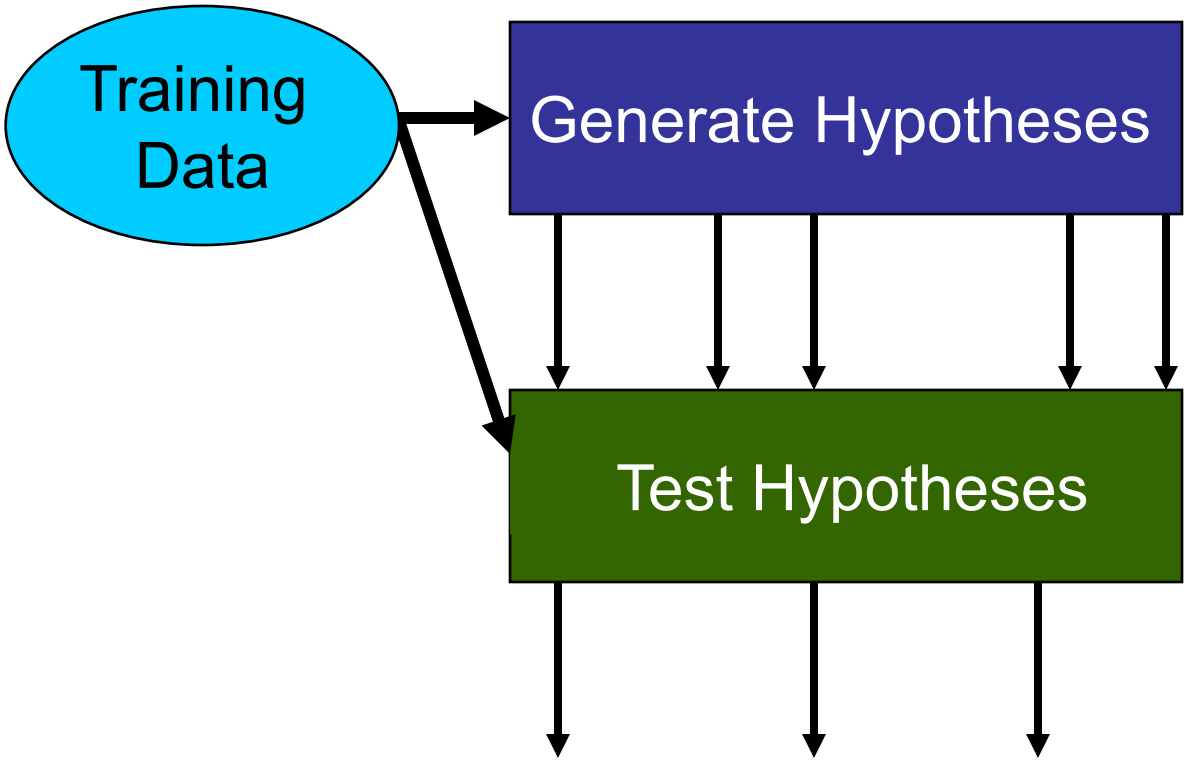
  Repeat until further refinements fail to improve fit:

  – Generate all possible refinements of current theory (e.g. adding all possible single literals to a clause).

  – Test each refined theory on the training data and pick ones that best improve fit.

- Results in a huge branching factor.
- Use greedy or beam search to control time complexity, subject to local maxima.

Bottom-Up Search



Bottom-Up Search

- Use data to directly drive the formation of a limited set of more promising hypotheses.
- Also known as:
  - Data Driven
  - Specific to General

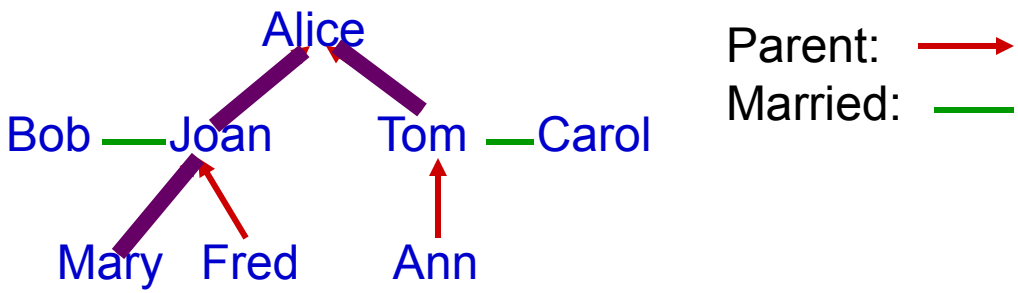
History of Bottom-Up Search in ILP

- Inverse resolution and CIGOL (Muggleton & Buntine, 1988)
- LGG (Plotkin, 1970) and GOLEM (Muggleton & Feng, 1990)

Relational Path Finding
(Richards & Mooney, 1992)

- Learn definite clauses based on finding paths of relations connecting the arguments of positive examples of the target predicate.

Uncle(Tom, Mary)



Parent(Joan,Mary) $\wedge$ Parent(Alice,Joan) $\wedge$ Parent(Alice,Tom) $\Rightarrow$ Uncle(Tom,Mary)

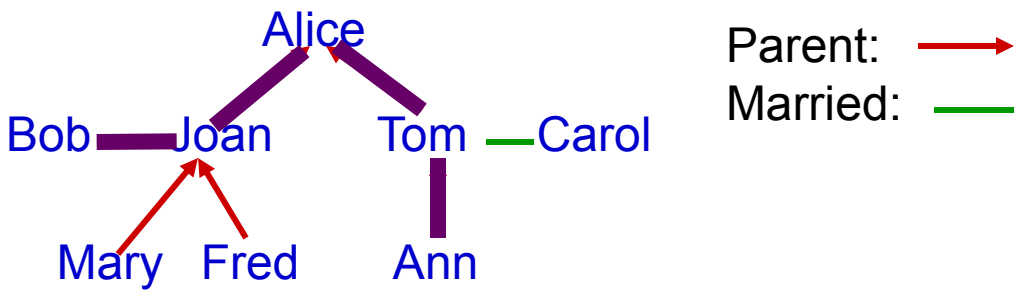
Parent(x,y) $\wedge$ Parent(z,x) $\wedge$ Parent(z,w) $\Rightarrow$ Uncle(w,y)

Parent(x,y) $\wedge$ Parent(z,x) $\wedge$ Parent(z,w) $\wedge$ Male(w) $\Rightarrow$ Uncle(w,y)

Relational Path Finding
(Richards & Mooney, 1992)

- Learn definite clauses based on finding paths of relations connecting the arguments of positive examples of the target predicate.

Uncle(Bob,Ann)



Parent(Tom,Ann) $\wedge$ Parent(Alice,Tom) $\wedge$ Parent(Alice,Joan) $\wedge$ Married(Bob,Joan) $\Rightarrow$ Uncle(Tom,Mary)

Parent(x,y) $\wedge$ Parent(z,x) $\wedge$ Parent(z,w) $\wedge$ Married(v,w) $\Rightarrow$ Uncle(v,y)

Parent(x,y) $\wedge$ Parent(z,x) $\wedge$ Parent(z,w) $\wedge$ Married(v,w) $\wedge$ Male(v) $\Rightarrow$ Uncle(w,y)

Integrating Top-Down and Bottom-up in ILP: Hybrid Methods

- CHILLIN (Zelle, Mooney, and Konvisser, 1994)
- PROGOL (Muggleton, 1995) and ALEPH (Srinivasan, 2001)

Bottom-Up Search in SRL

- Not much use of bottom-up techniques in structure-leaning methods for SRL.
- Most algorithms influenced by Bayes net and Markov net structure learning algorithms that are primarily top-down.
- Many (American) researchers in SRL are not sufficiently familiar with previous relational learning work in ILP.

BUSL: Bottom-Up Structure Learner
(Mihalkova and Mooney, 2007)

- Bottom-up (actually hybrid) structure learning algorithm for MLNs.
- Exploits partial propositionalization driven by relational path-finding.
- Uses a Markov-net structure learner to build a Markov net template that constrains clause construction.

BUSL General Overview

- For each predicate P in the domain, do:
  - Construct a set of template nodes and use them to partially propositionalize the data.
  - Construct a Markov network template from the propositional data.
  - Form candidate clauses based on this template.
- Evaluate all candidate clauses on training data and keep the best ones

Template Nodes

- Contain conjunctions of one or more variablized literals that serve as clause building blocks.
- Constructed by looking for groups of true constant-sharing ground literals in the data and variablize them.
- Can be viewed as partial relational paths in the data.

Propositionalizing Data

Relational Data:

Actor(brando) Actor(pacino) Director(coppola) Actor(eastwood) Director(eastwood)
WorkedFor(brando, coppola) WorkedFor(pacino, coppola) WorkedFor(eastwood, eastwood)
Movie(godFather, brando) Movie(godFather, coppola)
Movie(godFather, pacino) Movie(millionDollar,eastwood)

Current Predicate

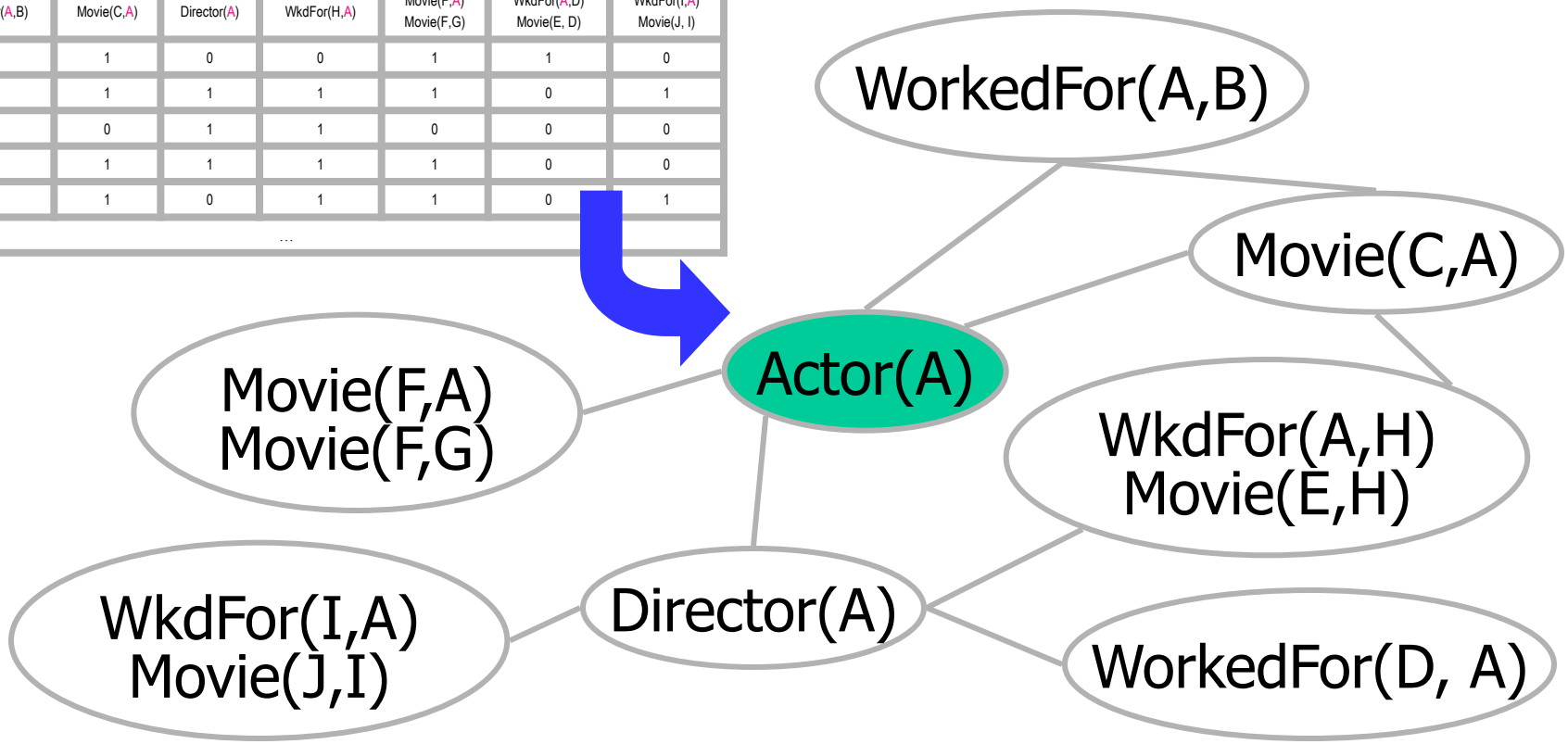
Template Nodes

Actor(A)	WkdFor(A,B)	Movie(C,A)	Director(A)	WkdFor(H,A)	Movie(F,A) Movie(F,G)	WkdFor(A,D) Movie(E, D)	WkdFor(I,A) Movie(J, I)
1	1	1	0	0	1	1	0
0	0	1	1	1	1	0	1
1	1	1	0	0	1	1	0
1	1	1	1	1	0	1	1

Constructing the Markov Net Template

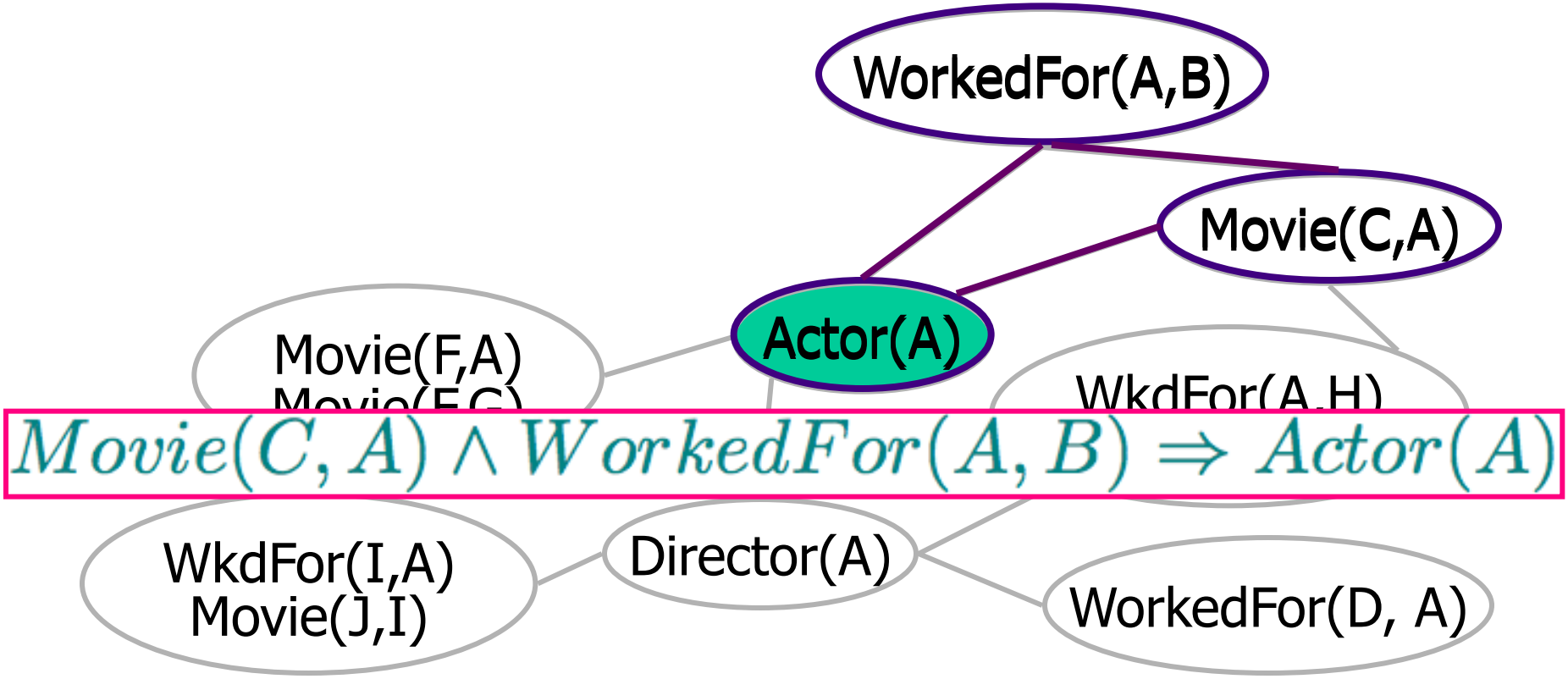
- Use an existing Markov network structure learner (Bromberg et al. 2006) to produce the Markov network template

Actor(A)	WkdFor(A,B)	Movie(C,A)	Director(A)	WkdFor(H,A)	Movie(F,A) Movie(F,G)	WkdFor(A,D) Movie(E, D)	WkdFor(I,A) Movie(J, I)
1	1	1	0	0	1	1	0
0	0	1	1	1	1	0	1
1	1	0	1	1	0	0	0
1	1	1	1	1	1	0	0
0	1	1	0	1	1	0	1
…							



Forming Clause Candidates

- Consider only candidates that comply with the cliques in the Markov network template



BUSL
Experiments

Data Sets

- UW-CSE
  - Data about members of the UW CSE department (Richardson & Domingos, 2006)
  - Predicates include Professor, Student, AdvisedBy, TaughtBy, Publication, etc.
- IMDB
  - Data about 20 movies
  - Predicates include Actor, Director, Movie, WorkedFor, Genre, etc.
- WebKB
  - Entity relations from the original WebKB domain (Craven et al. 1998)
  - Predicates include Faculty, Student, Project, CourseTA, etc.

Data Set Statistics

Data is organized as ***mega-examples***

- Each mega-example contains information about a group of related entities.
- Mega-examples are independent and disconnected from each other.

Data Set	# Mega-Examples	# Constants	# Types	# Predicates	# True Gliterals	Total # Gliterals
IMDB	5	316	4	10	1,540	32,615
UW-CSE	5	1,323	9	15	2,673	678,899
WebKB	4	1,700	3	6	2,065	688,193

Methodology: Learning & Testing

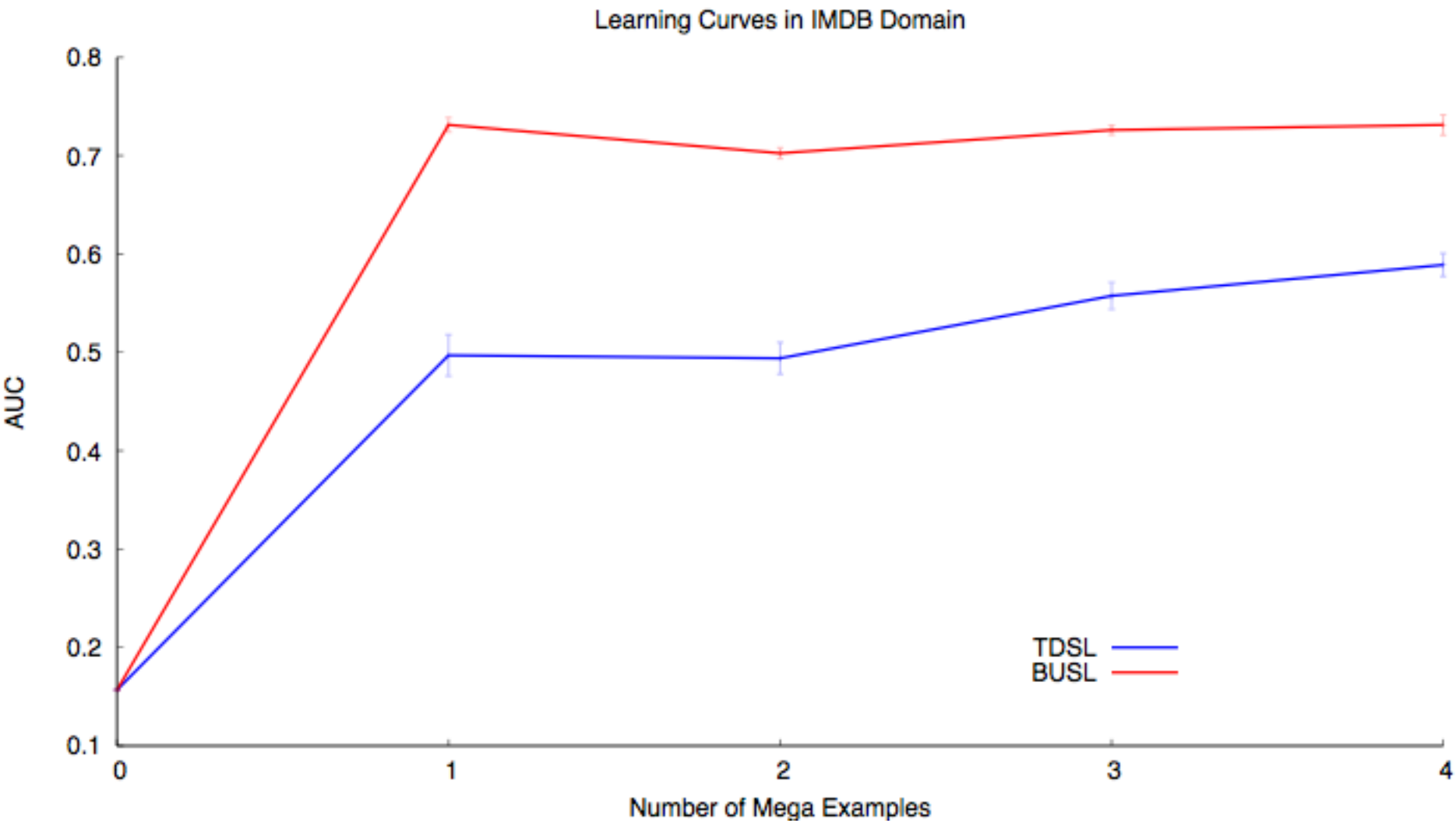
- Generated learning curves using leave one mega-example out.
  - Each run keeps one mega-example for testing and trains on the remaining ones, provided one by one
  - Curves are averaged over all runs
- Evaluated learned MLN by performing inference for the literals of each predicate in turn, providing the rest as evidence, and averaging the results.
- Compared BUSL to top-down MLN structure learner (TDSL) (Kok & Domingos, 2005)

Methodology: Metrics
Kok & Domingos (2005)

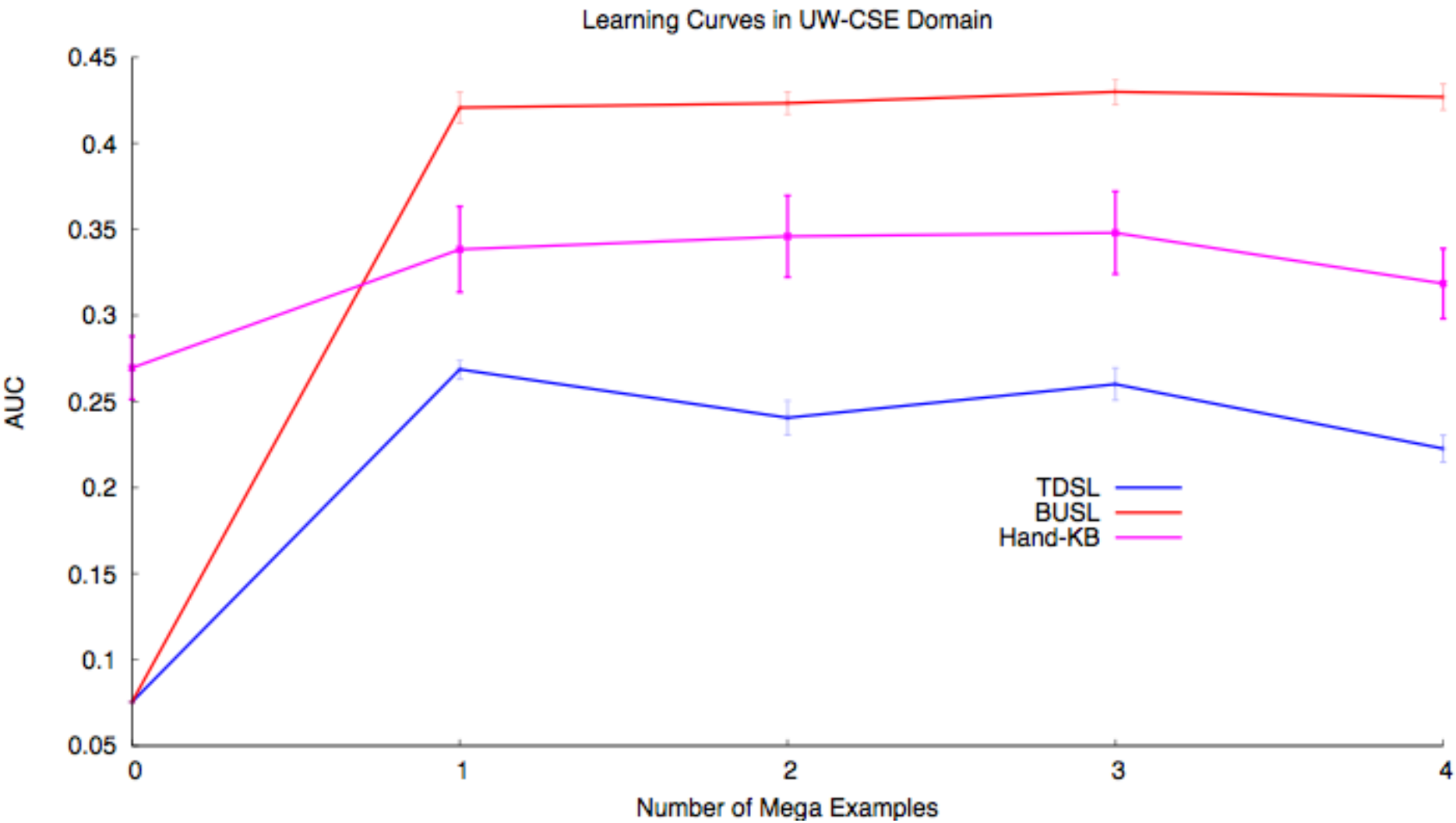
- CLL: Conditional Log Likelihood
  - The log of the probability predicted by the model that a literal has the correct truth value given in the data.
  - Averaged over all test literals.

- AUC-PR: Area under the precision recall curve
  - Produce a PR curve by varying a probability threshold
  - Find area under that curve

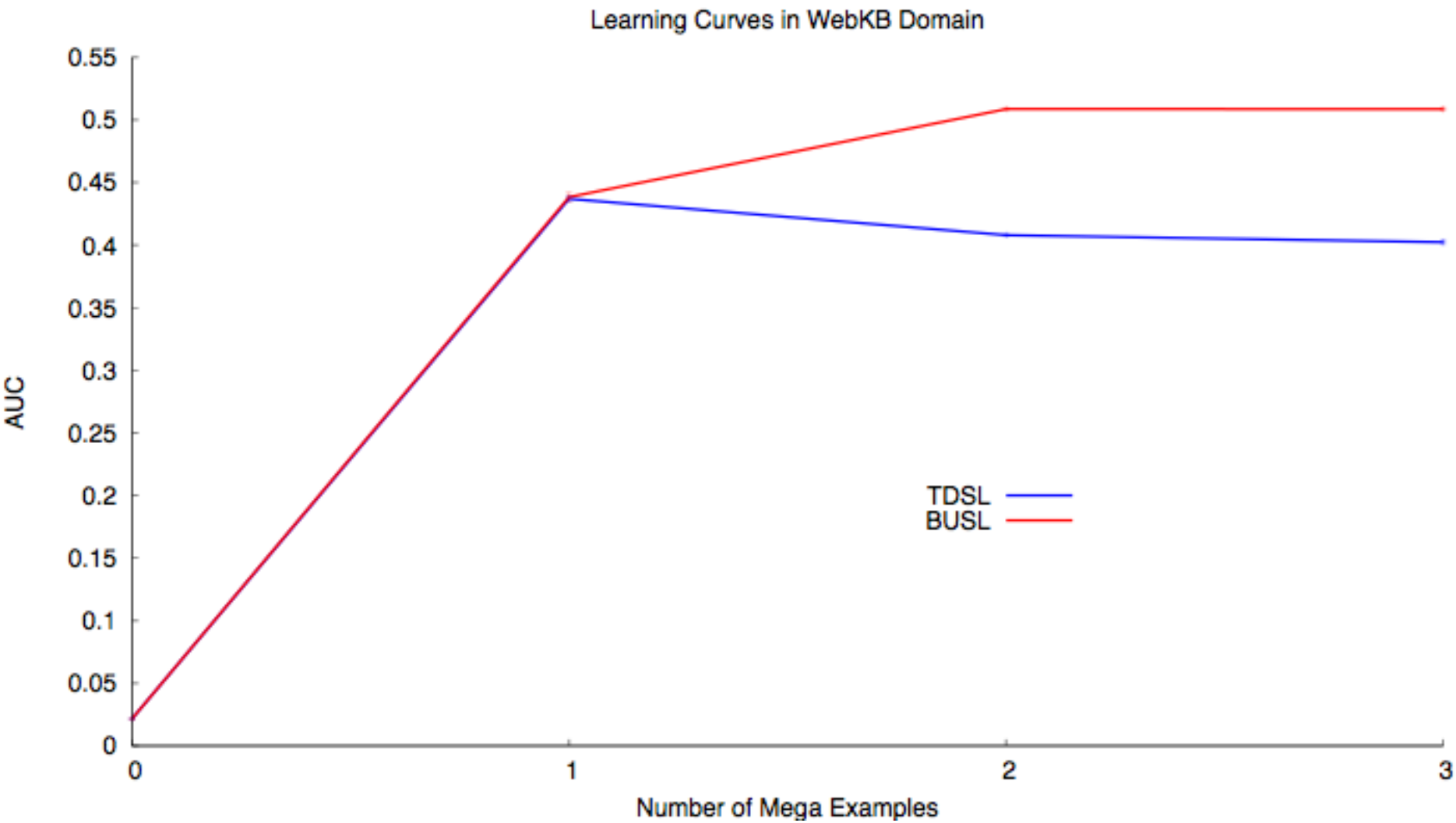
Results: AUC-PR in IMDB



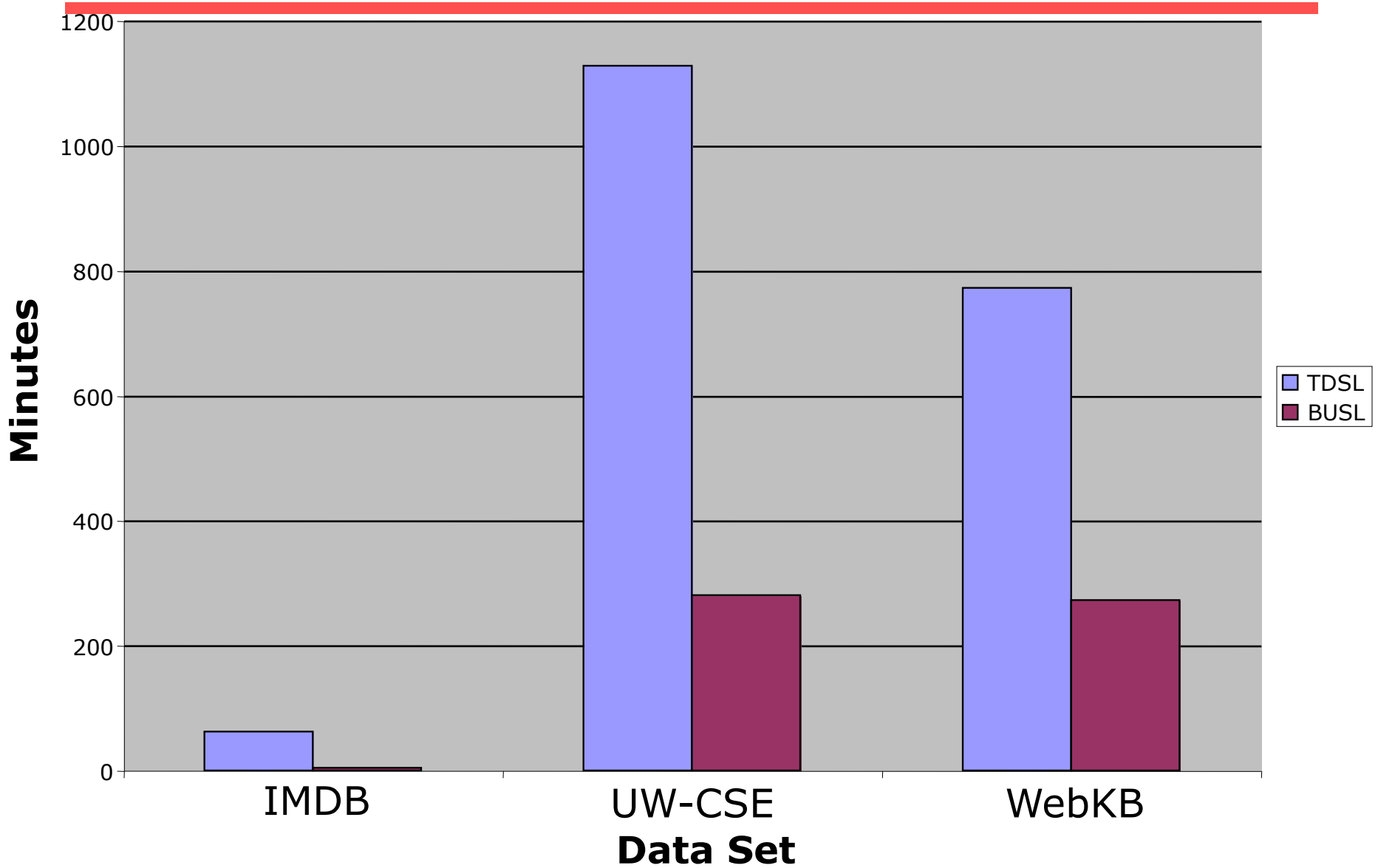
Results: AUC-PR in UW-CSE



Results: AUC-PR in WebKB



Results: Average Training Time



Discriminative MLN Learning with Hybrid ILP Methods (Huynh & Mooney, 2008)

- Discriminative learning assumes a particular target predicate is to be inferred given information using background predicates.
- Existing non-discriminative MLN structure learners did very poorly on several ILP benchmark problems in molecular biology.
- Use existing hybrid discriminative ILP methods (ALEPH) to learn candidate MLN clauses.

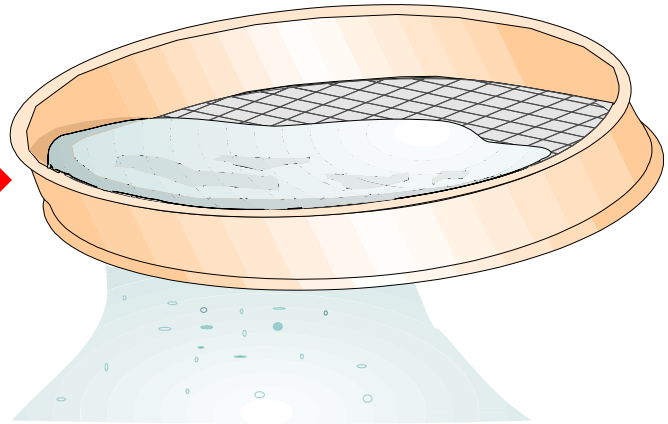
General Approach

Step 1

ILP
Clause Learner

Discriminative structure learning
(Generating candidate clauses)

Step 2



Discriminative weight learning
(Selecting good clauses)

Discriminative Structure Learning

- Goal: Learn the relations between background and target predicates.
- Solution: Use a variant of ALEPH (Srinivasan, 2001), called ALEPH++, to produce a larger set of candidate clauses.

Discriminative Weight Learning

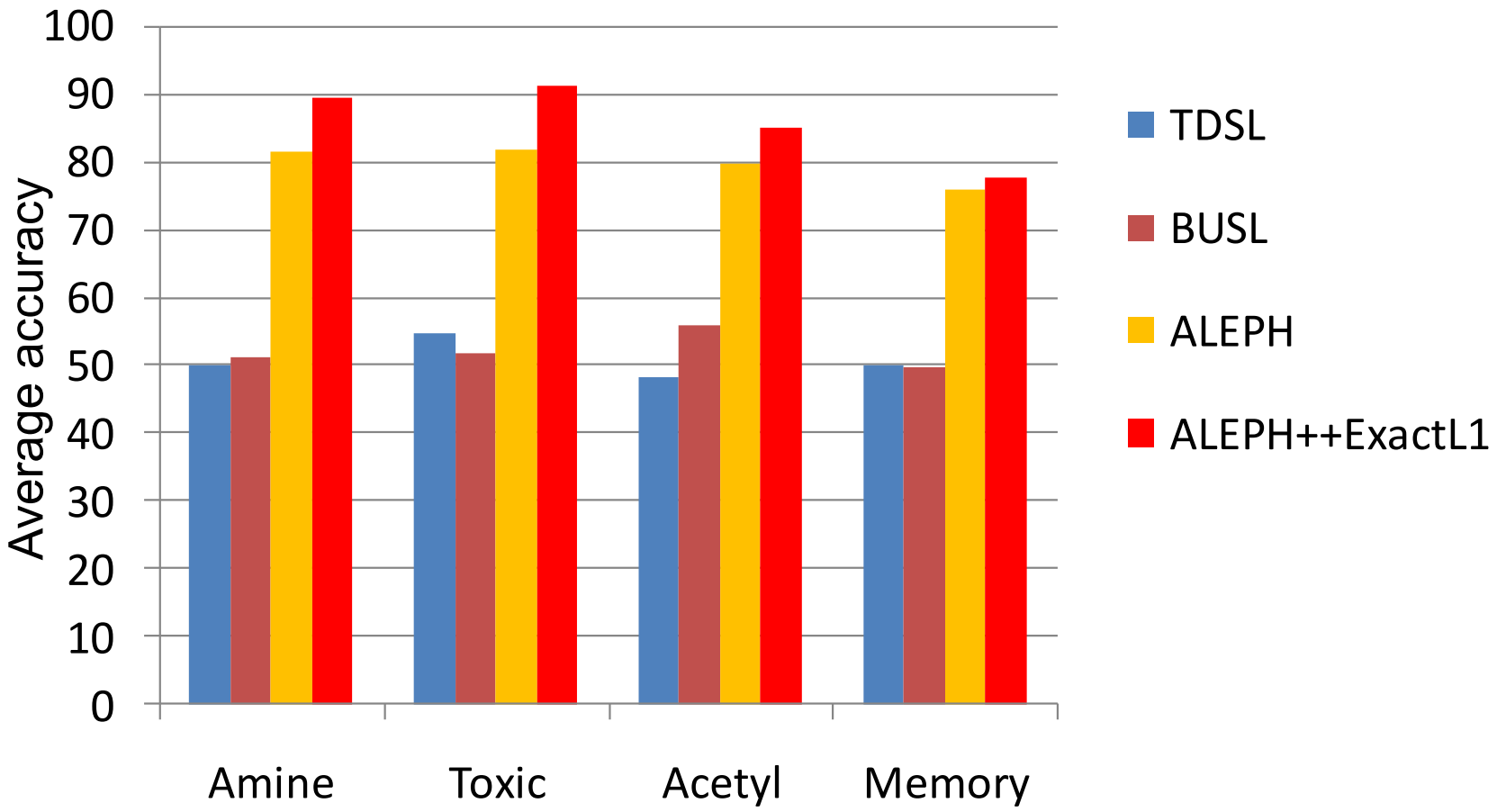
- Goal: Learn weights for clauses that allow accurate prediction of the target predicate.
- Solution: Maximize CLL of target predicate on training data.
  - Use exact inference for non-recursive clauses instead of approximate inference
  - Use L_1-regularization instead of L_2-regularization to encourage zero-weight clauses

Data Sets

- ILP benchmark data sets comparing drugs for Alzheimer's disease on four biochemical properties:
  - Inhibition of amine re-uptake
  - Low toxicity
  - High acetyl cholinesterase inhibition
  - Good reversal of scopolamine-induced memory

Data set	# Examples	% Pos. example	#Predicates
Alzheimer amine	686	50%	30
Alzheimer toxic	886	50%	30
Alzheimer acetyl	1326	50%	30
Alzheimer memory	642	50%	30

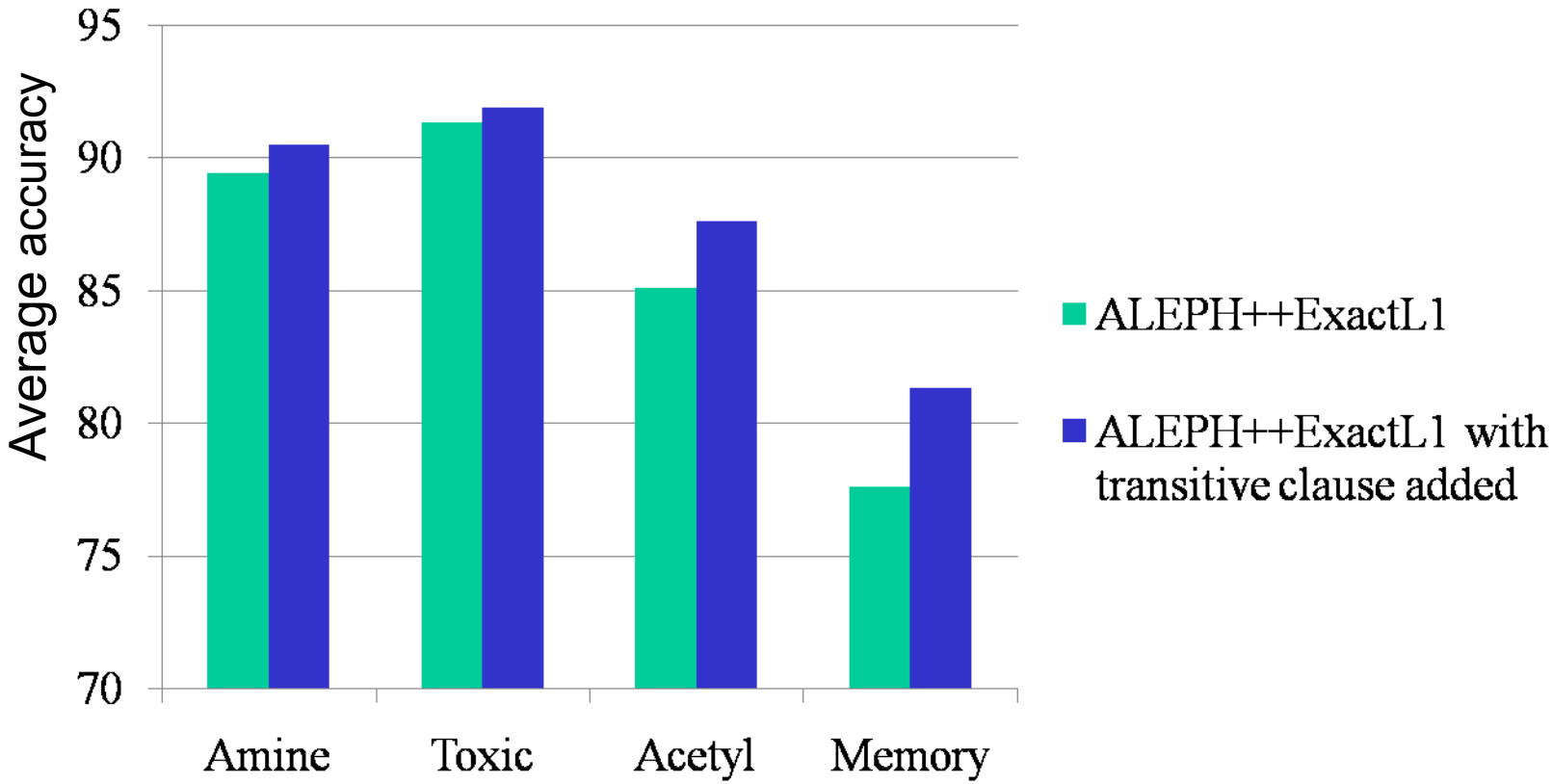
Results: Predictive Accuracy



Results: Adding Collective Inference

Add an ∞-weight transitive clause to learned MLNs

less_toxic(a,b) ∧ less_toxic(b,c) ⇒ less_toxic(a,c).

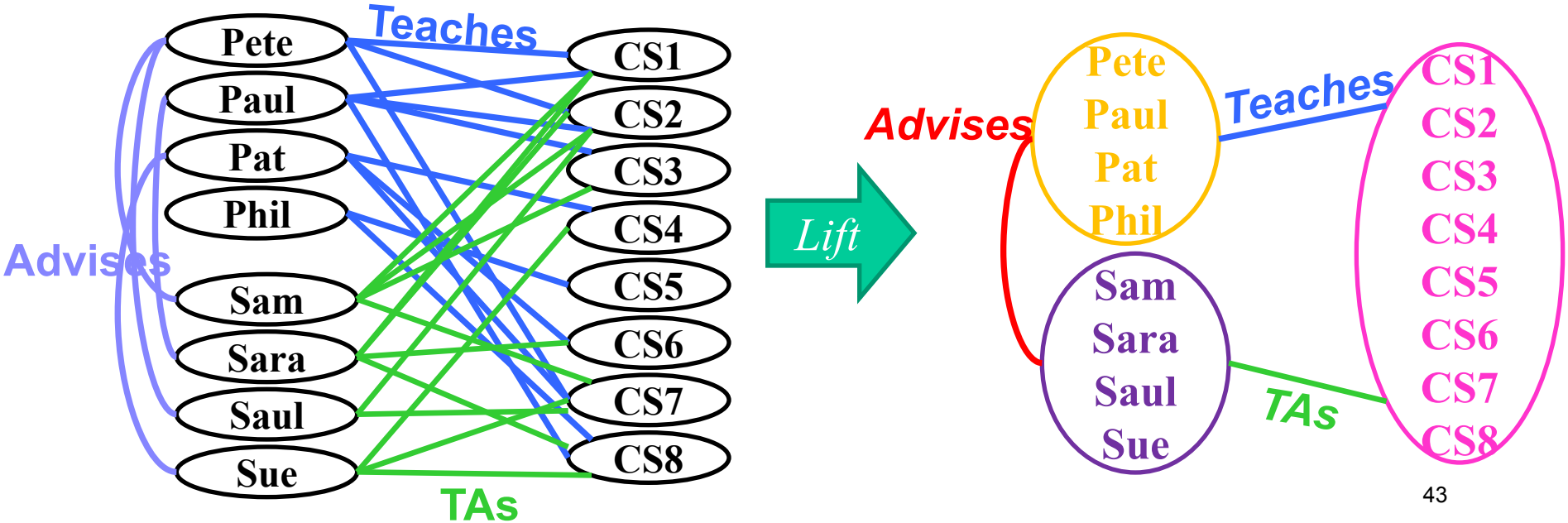


Learning via Hypergraph Lifting (LHL)
(Kok & Domingos, 2009)

- New bottom-up approach to learning MLN structure.
- Fully exploits a non-discriminative version of relational pathfinding.
- Current best structure learner for MLNs.
- See the poster here!

LHL = Clustering + Relational Pathfinding

- LHL "lifts" hypergraph into more compact rep.
  - Jointly clusters nodes into higher-level concepts
  - Clusters hyperedges
- Traces paths in lifted hypergraph



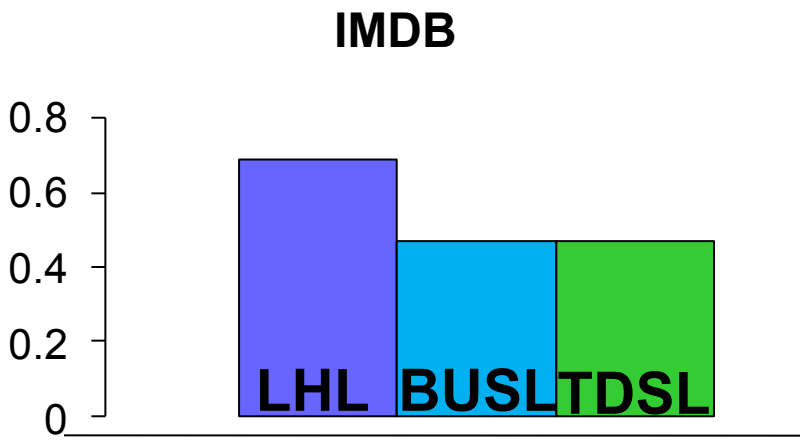
LHL Algorithm

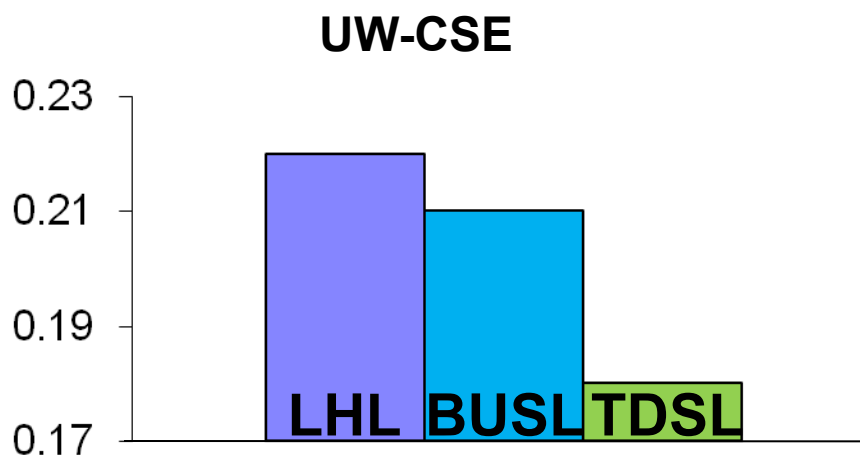
- LHL has **three** components:
  - **LiftGraph**: Lifts hypergraph by clustering
  - **FindPaths**: Finds paths in lifted hypergraph
  - **CreateMLN**: Creates clauses from paths, and adds good ones to MLN

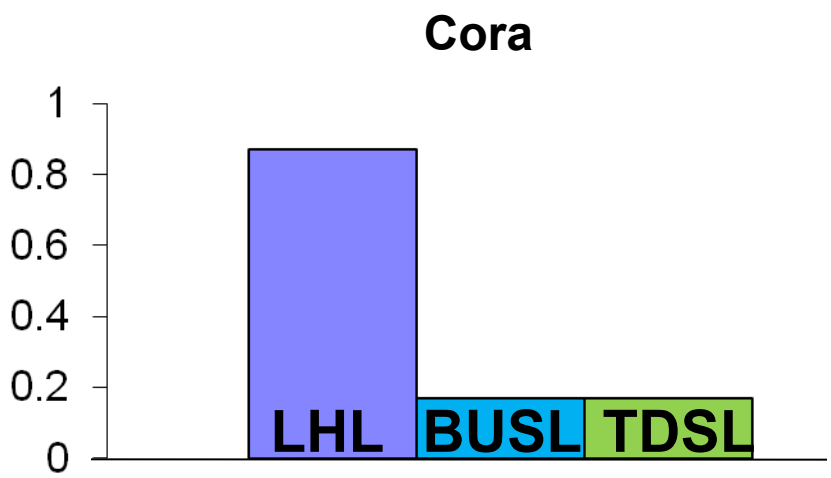
Additional Dataset

- **Cora**
  - Citations to computer science papers
  - Papers, authors, titles, etc., and their relationships
  - **687,422** ground atoms; **42,558** true ones

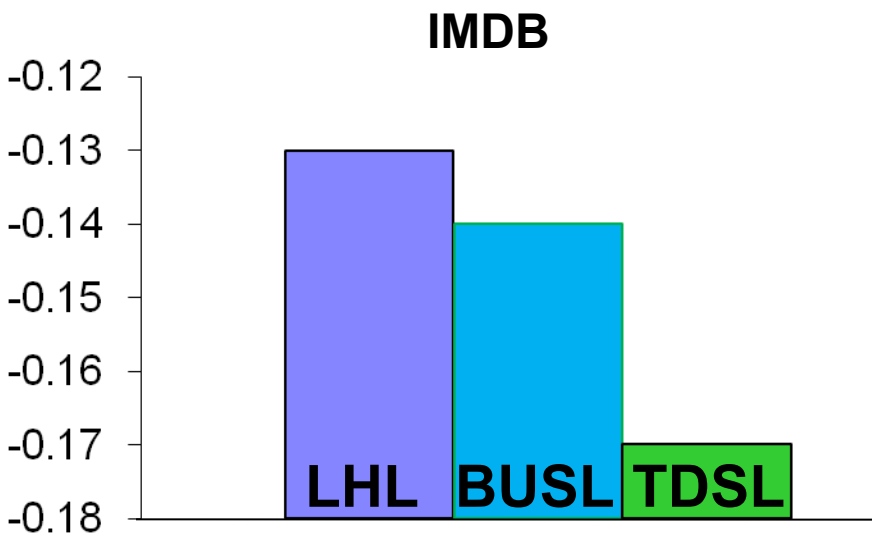
LHL vs. BUSL vs. MSL
Area under Prec-Recall Curve

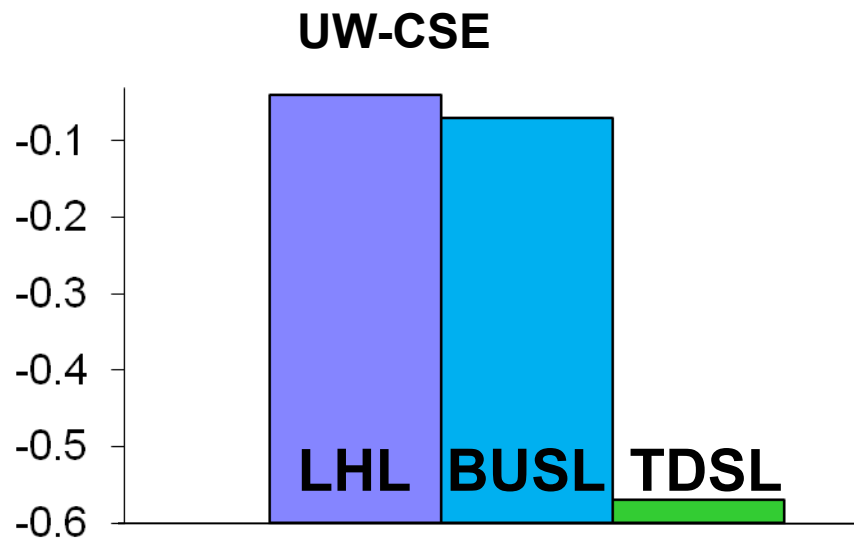


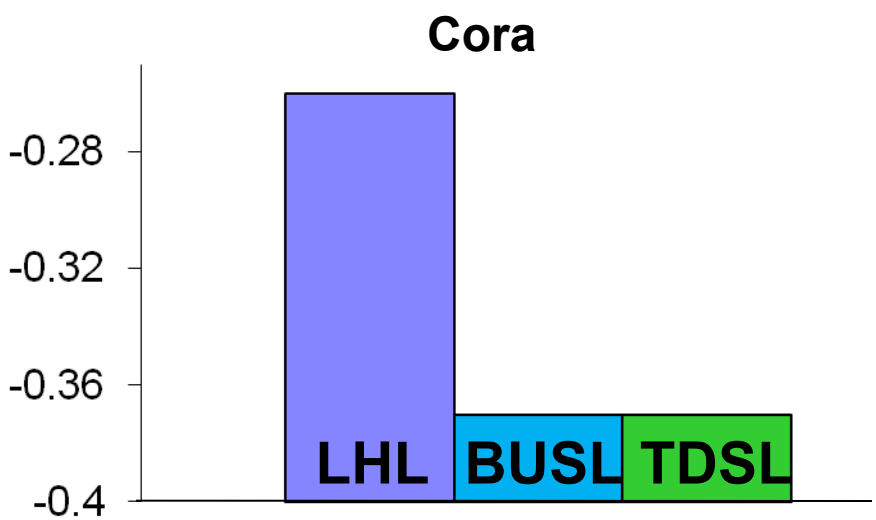




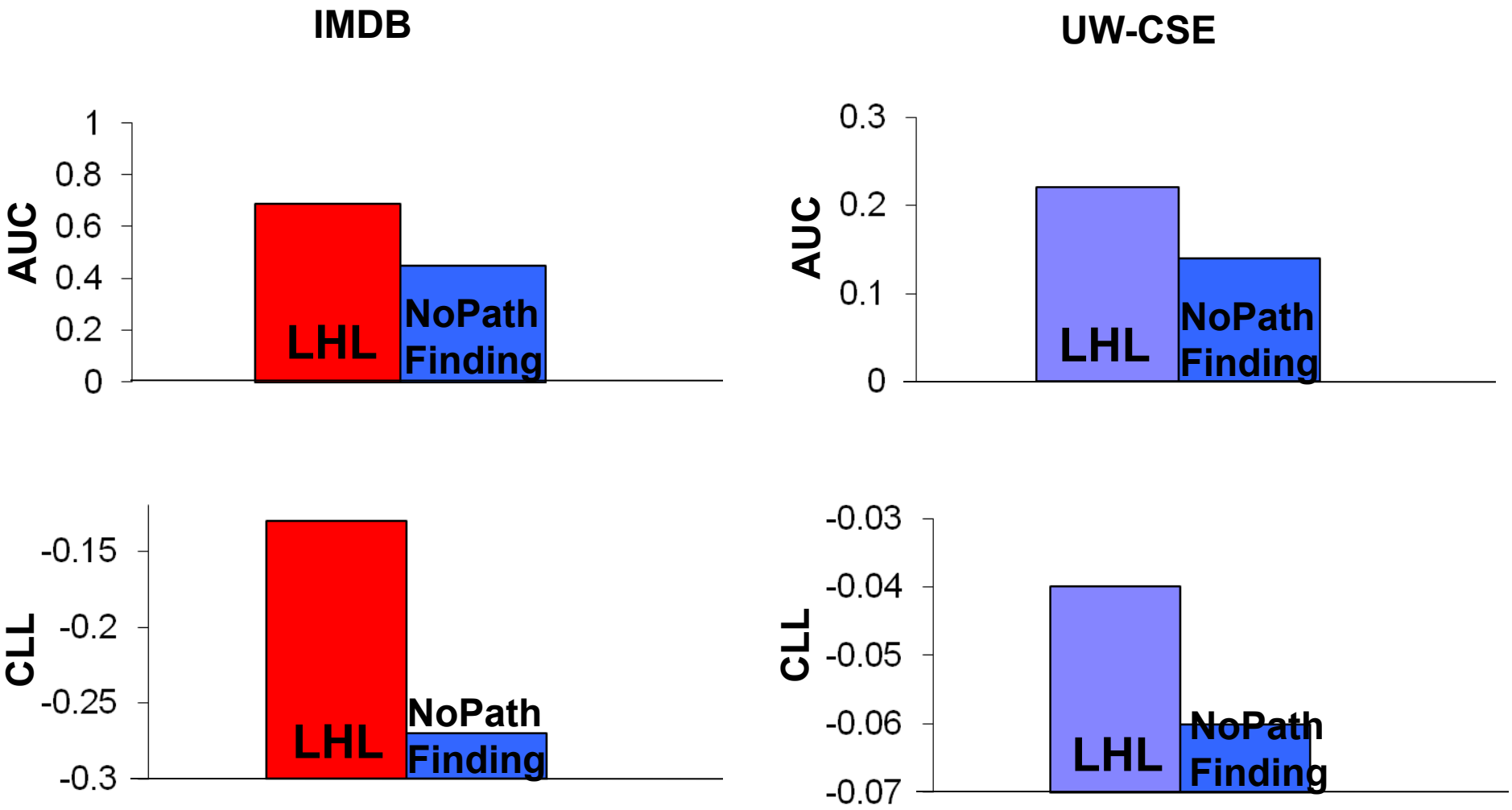
LHL vs. BUSL vs. MSL
Conditional Log-likelihood







LHL vs. NoPathFinding



TRANSFER LEARNING

Transfer Learning

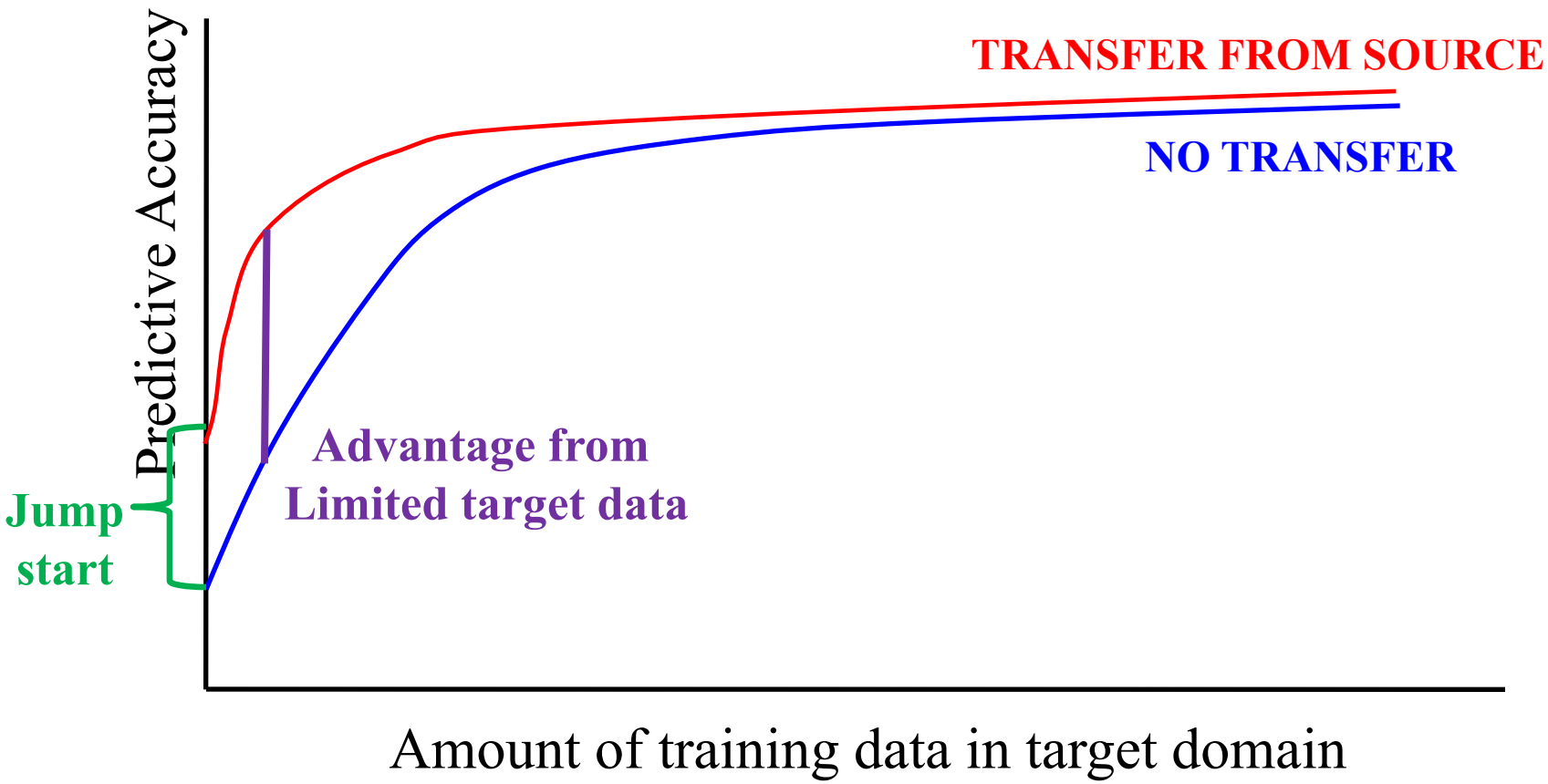
- Most machine learning methods learn each new task from scratch, failing to utilize previously learned knowledge.
- Transfer learning concerns using knowledge acquired in a previous **source** task to facilitate learning in a related **target** task.

Transfer Learning Advantages

- Usually assume significant training data was available in the source domain but limited training data is available in the target domain.
- By exploiting knowledge from the source, learning in the target can be:
  - **More accurate**: Learned knowledge makes better predictions.
  - **Faster**: Training time is reduced.

Transfer Learning Curves

- Transfer learning increases accuracy in the target domain.



Recent Work on Transfer Learning

- Recent DARPA program on Transfer Learning (TL) has led to significant recent research in the area.
- Some work focuses on feature-vector classification.
  - Hierarchical Bayes (Yu *et al*., 2005; Lawrence & Platt, 2004)
  - Informative Bayesian Priors (Raina *et al*., 2005)
  - Boosting for transfer learning (Dai *et al*., 2007)
  - Structural Correspondence Learning (Blitzer *et al*., 2007)
- Some work focuses on Reinforcement Learning
  - Value-function transfer (Taylor & Stone, 2005; 2007)
  - Advice-based policy transfer (Torrey *et al*., 2005; 2007)

Prior Work in Transfer and Relational Learning

This page is intentionally left blank

TL and SRL and I.I.D.

Standard Machine Learning assumes examples are:

Independent and Identically Distributed

SRL breaks the assumption that examples are independent (requires "collective classification")

TL breaks the assumption that test examples are drawn from the same distribution as the training instances

MLN Transfer
(Mihalkova, Huynh, & Mooney, 2007)

- Given two multi-relational domains, such as:

Source:

UW-CSE
Professor(A) Student(A)
Publication(T, A)
AdvisedBy(A, B)
…

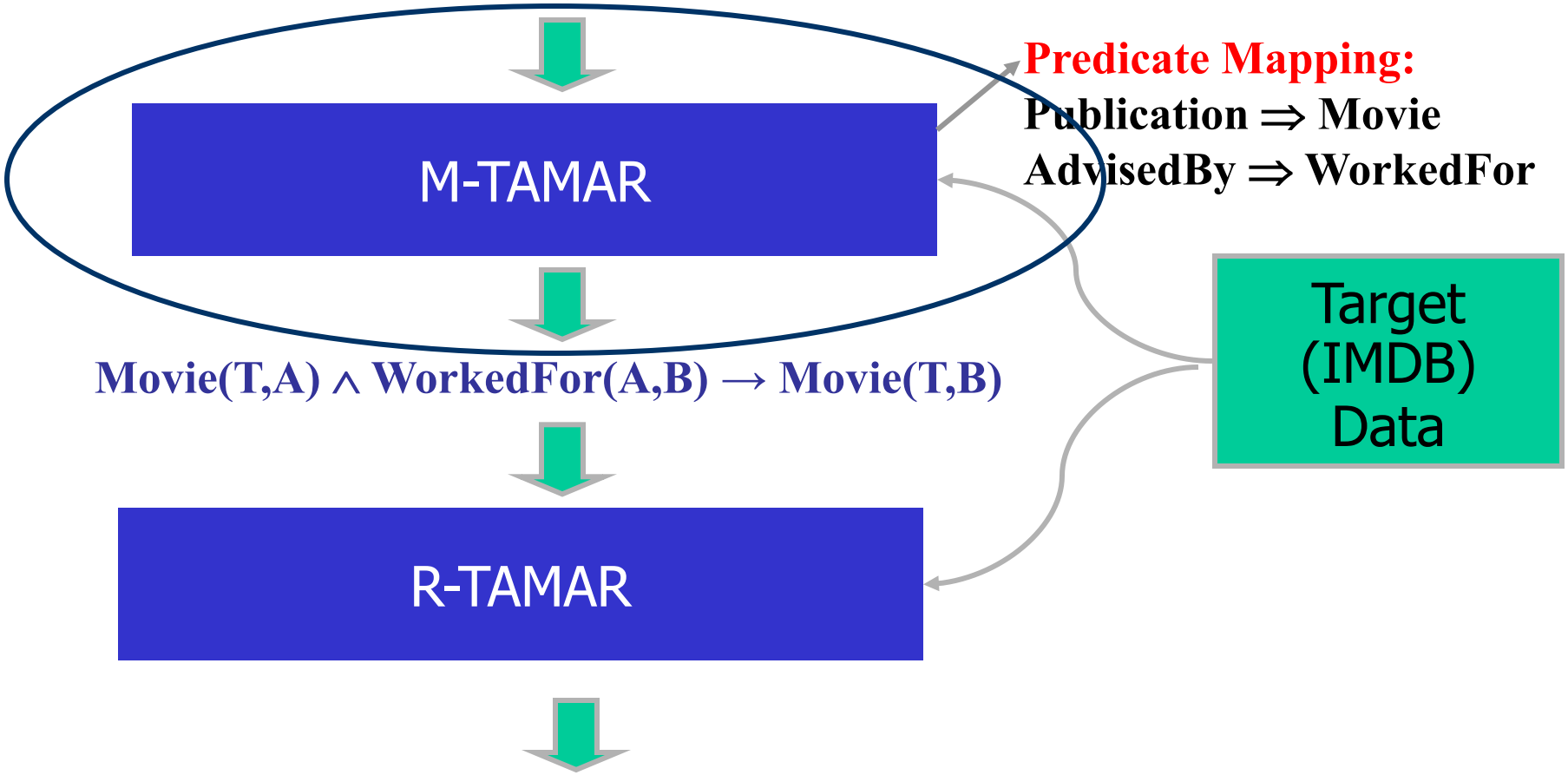
Target:

IMDB
Director(A) Actor(A)
Movie(T, A)
WorkedFor(A, B)
…

- Transfer a Markov logic network learned in the Source to the Target by:
  - Mapping the Source predicates to the Target
  - Revising the mapped knowledge

TAMAR
(Transfer via Automatic Mapping And Revision)

Publication(T,A) ∧ AdvisedBy(A,B) → Publication(T,B) **(clause from UW-CSE)**



Movie(T,A) ∧ WorkedFor(A,B) ∧ Relative(A,B) → Movie(T,B)

Predicate Mapping

- Each clause is mapped independently of the others.
- The algorithm considers all possible ways to map a clause such that:
  - Each predicate in the source clause is mapped to some target predicate.
  - Each argument type in the source is mapped to **exactly** one argument type in the target.
- Each mapped clause is evaluated by measuring its fit to the target data, and the most accurate mapping is kept.

Predicate Mapping Example

$$Publication(T, A) \wedge AdvisedBy(A, B) \Rightarrow Publication(T, B)$$

Publication(title, person) → Movie(name, person)

AdvisedBy(person, person) → WorkedFor(person, person)

Consistent Type Mapping:

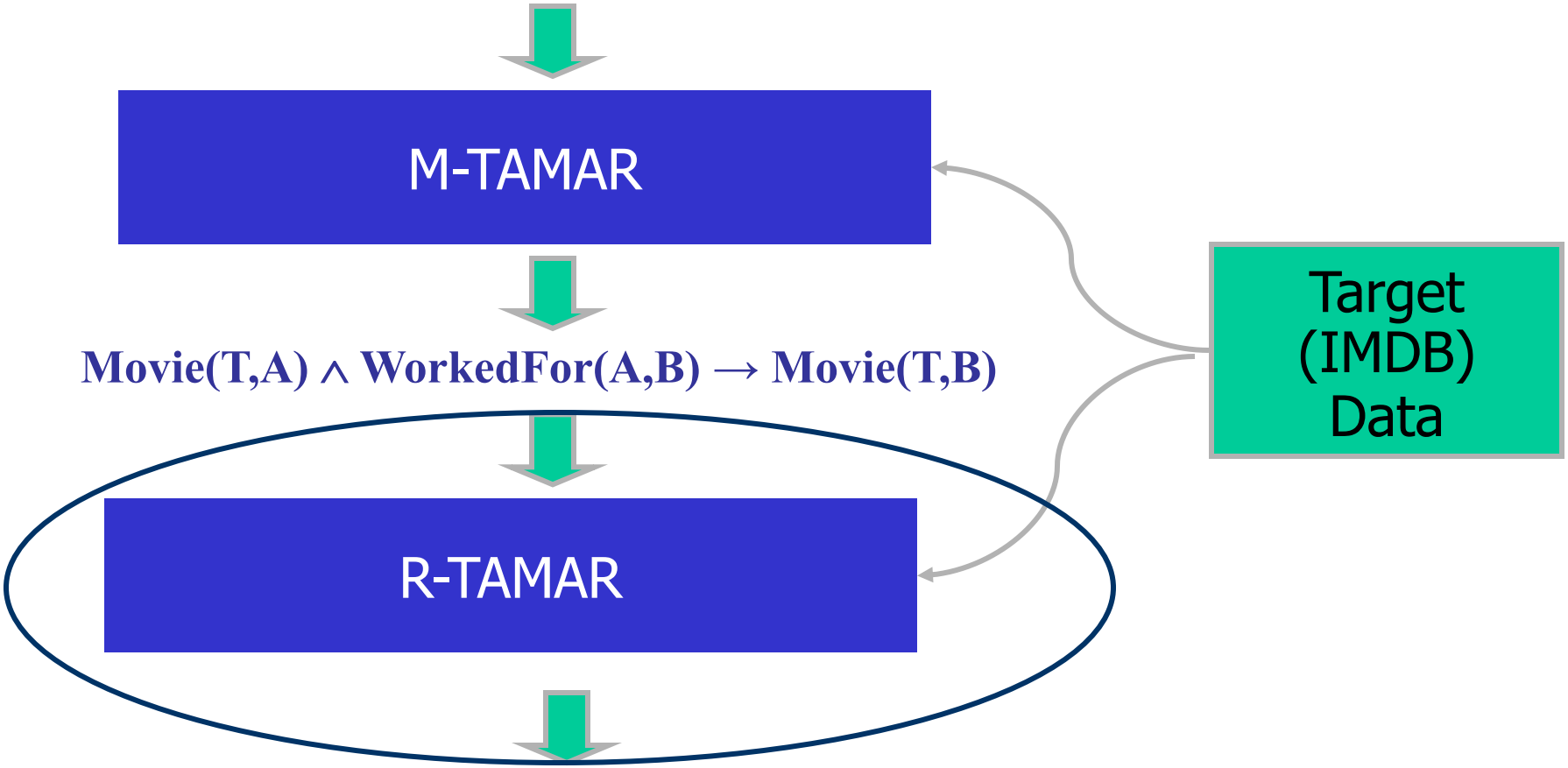
title → name

person → person

$$Movie(T, A) \wedge WorkedFor(A, B) \Rightarrow Movie(T, B)$$

TAMAR
(Transfer via Automatic Mapping And Revision)

Publication(T,A) ∧ AdvisedBy(A,B) → Publication(T,B) **(clause from UW-CSE)**

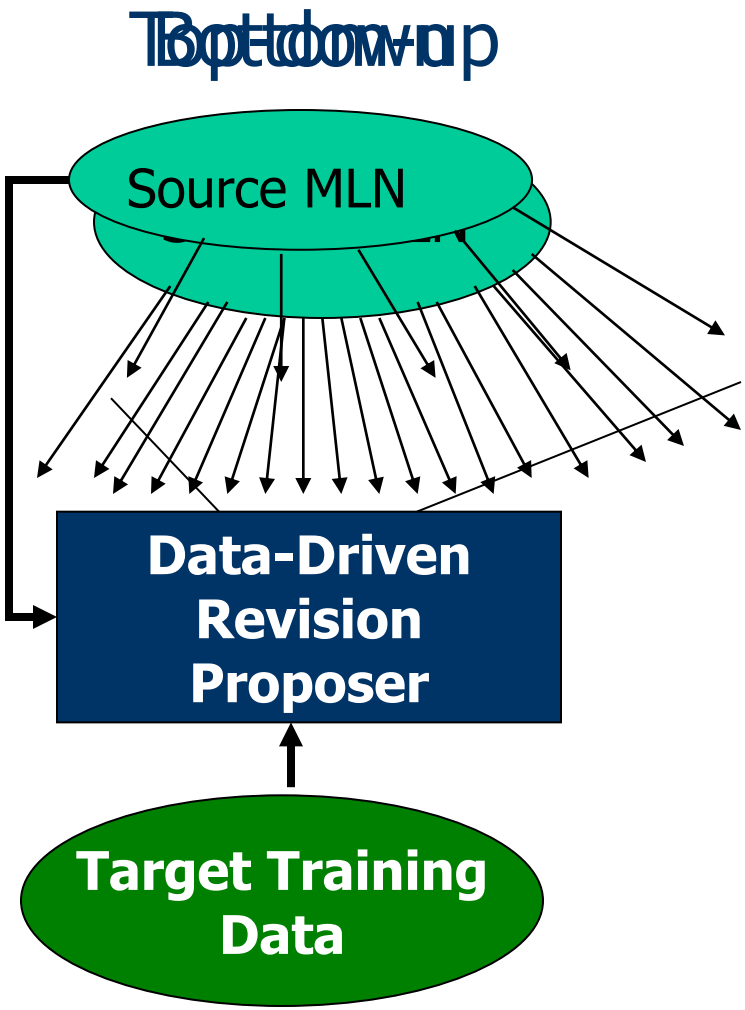


Movie(T,A) ∧ WorkedFor(A,B) → Movie(T,B)

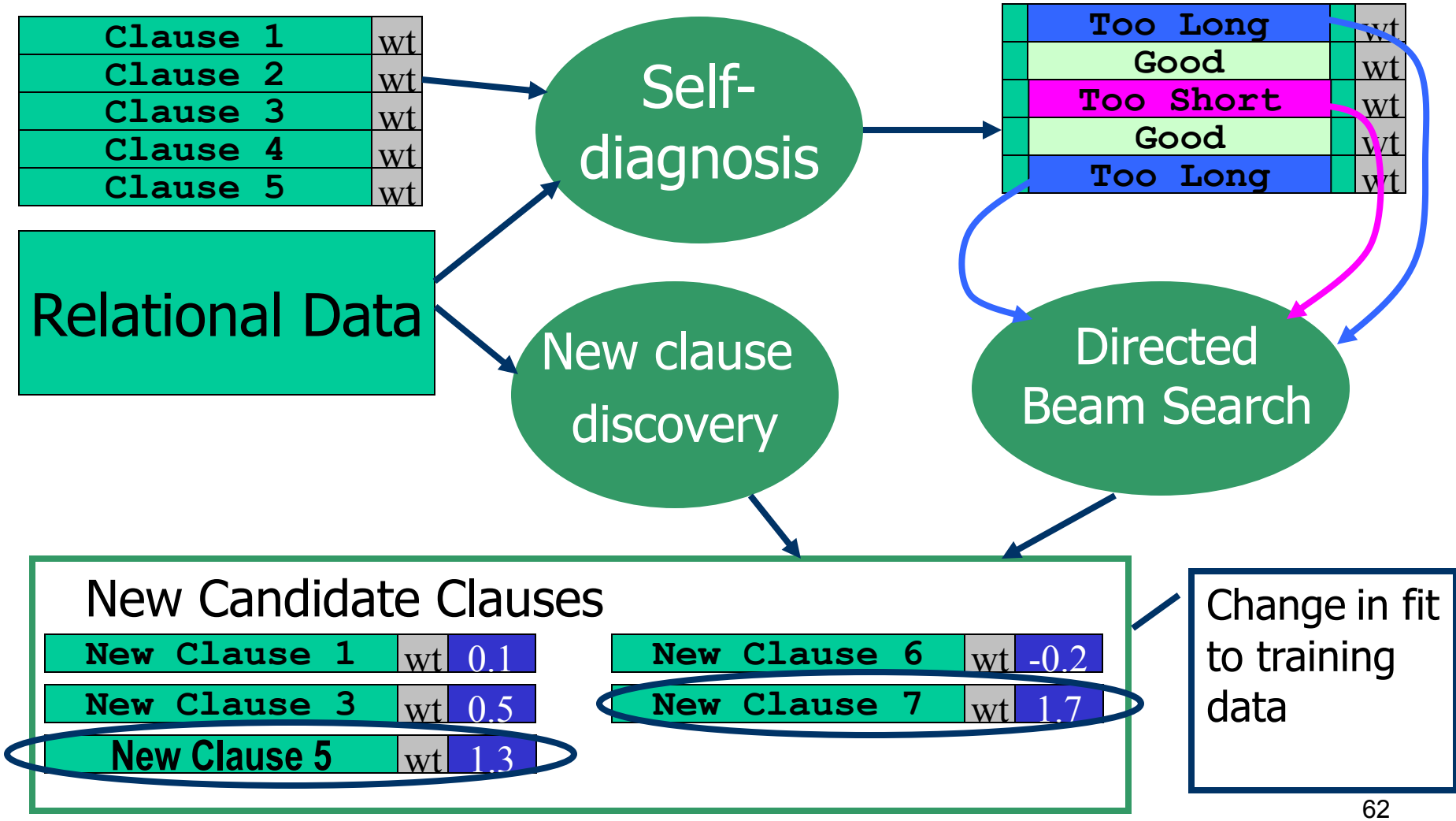
Movie(T,A) ∧ WorkedFor(A,B) ∧ Relative(A,B) → Movie(T,B)

Transfer Learning as Revision

- Regard mapped source MLN as an approximate model for the target task that needs to be accurately and efficiently revised.
- Thus our general approach is similar to that taken by theory revision systems (FORTE, Richards & Mooney, 1995).
- Revisions are proposed in a bottom-up fashion.



R-TAMAR


Clause 1
Clause 2
Clause 3
Clause 4
Clause 5
wt
Relational Data
Self-diagnosis
New clause discovery
Too Long
Good
Too Short
Good
Too Long
Directed Beam Search
New Candidate Clauses
New Clause 1 wt 0.1
New Clause 3 wt 0.5
New Clause 5 wt 1.3
New Clause 6 wt -0.2
New Clause 7 wt 1.7
Change in fit to training data

Structure Revisions

- Using directed beam search:
  - Literal deletions attempted only from clauses marked for shortening.
  - Literal additions attempted only for clauses marked for lengthening.
- Training is much faster since search space is constrained by:
  1. Limiting the clauses considered for updates.
  2. Restricting the type of updates allowed.

New Clause Discovery

- Uses Relational Pathfinding

Weight Revision

Publication(T,A) ∧ AdvisedBy(A,B) → Publication(T,B)

M-TAMAR

Movie(T,A) ∧ WorkedFor(A,B) → Movie(T,B)

R-TAMAR

Movie(T,A) ∧ WorkedFor(A,B) ∧ Relative(A,B) → Movie(T,B)

MLN Weight Training

0.8 Movie(T,A) ∧ WorkedFor(A,B) ∧ Relative(A,B) → Movie(T,B)

Target (IMDB) Data

TAMAR
Experiments

Systems Compared

- TAMAR: Complete transfer system.
- ScrTDSL: Algorithm of Kok & Domingos (2005) learning from scratch.
- TrTDSL: Algorithm of Kok & Domingos (2005) performing transfer, using M-TAMAR to produce a mapping.

Manually Developed Source KB

- UW-KB is a hand-built knowledge base (set of clauses) for the UW-CSE domain.
- When used as a source domain, transfer learning is a form of theory refinement that also includes mapping to a new domain with a different representation.

Metrics to Summarize Curves

- Transfer Ratio (Cohen et al. 2007)

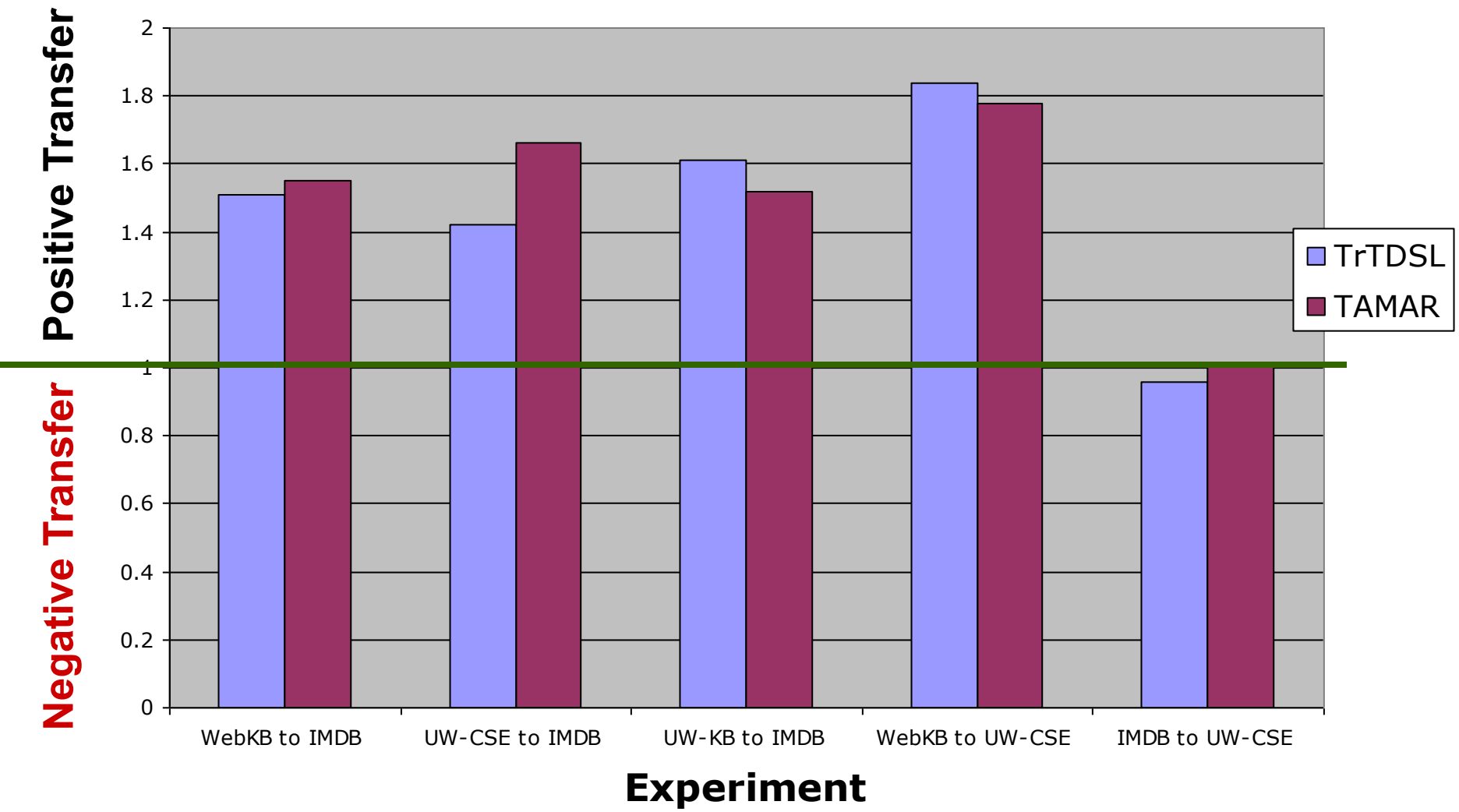
$$\text{Transfer Ratio} = \frac{\text{Area under learning curve of transfer learner}}{\text{Area under learning curve of learner from scratch}}$$

  – Gives overall idea of improvement achieved over learning from scratch

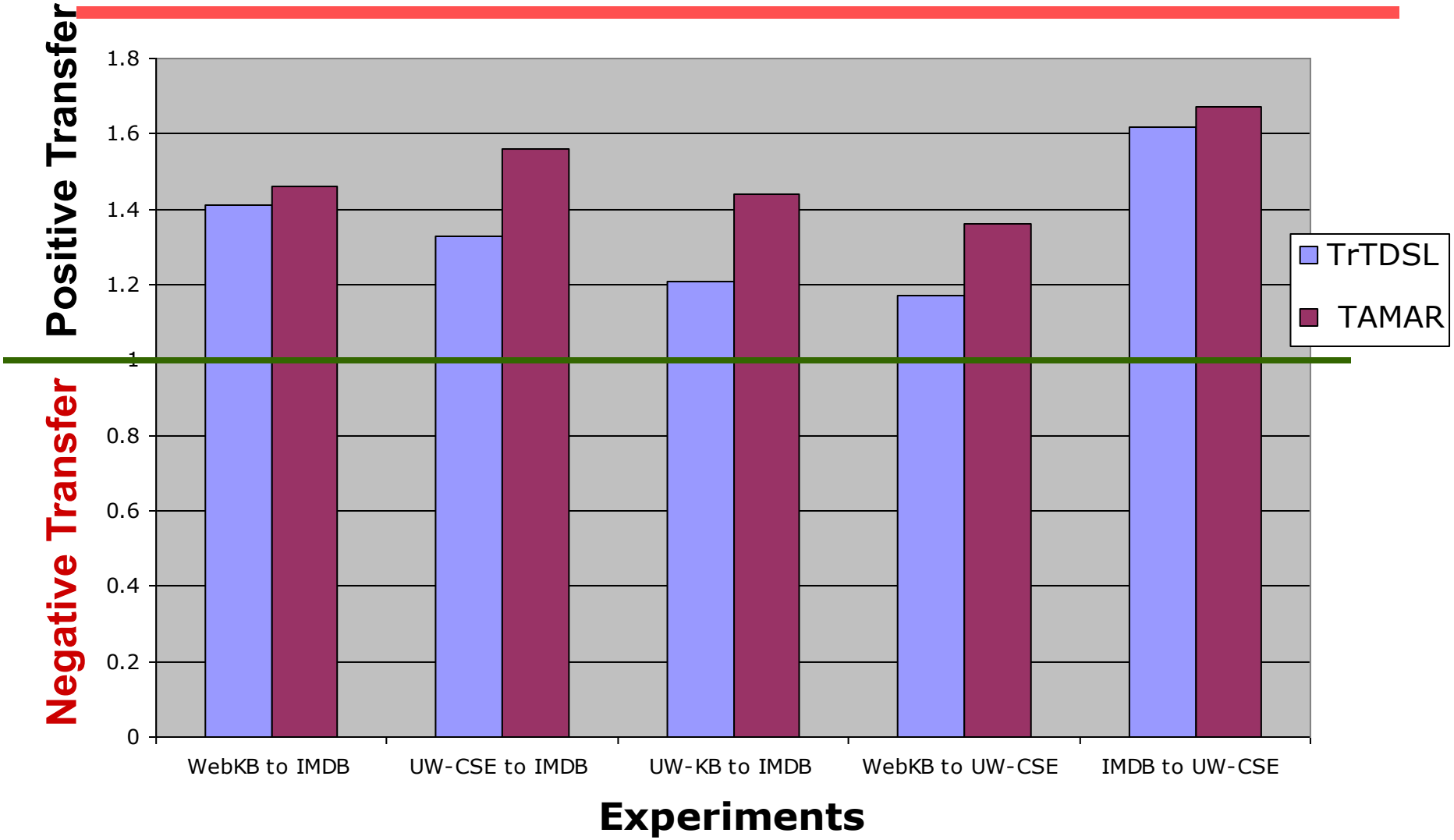
Transfer Scenarios

- Source/target pairs tested:
  - WebKB ⇒ IMDB
  - UW-CSE ⇒ IMDB
  - UW-KB ⇒ IMDB
  - WebKB ⇒ UW-CSE
  - IMDB ⇒ UW-CSE
- WebKB not used as a target since one mega-example is sufficient to learn an accurate theory for its limited predicate set.

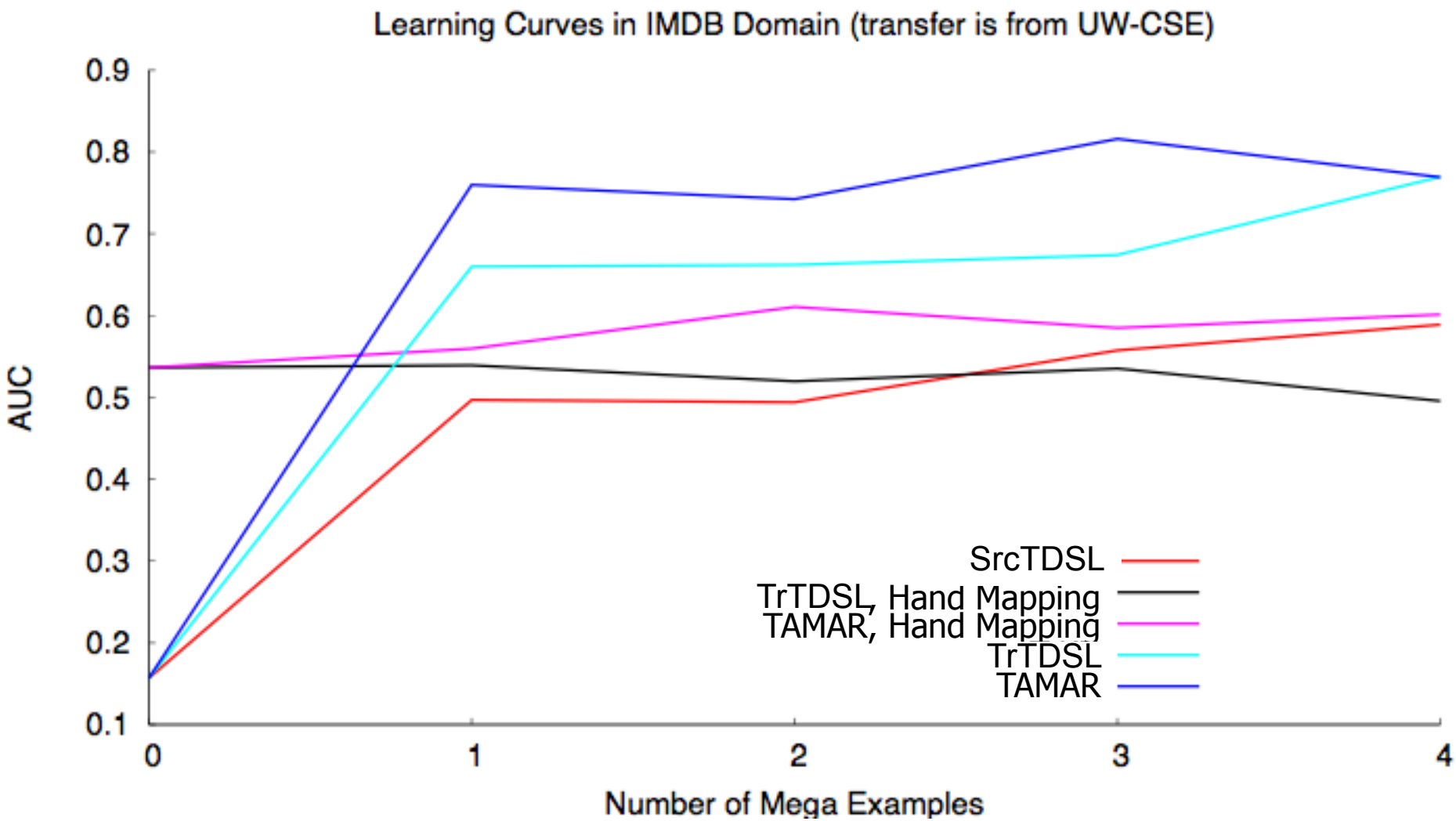
AUC Transfer Ratio



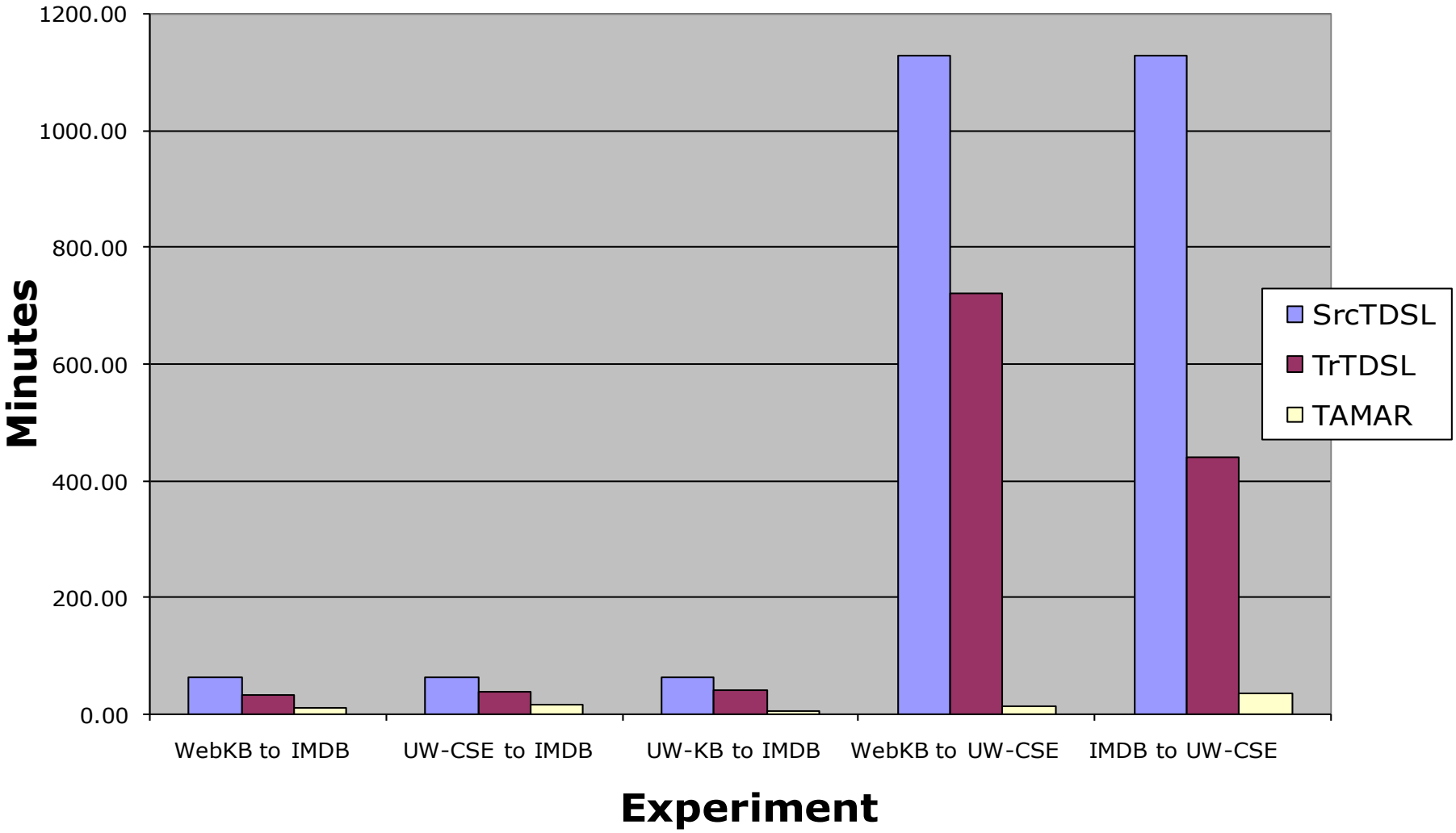
CLL Transfer Ratio



Sample Learning Curve



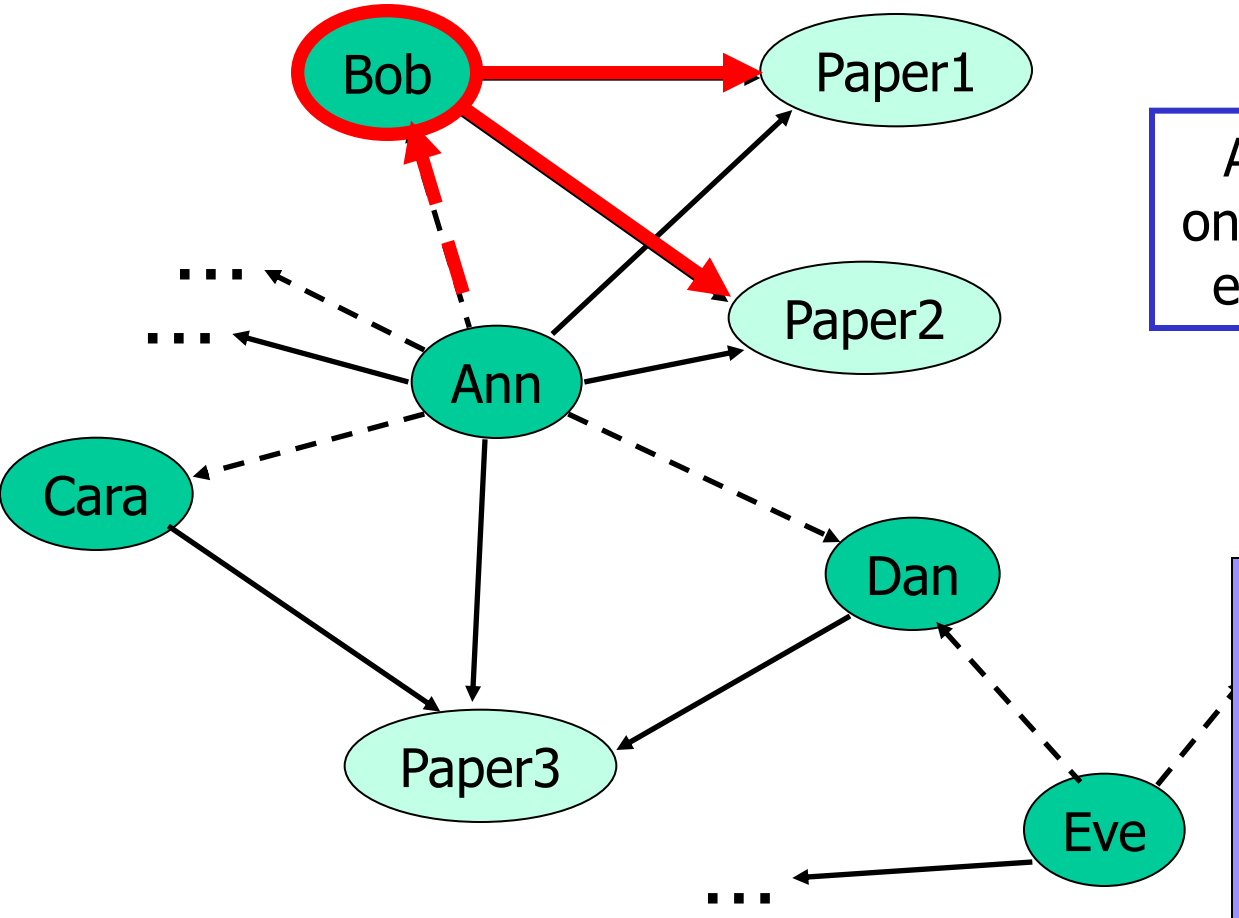
Total Training Time in Minutes



Transfer Learning with Minimal Target Data
(Mihalkova & Mooney, 2009)

- Recently extended TAMAR to learn with ***extremely little*** target data.
- Just use minimal target data to determine a good predicate mapping from the source.
- Transfer mapped clauses without revision or weight learning.

Minimal Target Data



Assume knowledge of only a few entities, in the extreme case just one.

Predicates/relations:

written-by (doc, person)

advised-by(person, person)

SR2LR Basic Idea (Short Range to Long Range)

- Clauses can be divided into two categories
  - **Short-range**: concern information about a single entity

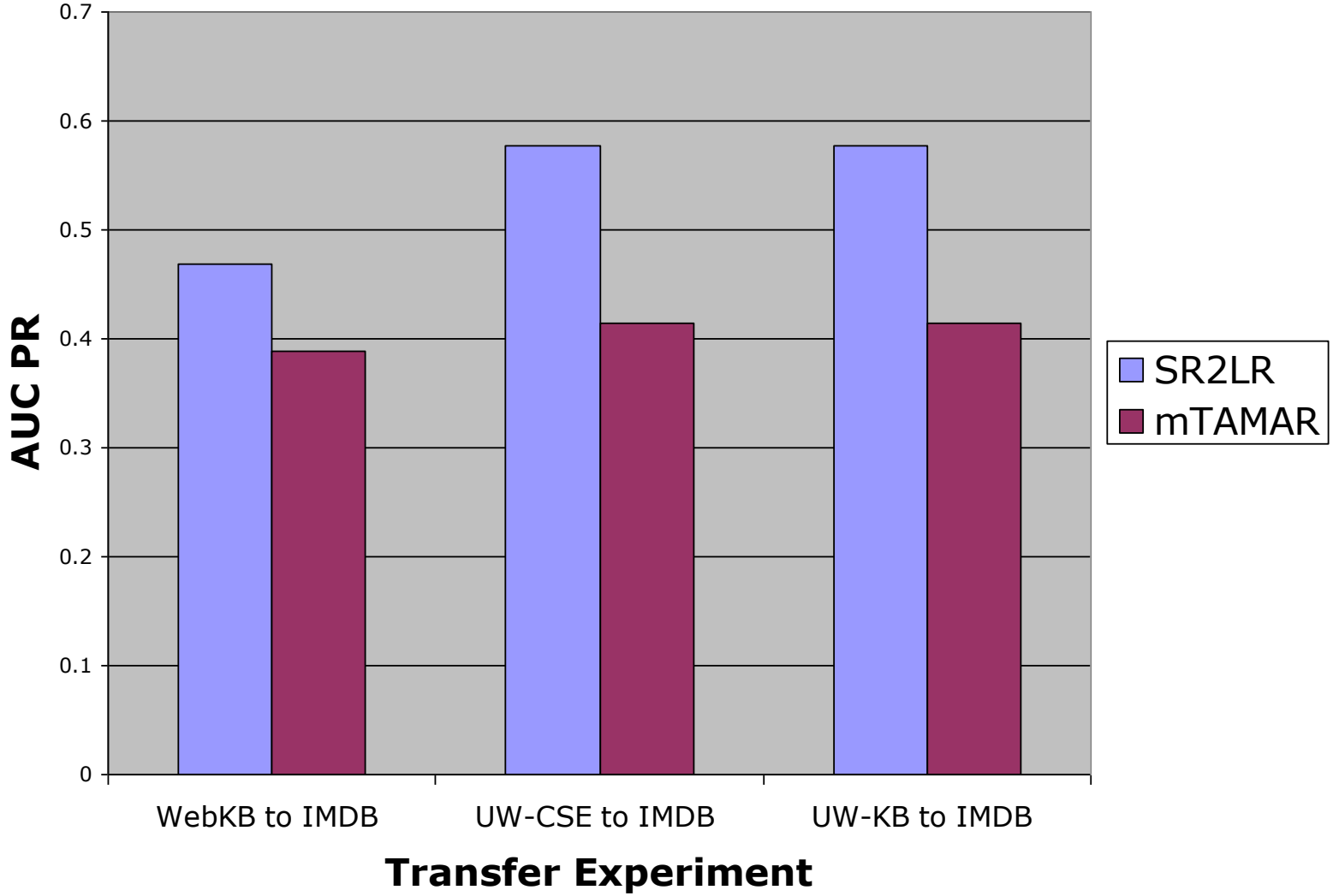
$$\text{advised-by}(a, b) \Rightarrow \neg\text{is-professor}(a)$$

  - **Long-range**: relate information about multiple entities

$$\text{written-by}(m, a) \wedge \text{written-by}(m, b) \wedge \text{is-professor}(b) \Rightarrow \text{advised-by}(a, b)$$

- Key:
  - Discover useful ways of mapping source predicates to the target domain by testing them only on short-range clauses
  - Then apply them to the long-range clauses

Results for Single Entity Training Data in IMBD Target Domain



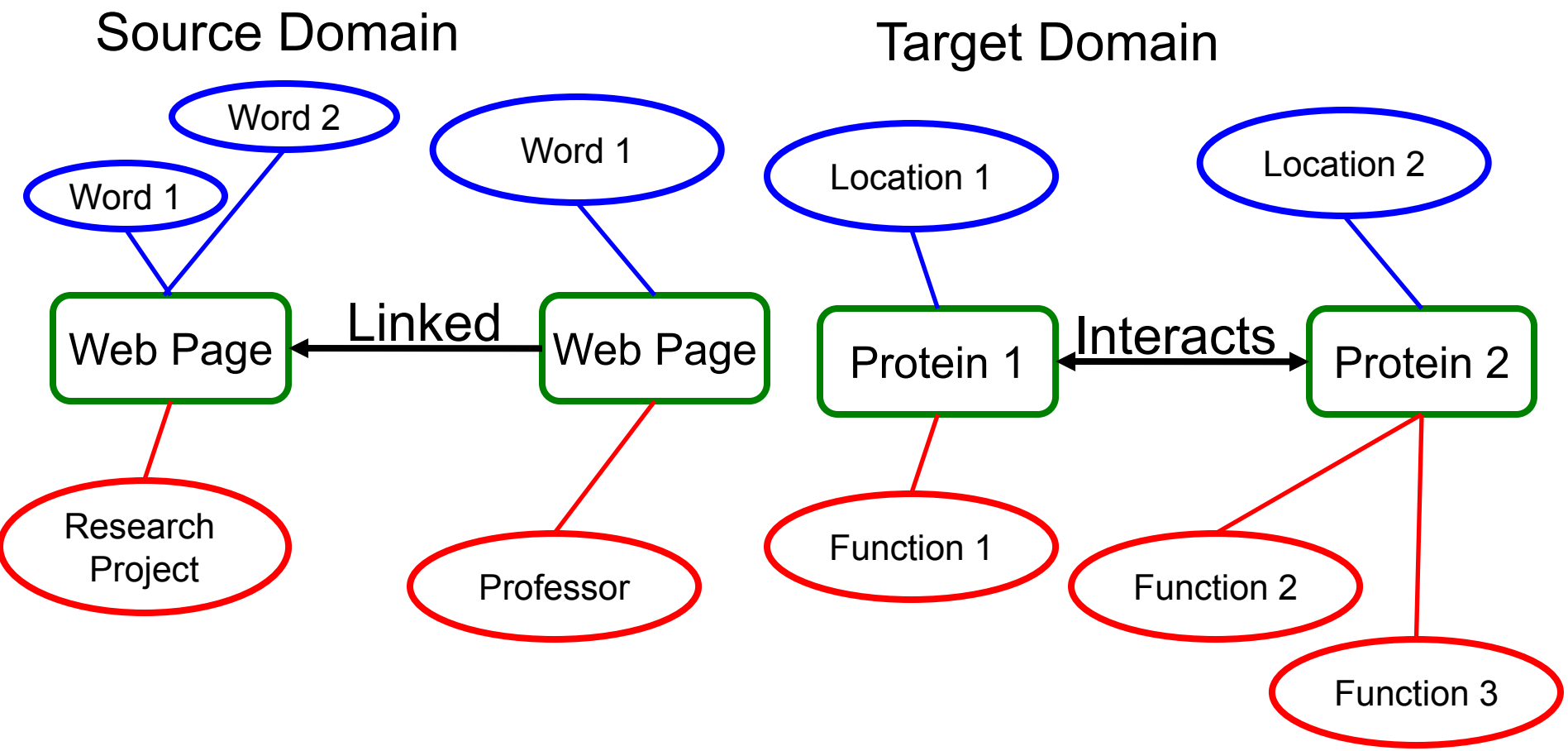
Deep Transfer with 2nd-Order MLNs (Davis & Domingos, 2009)

- Transfer very abstract patterns between disparate domains.

- Learn patterns in 2nd-order logic that variablize over predicates.

Deep Transfer

Generalizing to Very Different Domains



Deep Transfer via Markov Logic (DTM)

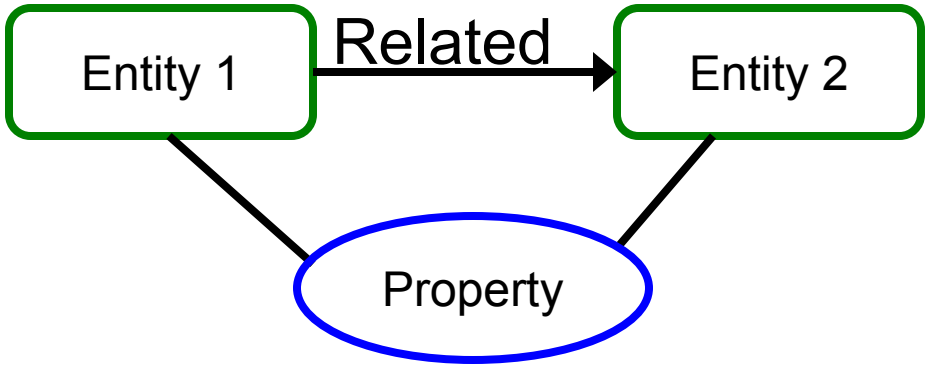
- **Representation:** 2nd-order formulas
  - Abstract away predicate names
  - Discern high-level structural regularities
- **Search:** Find good 2nd-order formulas
- **Evaluation:** Check if 2nd-order formula captures a regularity beyond product of sub-formulas
- **Transfer:** Knowledge provides declarative bias in the target domain

Datasets

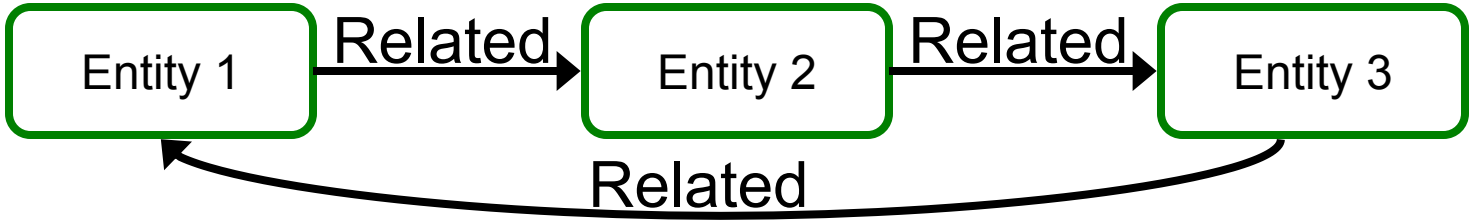
- Yeast Protein (Davis et al. 2005)
  - Protein-protien interaction data from yeast
  - 7 predicates, 7 types, 1.4M ground atoms
  - Predict Function,Interaction
- WebKB (Craven et al. 2001)
  - Webpages from 4 CS departments
  - 3 predicates, 3 types, 4.4M ground atoms
  - Predict Page Class, Linked
- Facebook Social Network (source only)
  - 13 predicates, 12 types, 7.2M ground atoms

High-Scoring 2nd-Order Cliques

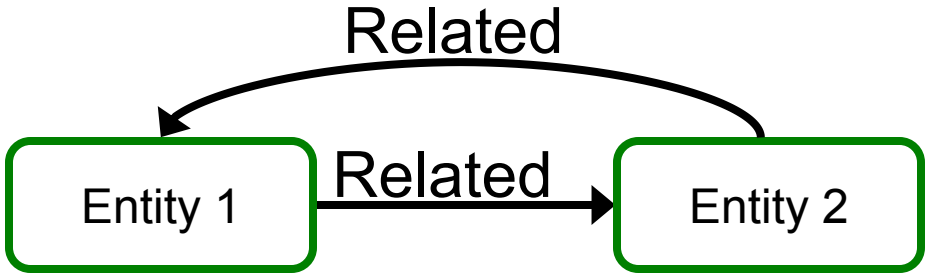
Homophily



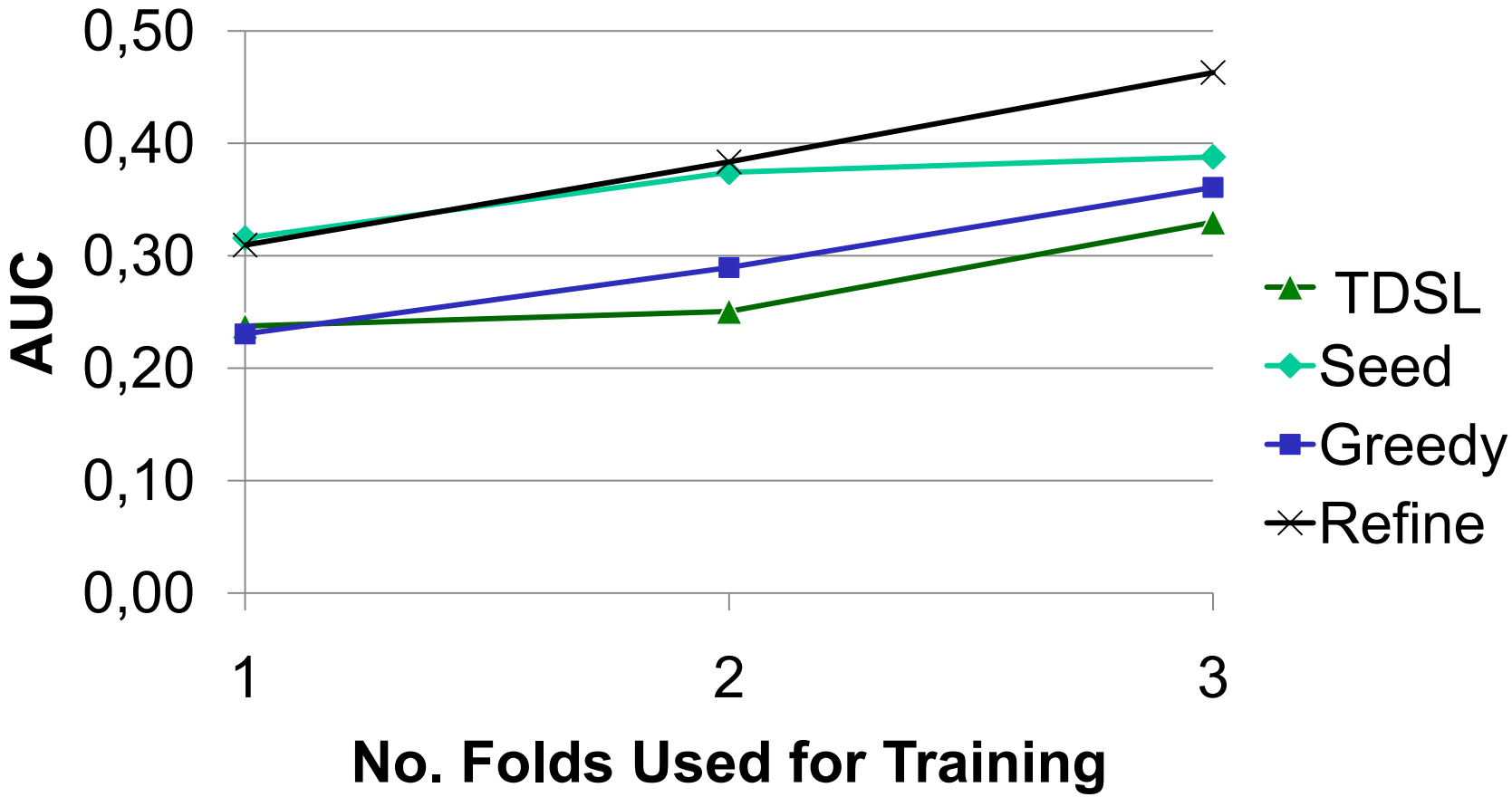
Transitivity



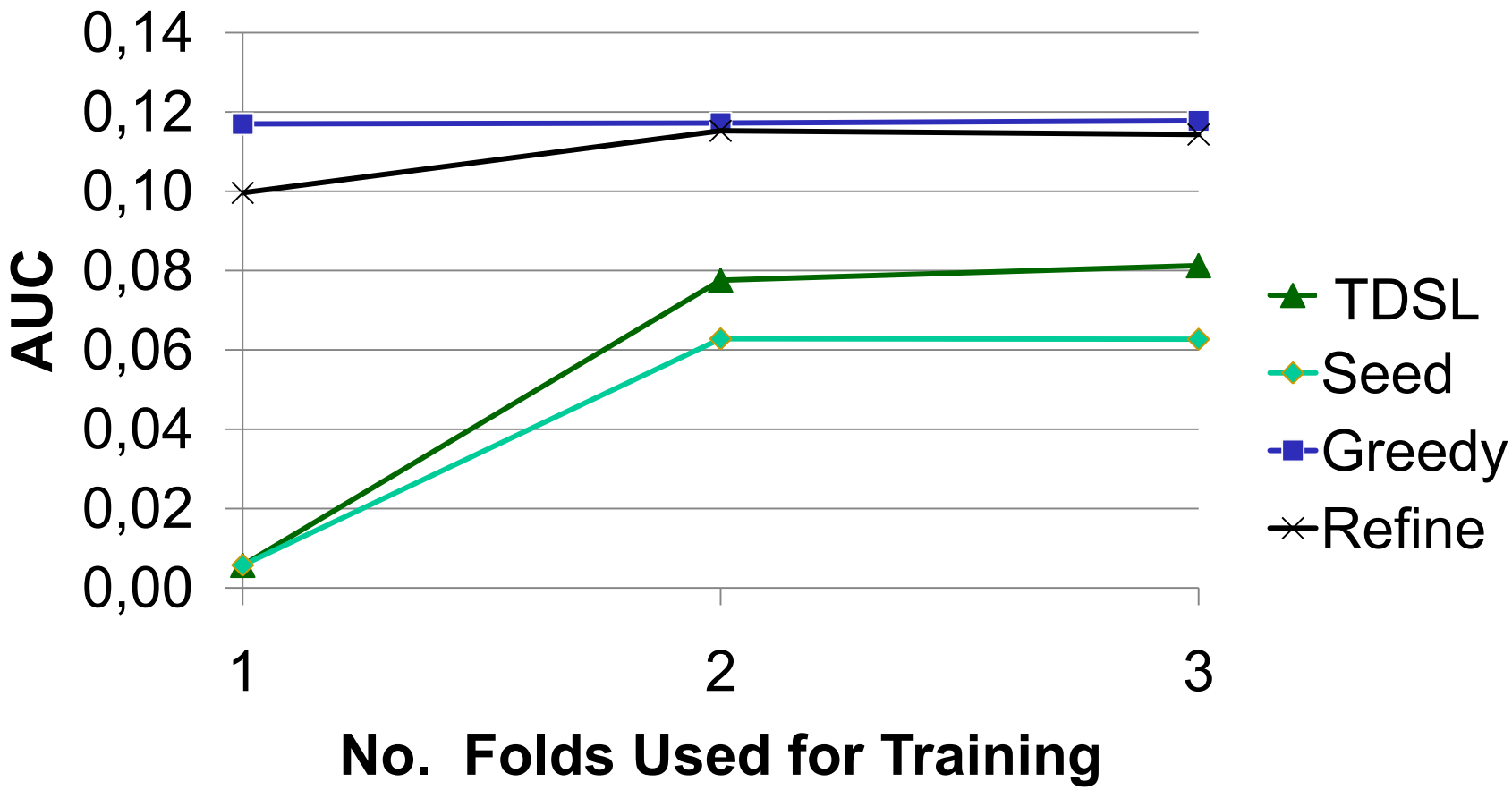
Symmetry



WebKB to Yeast Protein to Predict Function



Facebook to WebKB to Predict Linked



Future Research Issues

- More realistic application domains.
- More bottom-up transfer learners.
- Application to other SRL models (e.g. SLPs, BLPs).
- More flexible predicate mapping
  - Allow argument ordering or arity to change.
  - Map 1 predicate to conjunction of > 1 predicates
    - $\text{AdvisedBy}(X,Y) \Rightarrow \text{Actor}(M,X) \wedge \text{Director}(M,Y)$

Multiple Source Transfer

- Transfer from ***multiple*** source problems to a given target problem.
- Determine which clauses to map and revise from different source MLNs.

Source Selection

- Select useful source domains from a large number of previously learned tasks.
- Ideally, picking source domain(s) is sub-linear in the number of previously learned tasks.

Conclusions

- Two important ways to improve structure learning for SRL models such as MLNs:
  - **Bottom-up Search**: BUSL, Aleph-MLN, LHL
  - **Transfer Learning:** TAMAR, SR2LR, 2ndOrderMLN
- Both improve both the speed of training and the accuracy of the learned model.
- Ideas from "classical ILP" can be very effective for improving SRL.

Questions?

Related papers at:

http://www.cs.utexas.edu/users/ml/publication/srl.html

Why MLNs?

- Inherit the expressivity of first-order logic
  - Can apply insights from ILP
- Inherit the flexibility of probabilistic graphical models
  - Can deal with noisy & uncertain environments
- Undirected models
  - Do not need to learn causal directions
- Subsume all other SRL models that are special cases of first-order logic or probabilistic graphical models [Richardson 04]
- Publicly available software package: Alchemy

Predicate Mapping Comments

- A particular source predicate can be mapped to *different* target predicates in *different* clauses.
  - This makes our approach context sensitive.
- More scalable.
  - In the worst-case, the number of mappings is exponential in the number of predicates.
  - The number of predicates in a clause is generally much smaller than the total number of predicates in a domain.

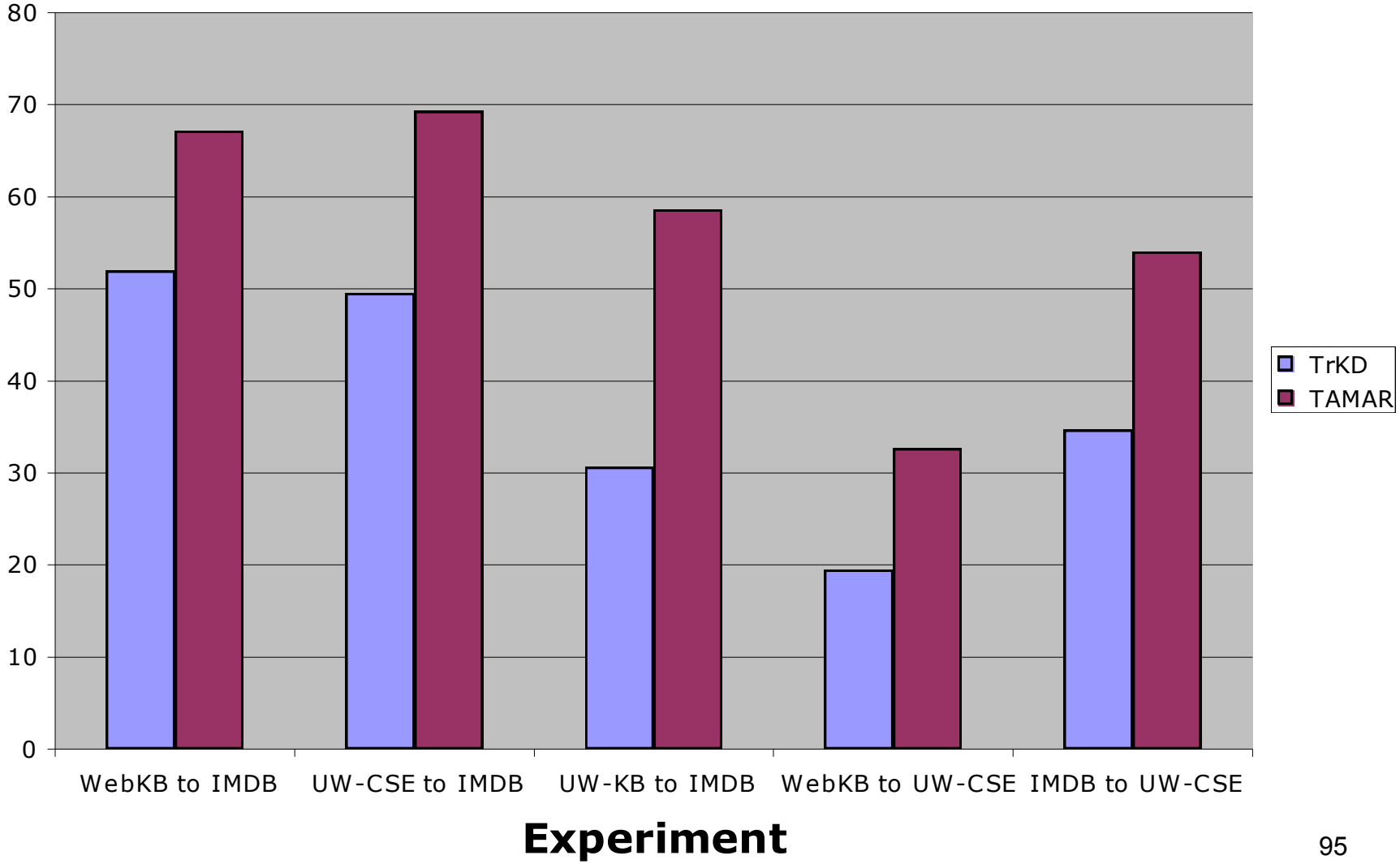
Relationship to Structure Mapping Engine (Falkenheiner et al., 1989)

- A system for mapping relations using analogy based on a psychological theory.
- Mappings are evaluated based only on the structural relational similarity between the two domains.
- Does not consider the ***accuracy*** of mapped knowledge in the target when determining the preferred mapping.
- Determines a single global mapping for a given source & target.

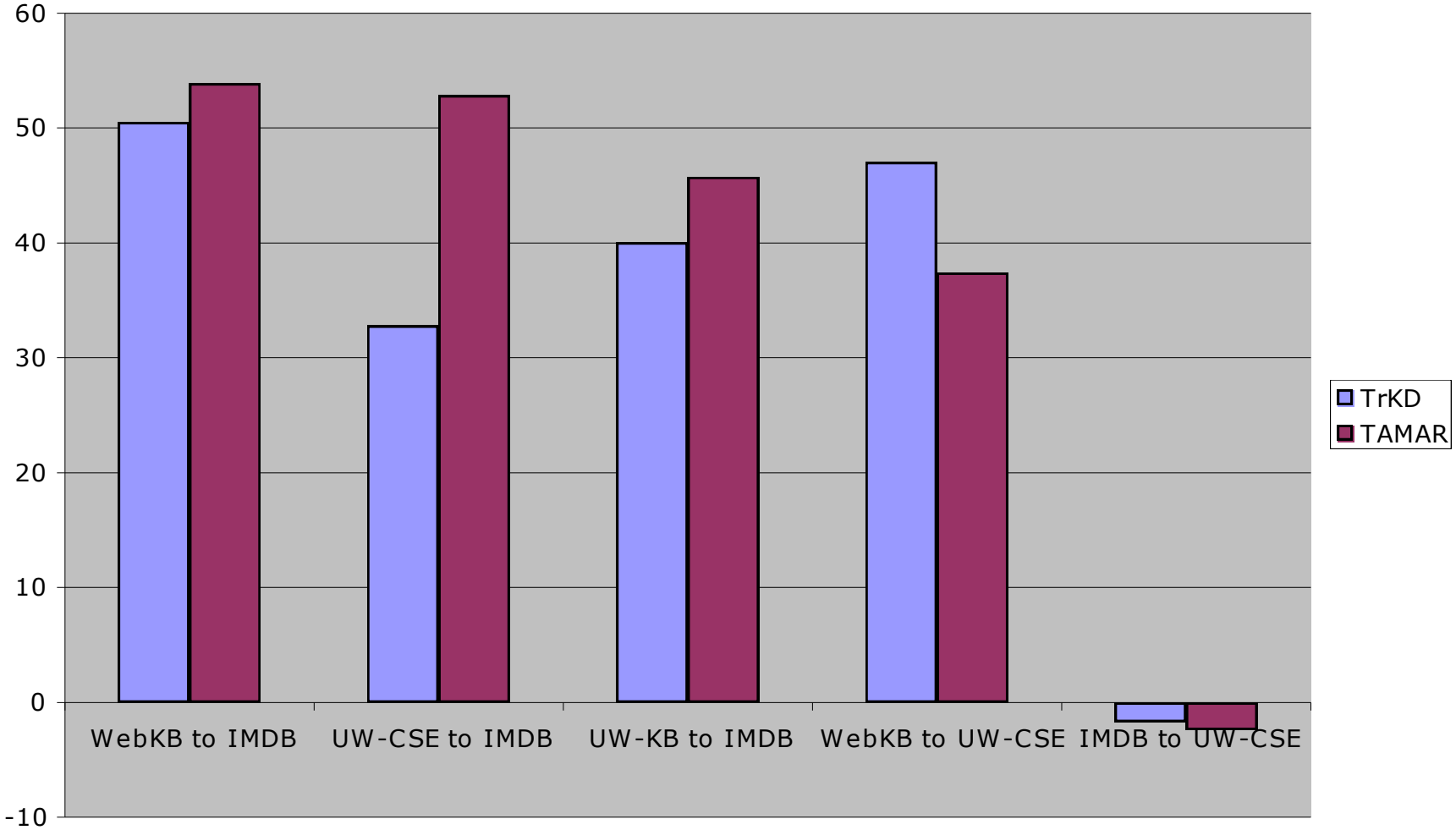
Summary of Methodology

1. Learn MLNs for each point on learning curve
2. Perform inference over learned models
3. Summarize inference results using 2 metrics: CLL and AUC, thus producing two learning curves
4. Summarize each learning curve using transfer ratio and percentage improvement from one mega-example

CLL Percent Improvement



AUC Percent Improvement



Experiment