Cross-Domain Transfer of Constraints for Inductive Process Modeling

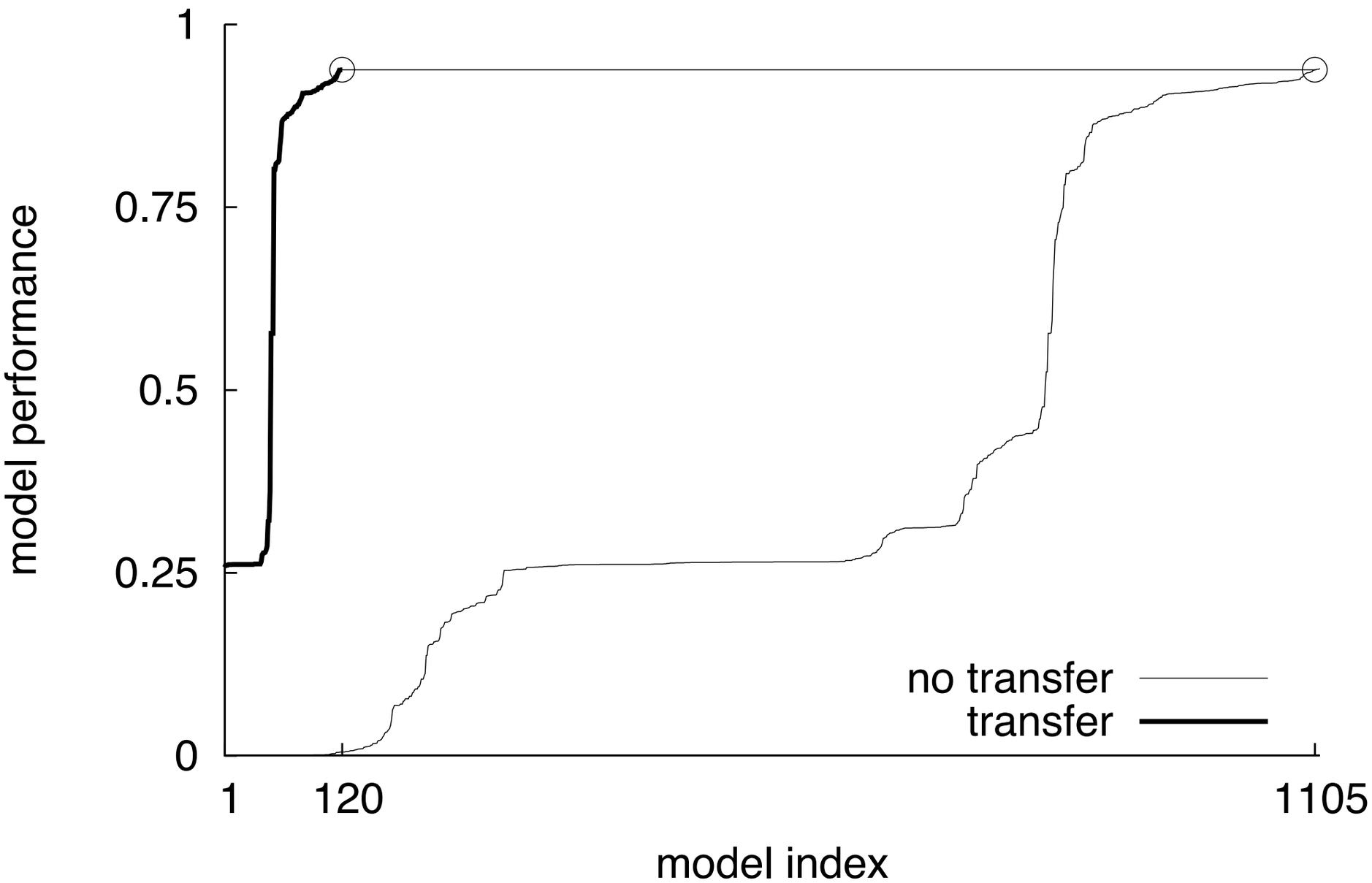
Will Bridewell and
Ljupco Todorovski

Computational Learning Laboratory
CSLI, Stanford University

with acknowledgements to Pat Langley and Tolga Könik.

willb@csli.stanford.edu

Ross-96 to Ross-97



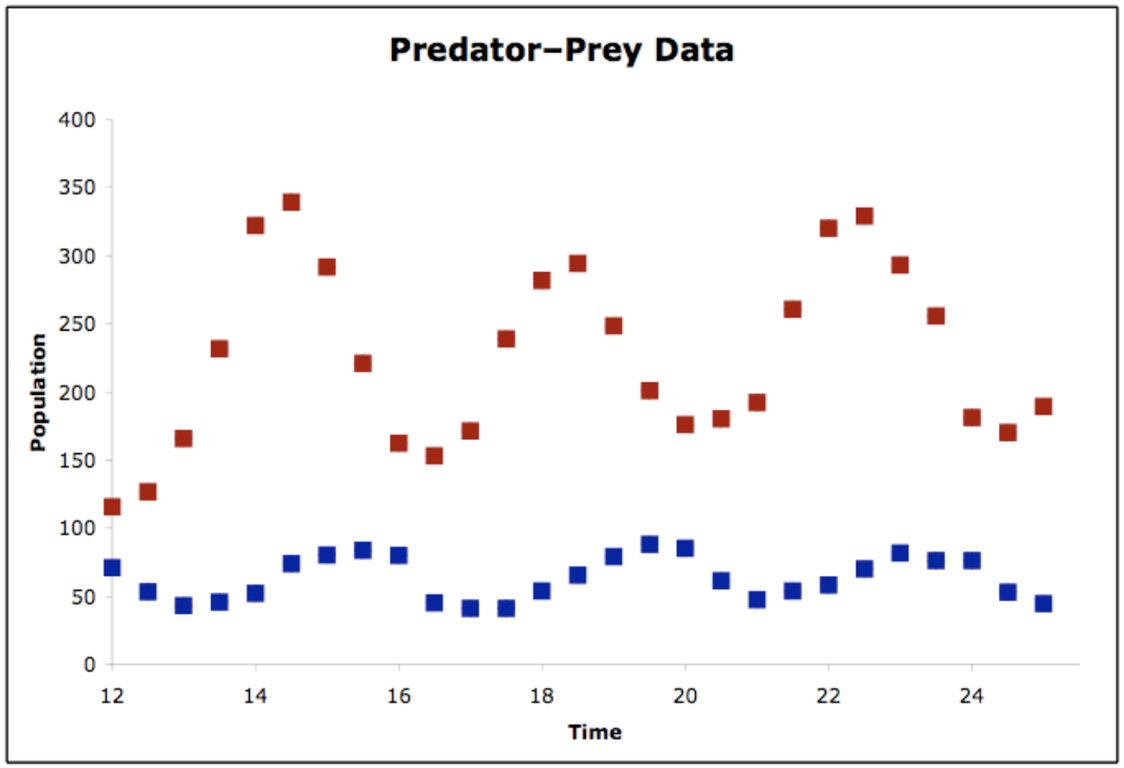
Outline

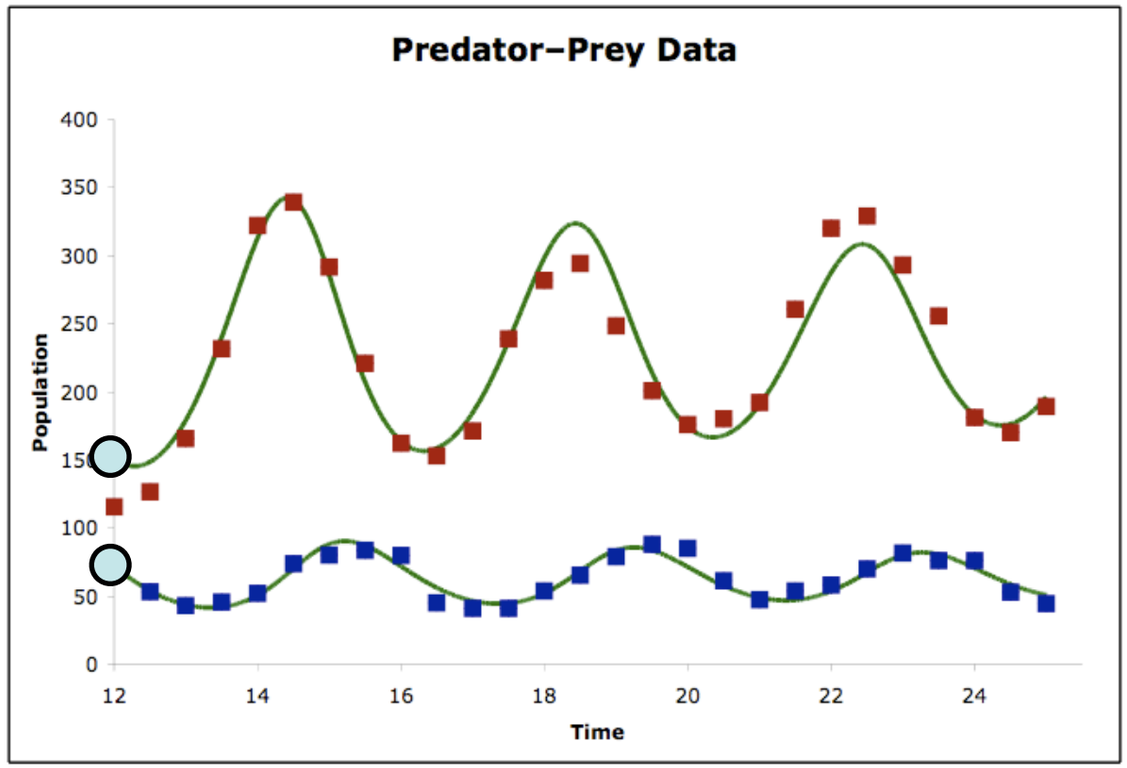
- Inductive process modeling
- Constraining model structures
- Learning constraints for process modeling
- Measures for evaluating transfer
- Cross domain transfer

Inductive Process Modeling

Model

Observations → Predictions





Model Objectives: Explanation and Prediction

Bridewell et al. 2008, MLj

Quantitative Process Models

- Ordinary Differential Equations

$$d\mathit{hare.density}/dt = 2.5 * \mathit{hare.density} + -0.1 * \mathit{hare.density} * \mathit{wolf.density}$$
$$d\mathit{wolf.density}/dt = -1.2 * \mathit{wolf.density} + 0.3 * 0.1 * \mathit{hare.density} * \mathit{wolf.density}$$

- Processes

process exponential_growth
  equations $d[\mathit{hare.density}, t, 1] = 2.5 * \mathit{hare.density}$

process exponential_loss
  equations $d[\mathit{wolf.density}, t, 1] = -1.2 * \mathit{wolf.density}$

process predation_holling_type_1
  equations $d[\mathit{hare.density}, t, 1] = -0.1 * \mathit{hare.density} * \mathit{wolf.density}$
$d[\mathit{wolf.density}, t, 1] = 0.3 * 0.1 * \mathit{hare.density} * \mathit{wolf.density}$

Quantitative Process Models

- Ordinary Differential Equations

  d*hare.density*/dt = 2.5 * *hare.density*
  d*wolf.density*/dt = −1.2 * *wolf.density*

- Processes

  process exponential_growth
    equations *d[hare.density, t, 1]* = 2.5 * *hare.density*

  process exponential_loss
    equations *d[wolf.density, t, 1]* = −1.2 * *wolf.density*

Quantitative Process Models

- Ordinary Differential Equations

$$\frac{dhare.density}{dt} = 2.5 * hare.density + -0.1 * hare.density * wolf.density / (1 + 0.2 * -0.1 * hare.density)$$

$$\frac{dwolf.density}{dt} = -1.2 * wolf.density + 0.3 * 0.1 * hare.density * wolf.density / (1 + 0.2 * -0.1 * hare.density)$$

- Processes

process exponential_growth
  equations $d[hare.density, t, 1] = 2.5 * hare.density$

process exponential_loss
  equations $d[wolf.density, t, 1] = -1.2 * wolf.density$

process predation_holling_type_2
  equations $d[hare.density, t, 1] = -0.1 * hare.density * wolf.density / (1 + 0.2 * -0.1 * hare.density)$
  $d[wolf.density, t, 1] = 0.3 * 0.1 * hare.density * wolf.density / (1 + 0.2 * -0.1 * hare.density)$

Model Components

generic process predation_Holling_1{predation}
  entities *P1*{prey}, *P2*{predator}
  parameters *r*[0, infinity], *e*[0, infinity]
  equations d[*P1.density*, t, 1] = −1 * *r* * *P1.density* * *P2.density*
    d[*P2.density*, t, 1] = *e* * *r* * *P1.density* * *P2.density*

Model Components

generic process predation_Holling_1{predation}
entities *P1*{prey}, *P2*{predator}
parameters *r*[0, infinity], *e*[0, infinity]
equations d[*P1.density*, t, 1] = −1 * *r* * *P1.density* * *P2.density*
d[*P2.density*, t, 1] = *e* * *r* * *P1.density* * *P2.density*

P1: hare	P2: wolf
r: 0.1	e: 0.3

← Instantiation

Model Components

generic process predation_Holling_1{predation}
  entities *P1*{prey}, *P2*{predator}
  parameters *r*[0, infinity], *e*[0, infinity]
  equations d[*P1.density*, t, 1] = −1 * *r* * *P1.density* * *P2.density*
    d[*P2.density*, t, 1] = *e* * *r* * *P1.density* * *P2.density*

P1: hare	P2: wolf
r: 0.1	e: 0.3

← Instantiation

process wolves_eat_hares
  equations d[*hare.density*, t, 1] = −1 * 0.1 * *hare.density* * *wolf.density*
    d[*wolf.density*, t, 1] = 0.3 * 0.1 * *hare.density* * *wolf.density*

The IPM System

(a naive approach)

- **Given:**
  - Data
  - A library of generic entities and processes
  - Instantiated entities

- Ground the generic processes with instantiated entities
- Generate **all** combinations of the instantiated processes
- Fit the numeric parameters of each structure

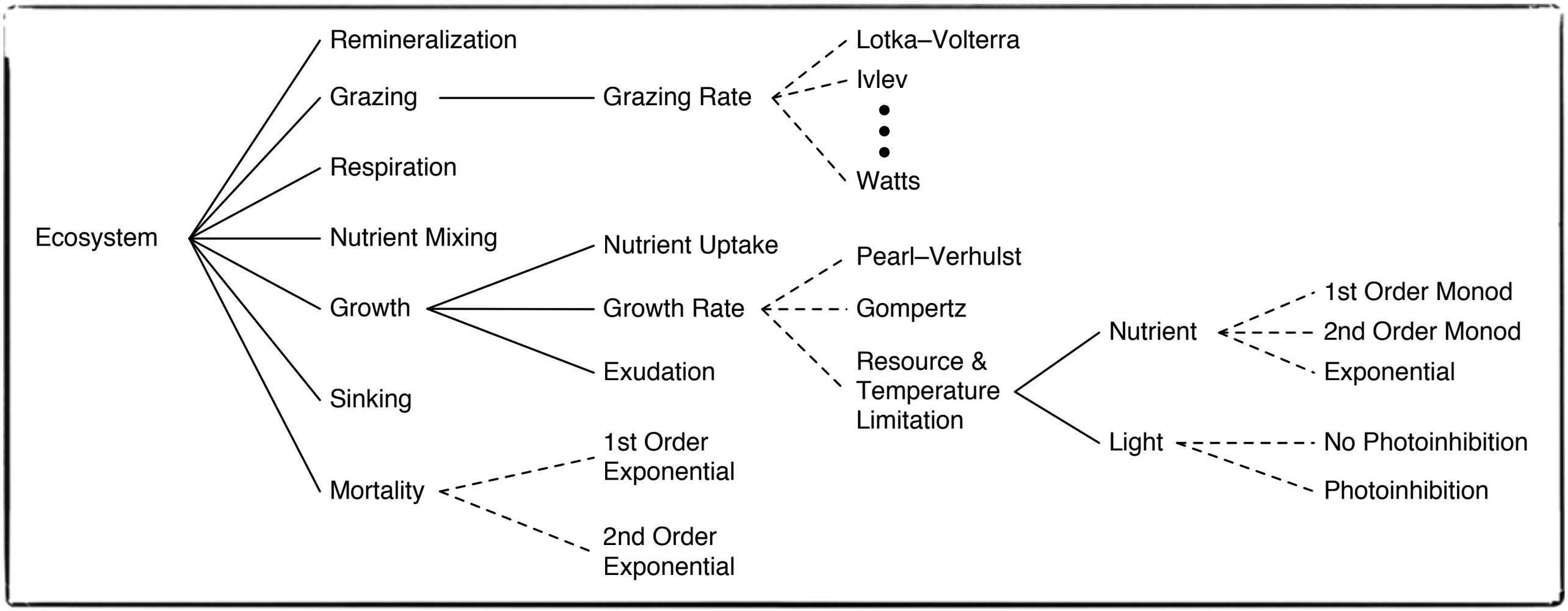
- **Output:** The best model based on its fit to the data

- **Problem:** Many structures are implausible

Structural Constraints

- Eliminate implausible structures
- Reduce the search space
- Make complex domains tractable
- Improve model accuracy during incomplete search

HIPM, Todorovski et al. AAAI-05



Hierarchical Constraints

HIPM, Todorovski et al. AAAI-05

Learning Constraints

Goal:

Identify implicit or unknown constraints
to use in **future** modeling tasks

Plan:

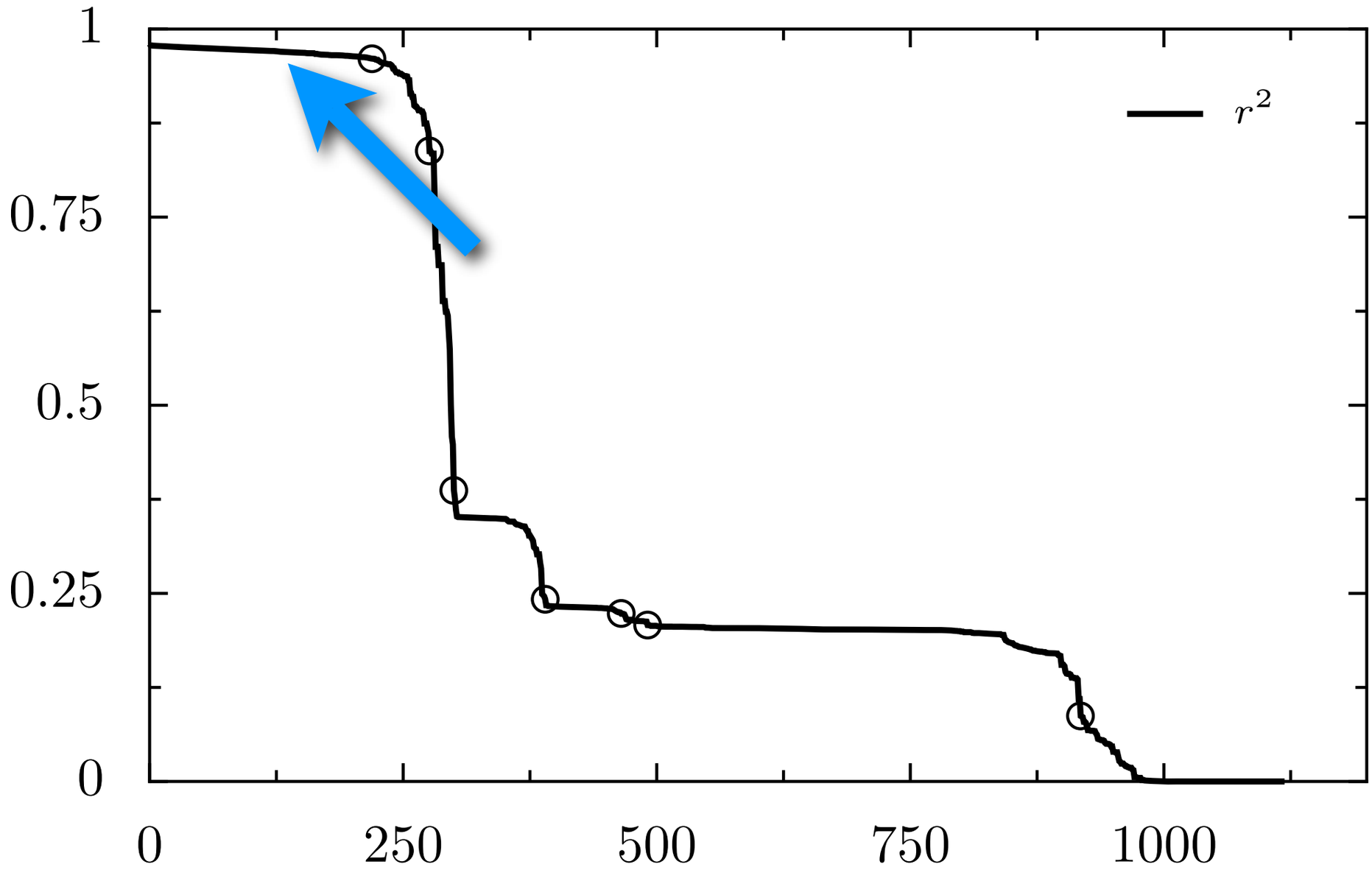
Analyze the space of model structures
Use machine learning techniques to help

Key Idea:

Don't throw away any models
Even the bad ones contain valuable information

Bridewell & Todorovski
(2007), ILP and KCAP

Inspiration for Constraint Induction

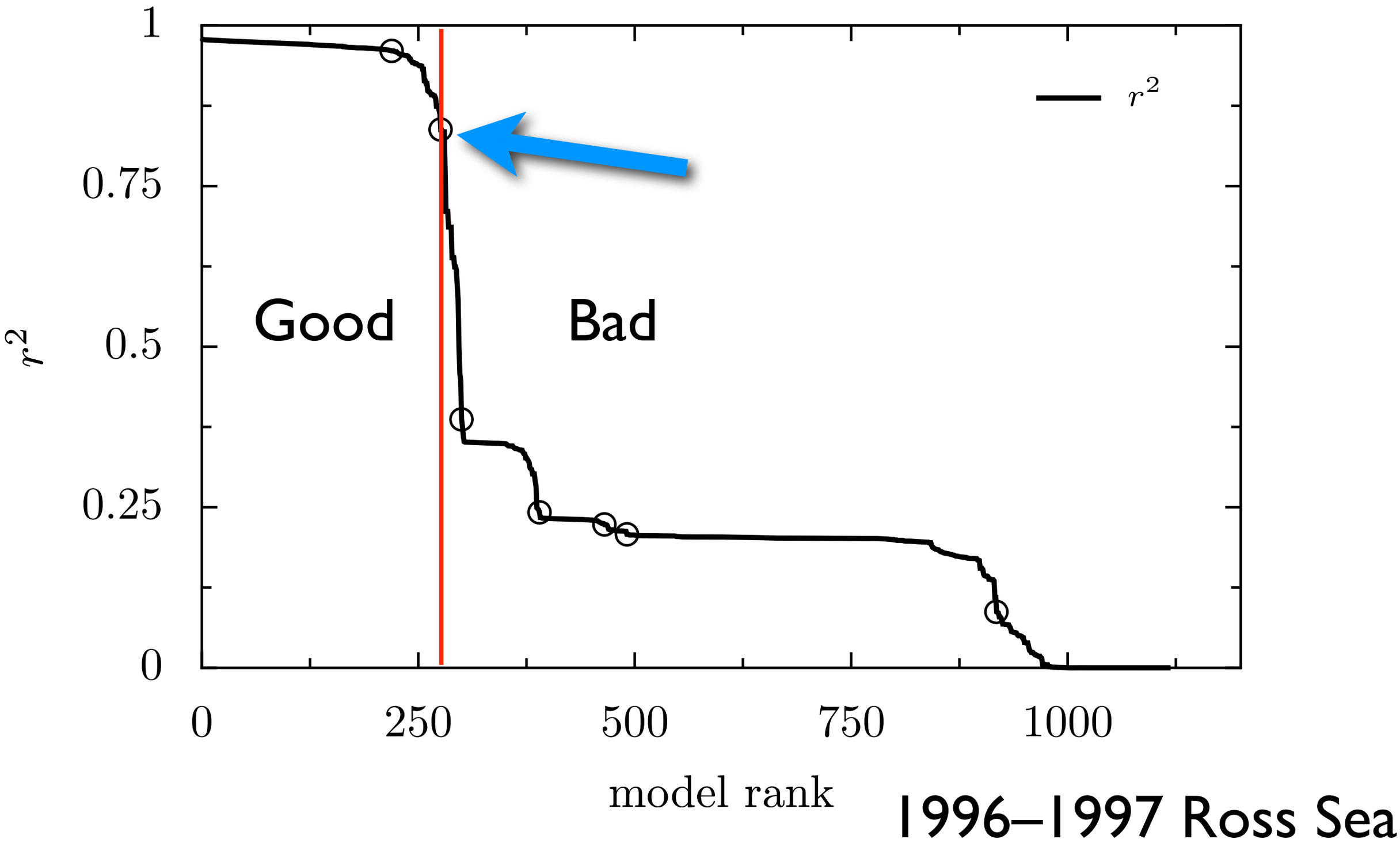


1996–1997 Ross Sea

Learning Constraints

1. Build and parameterize process models
2. Store the models for analysis
3. Formally describe the model structures
4. Identify good and bad models
5. Use ILP to generate descriptions of accurate and inaccurate models
6. Convert descriptions into constraints for IPM

Good and Bad Models



1996–1997 Ross Sea

Descriptive Rules

Run 1: Generate a theory for accurate models.

```
accurate_model(A) :-
   does_not_include_process(A,death_exp2),
   includes_process_entity(A,monod_lim,iron).
```

Run 2: Generate a theory for inaccurate models.

```
inaccurate_model(A) :-
   does_not_include_process_entity(A,monod_2nd,iron),
   does_not_include_process_entity(A,monod_lim,iron).

inaccurate_model(A) :-
   includes_process(A,death_exp2),
   includes_process_entity(A,deangelis_beddington,phytoplankton).
```

We chose Aleph by Ashwin Srinivasan due to its ready availability and capabilities.

Step 6: Convert Theories into Structural Constraints

Rules for accurate models become sufficient conditions for retaining model structures

The negation of rules for inaccurate models become necessary conditions for retaining the model structures.

Type of Transfer

- Subsuming Domain: Aquatic Ecosystems
  - Marine Ecosystem -- Southern Ocean
  - Freshwater Ecosystem -- Bled Lake
- Shared Processes
- Different Environment
  - nutrients, light patterns, ice, temperature
- Different Agents
  - phytoplankton, zooplankton

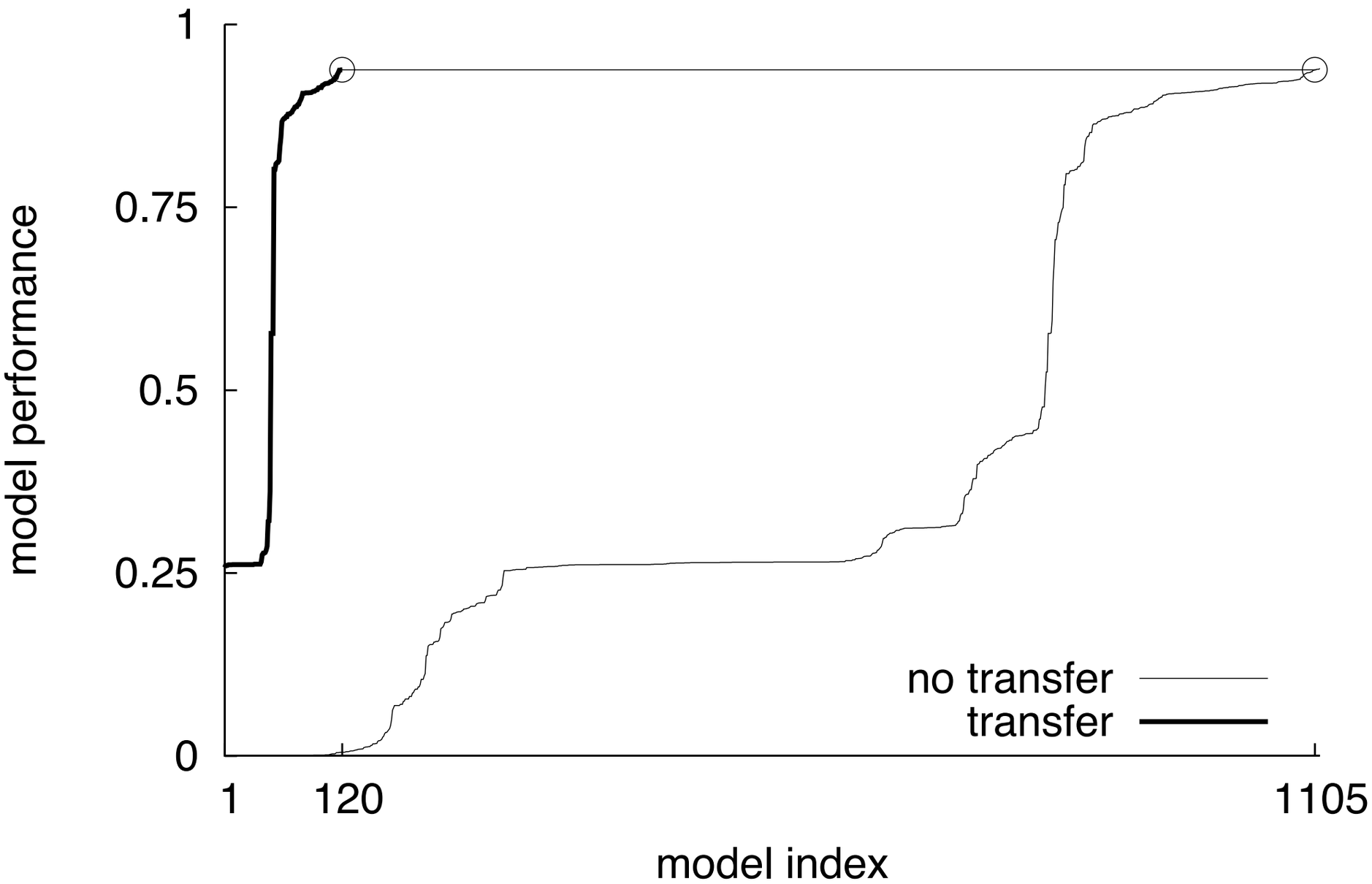
Measures

- Worst Case Modeling scenario

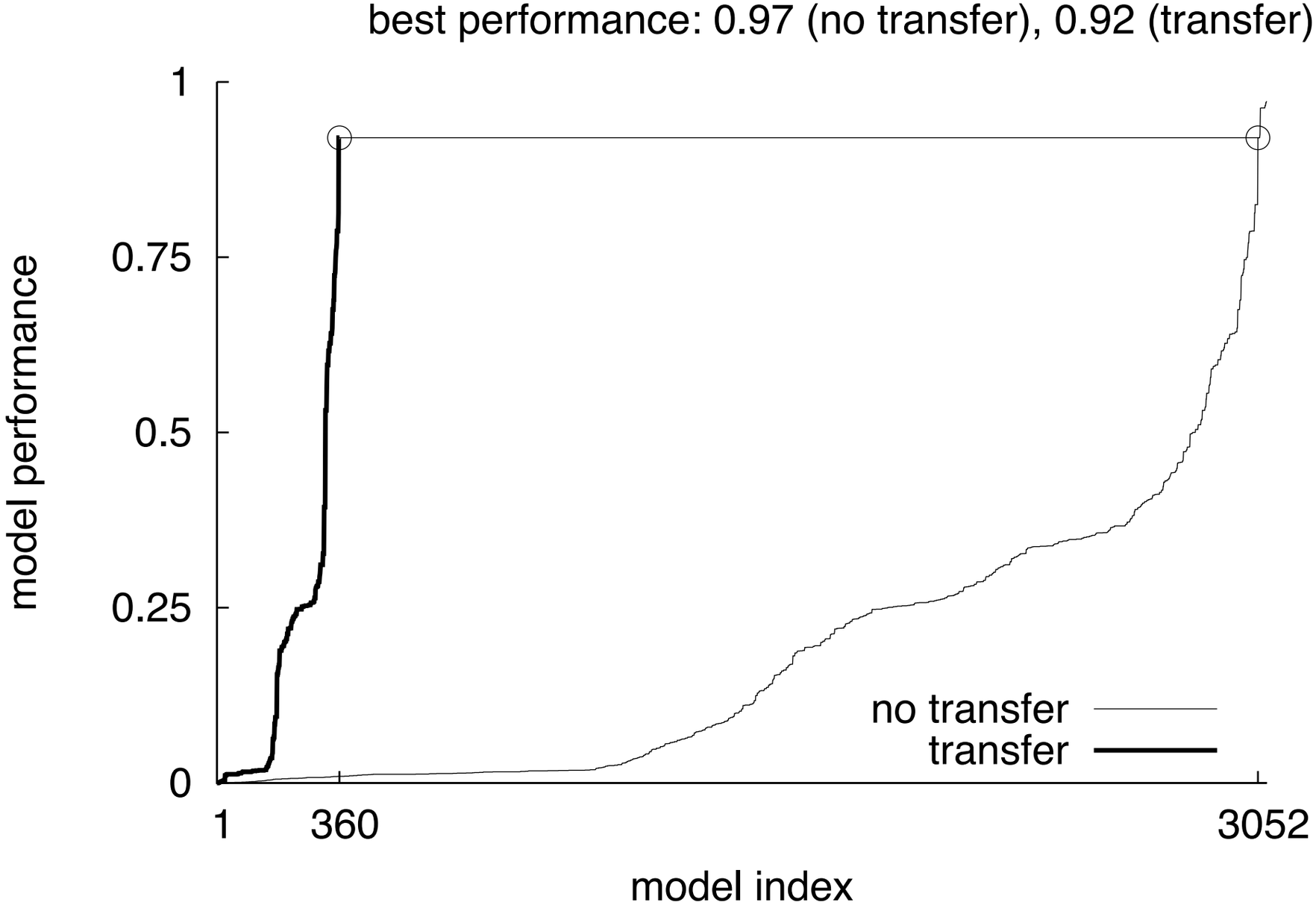
- performance loss (in r^2) due to transfer
  - remember “exhaustive” search

- how much faster do we see the most accurate model in the transfer case vs. no transfer?

Ross-96 to Ross-97

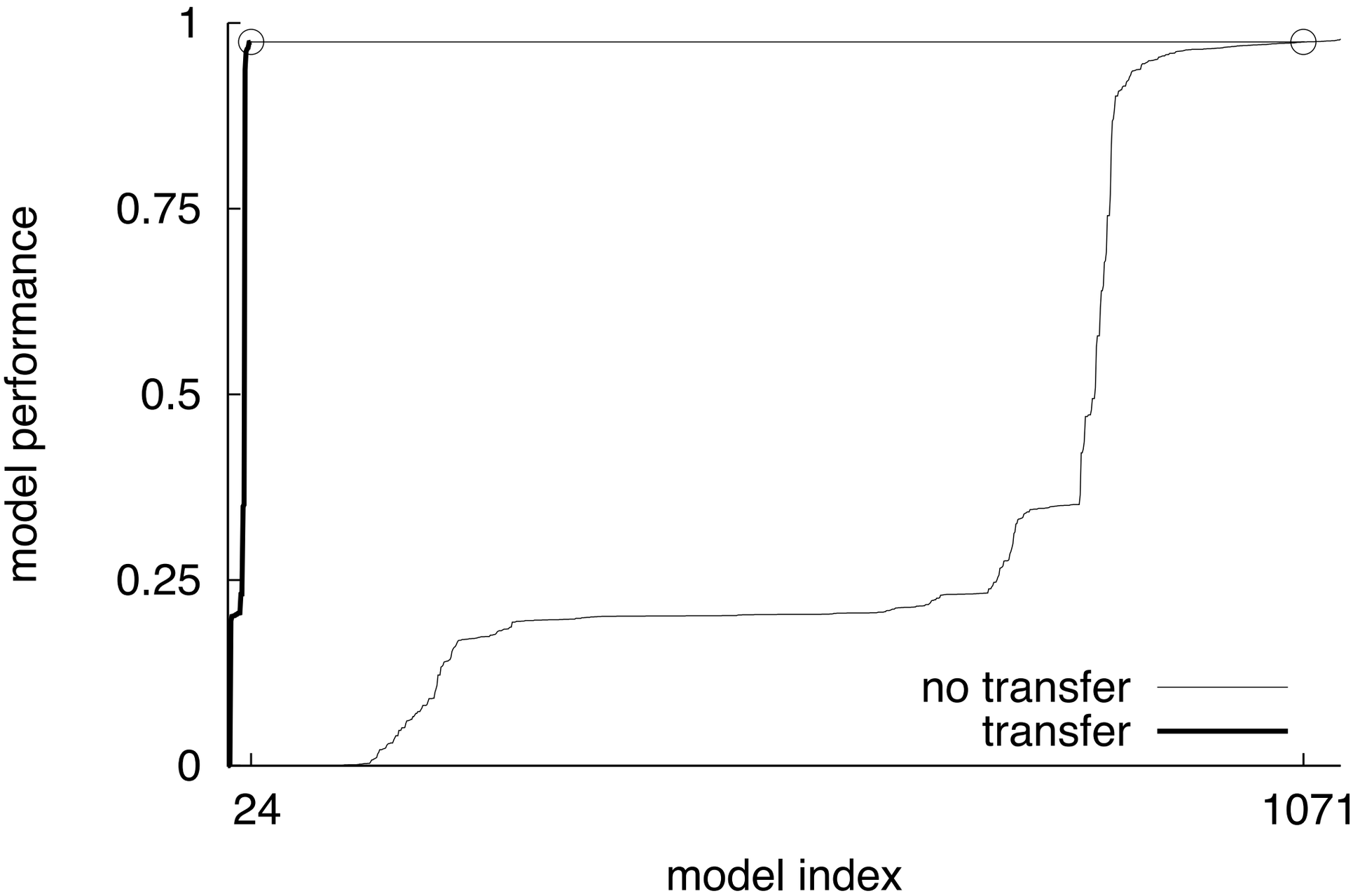


Ross-96 to Bled-02



Bled-02 to Ross-96

best performance: 0.98 (no transfer), 0.97 (transfer)



Related Work

- Inductive Process Modeling and other quantitative modelers
  - IPM (Langley et al., see ICML 2002)
  - HIPM (Todorovski et al., see AAAI 2005)
  - Lagramge (Todorovski & Dzeroski)
  - PRET (Bradley & Stolle)

- Some similar work
  - Learning Constraint Networks (Bessiere et al.)
  - Relational Clichés (Silverstein & Pazzani; Morin & Matwin)
  - Mode Declarations (McCreath & Sharma)
  - Rule Reliability from Prior Performance (Mark Reid)

Future Directions

- Extended analysis of cross-domain transfer

- Incorporating learned constraints into IPM
  - See Bravo et al. in the ICML workshop IPM-07

- Transferring this approach to other tasks
  - Planning, classifier learning, etc.
  - (See me if your work could benefit.)

- Building an integrated creative system