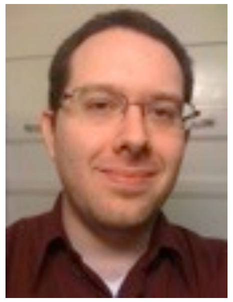
Machine Learning: We Do, and They Don't

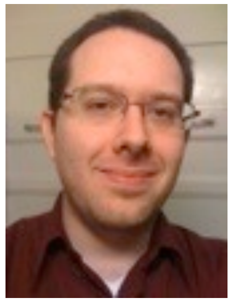
Charles Sutton
Hamming Seminar
University of Edinburgh
15 Feb 2012

When we write programs that "learn",
it turns out that we do and they don't.

--Alan Perlis



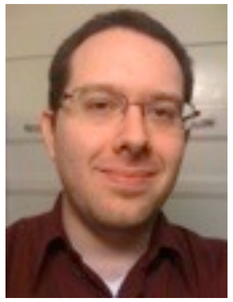
Me



Me

Barista

So, are you finished with exams?

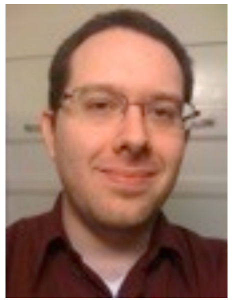


Me

Barista

So, are you finished with exams?

Almost done. I only have a few more to mark.



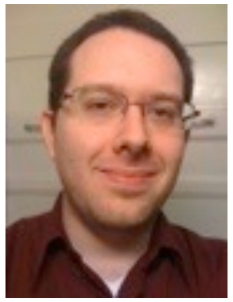
Me

Barista

So, are you finished with exams?

Almost done. I only have a few more to mark.

Oh, you're a lecturer!
What do you do research on?



Me

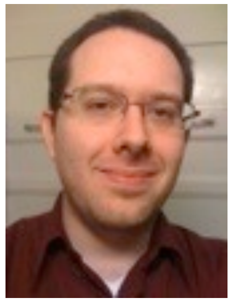
Barista

So, are you finished with exams?

Almost done. I only have a few more to mark.

Oh, you're a lecturer!
What do you do research on?

Oh, I work on artificial intelligence.



Me

Barista

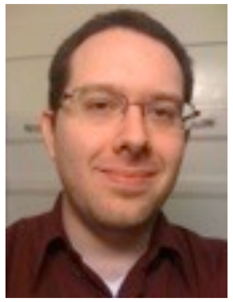
Barista: So, are you finished with exams?

Me: Almost done. I only have a few more to mark.

Barista: Oh, you're a lecturer! What do you do research on?

Me: Oh, I work on artificial intelligence.

Barista: That's really cool!



Me

Barista

So, are you finished with exams?

Almost done. I only have a few more to mark.

Oh, you’re a lecturer!
What do you do research on?

Oh, I work on artificial intelligence.

That’s really cool!

Do you believe in aliens?









$\overset{??}{=}$

Sneaking up on AI

[Photos: stock photos; NIH; jurvetson (Flickr); Michael Linnenbach]

Sneaking up on AI

[Photos: stock photos; NIH; jurvetson (Flickr); Michael Linnenbach]

Sneaking up on AI

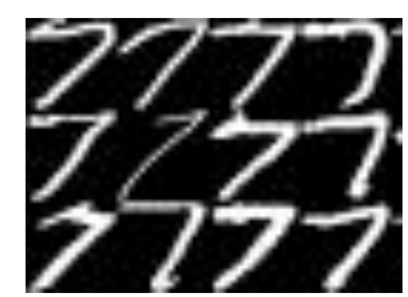
[Photos: stock photos; NIH; jurvetson (Flickr); Michael Linnenbach]

Sneaking up on AI

[Photos: stock photos; NIH; jurvetson (Flickr); Michael Linnenbach]

Sneaking up on AI

Spam filtering



Recognising handwritten digits

[Photos: stock photos; NIH; jurvetson (Flickr); Michael Linnenbach]

Sneaking up on AI

Web search

collaborative filtering

I'll tell you in a minute

Spam filtering

Recognising handwritten digits

[Photos: stock photos; NIH; jurvetson (Flickr); Michael Linnenbach]

Sneaking up on AI

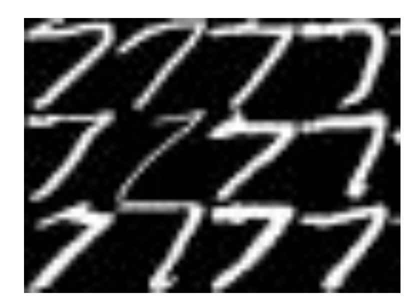
Autonomous driving

Web search

collaborative filtering

I'll tell you in a minute

Spam filtering



Recognising handwritten digits

[Photos: stock photos; NIH; jurvetson (Flickr); Michael Linnenbach]

Sneaking up on AI

Autonomous driving

Web search

collaborative filtering

I'll tell you in a minute

Spam filtering

Recognising handwritten digits

[Photos: stock photos; NIH; jurvetson (Flickr); Michael Linnenbach]

Sneaking up on A

Autonomous driving

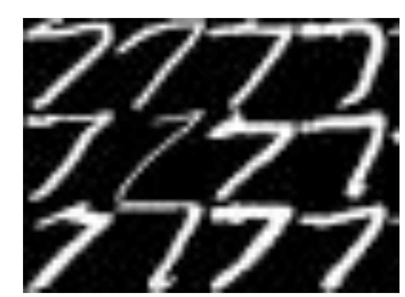
Web search

collaborative filtering

I'll tell you in a minute

Applications motivate new methodology

Spam filtering



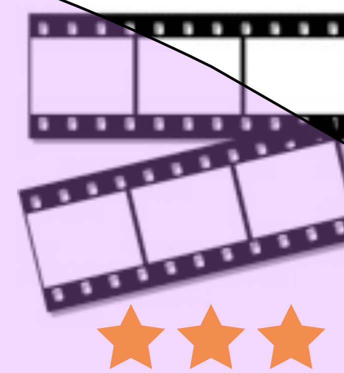
Recognising handwritten digits

[Photos: stock photos; NIH; jurvetson (Flickr); Michael Linnenbach]

I don't know how to get here

Maybe we can push this up

Autonomous driving



collaborative filtering

I'll tell you in a minute

Web search

Tractable region

Spam filtering

Recognising handwritten digits

Two Imperatives

- Choose applications carefully
- Approach applications honestly

Named Entity Recognition

Example Application

October 14, 2002, 4:00 a.m. PT

For years, Microsoft Corporation CEO Bill Gates railed against the economic philosophy of open-source software with Orwellian fervor, denouncing its communal licensing as a "cancer" that stifled technological innovation.

Today, Microsoft claims to "love" the open-source concept, by which software code is made public to encourage improvement and development by outside programmers. Gates himself says Microsoft will gladly disclose its crown jewels--the coveted code behind the Windows operating system--to select customers.

"We can be open source. We love the concept of shared source," said Bill Veghte, a Microsoft VP. "That's a super-important shift for us in terms of code access.“

Richard Stallman, founder of the Free Software Foundation, countered saying…

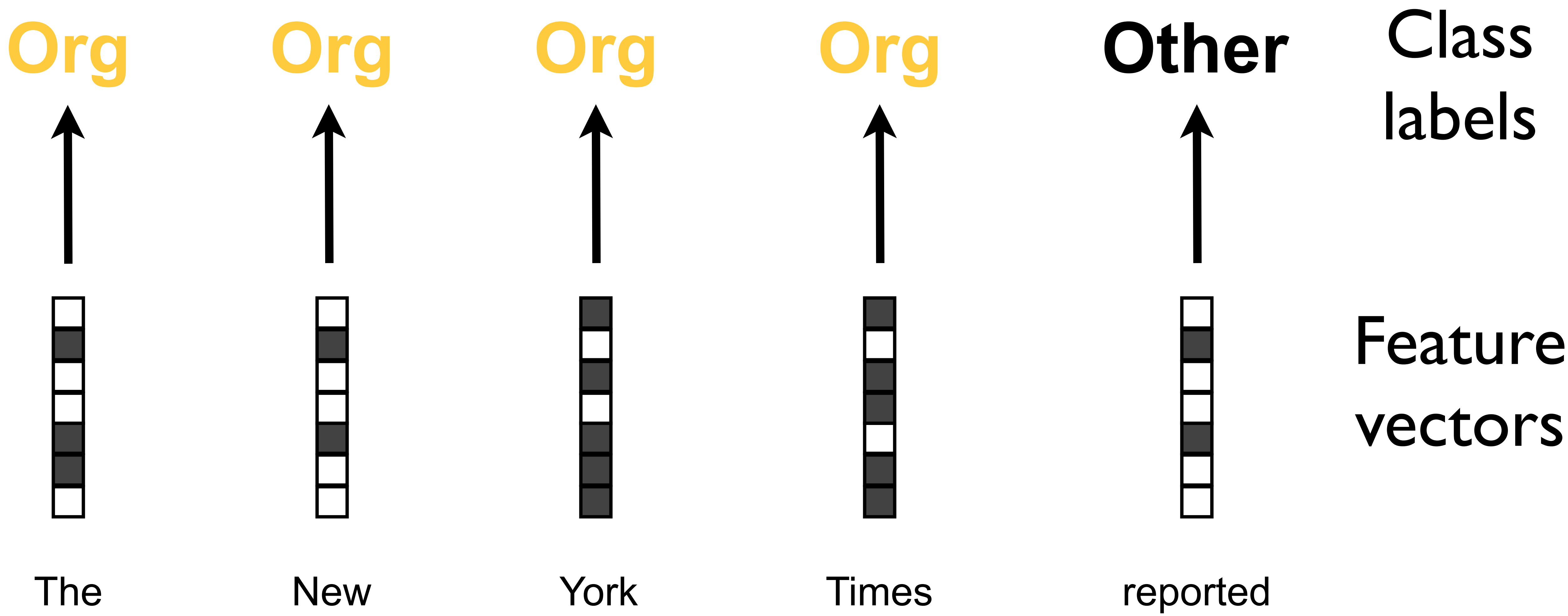
Microsoft Corporation
Bill Gates
Microsoft
Gates
Microsoft
Bill Veghte
Microsoft
Richard Stallman
Free Software Foundation

Labels
ORGANIZATION
PERSON

First step in information extraction

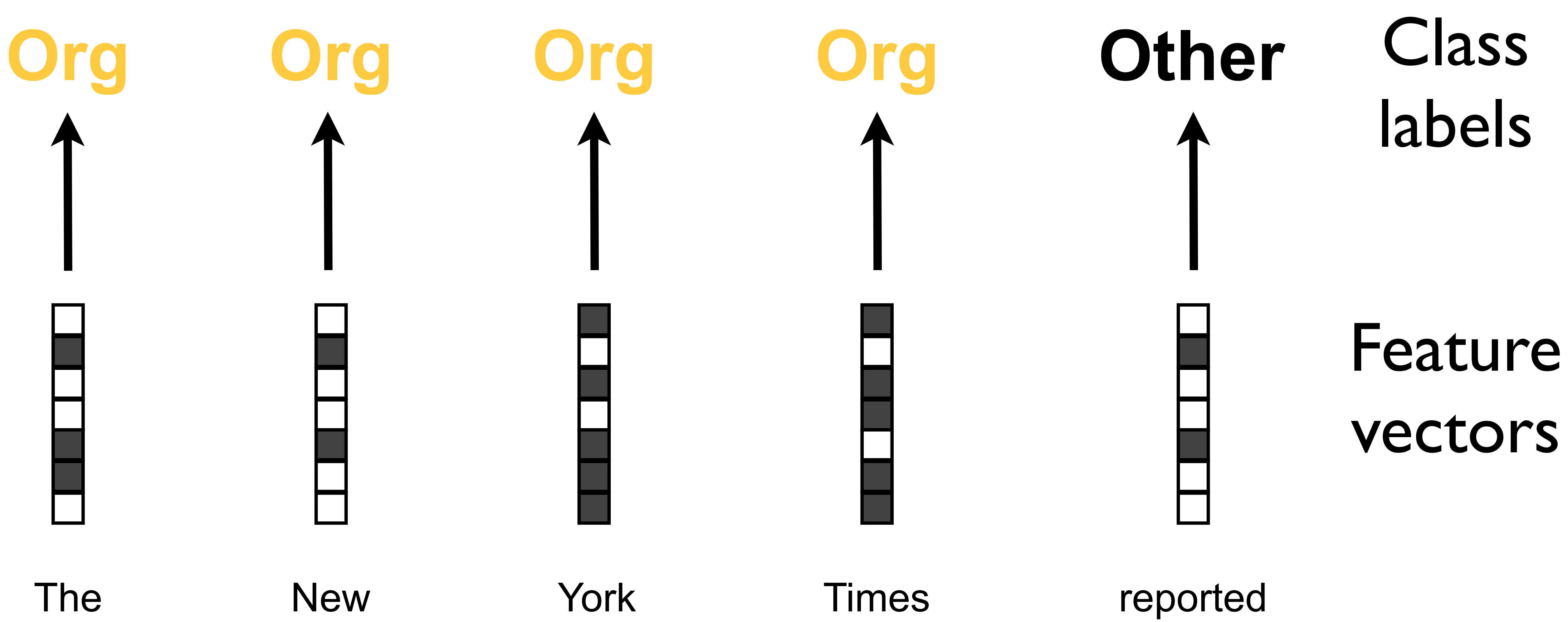
Named Entity Recognition

As Classification



Named Entity Recognition

As Classification



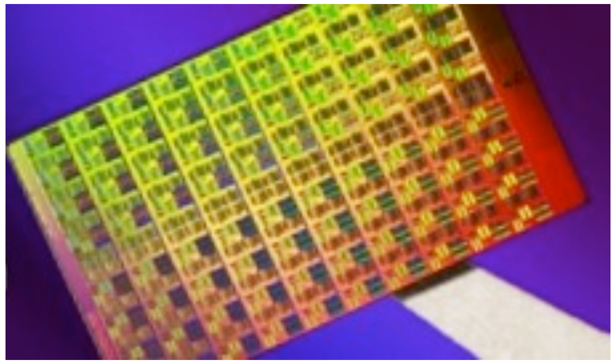
Problem: Labels are interdependent!

Computer System Performance

Another Example Application

Understanding system performance is hard.

- Layered on third-party libraries and frameworks
- Parallelism is mainstream
- Distributed systems of thousands of machines
- Hardware innovation is accelerating



Goal: Understand performance data

Try 1: Cast as regression

Req1 10:33.10am →

Req2 10:33.43am →

Req3 10:34.05am →

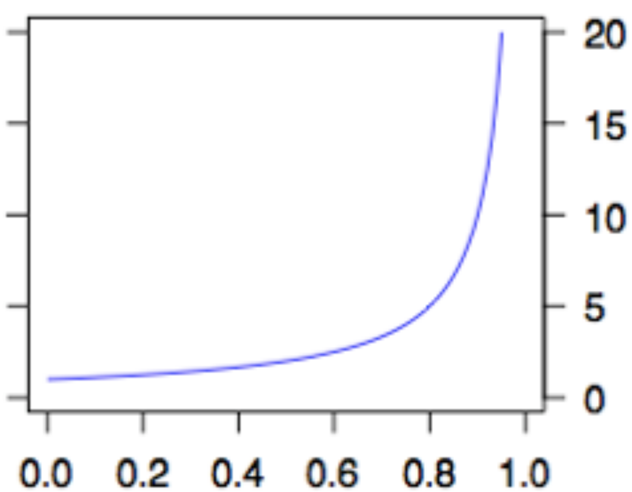
120ms →

213ms →

175ms →

Data

Model



Avg Latency (ms)

Workload (req/s)

Goal: Understand performance data

Try 1: Cast as regression

Req1 10:33.10am →

Req2 10:33.43am →

Req3 10:34.05am →

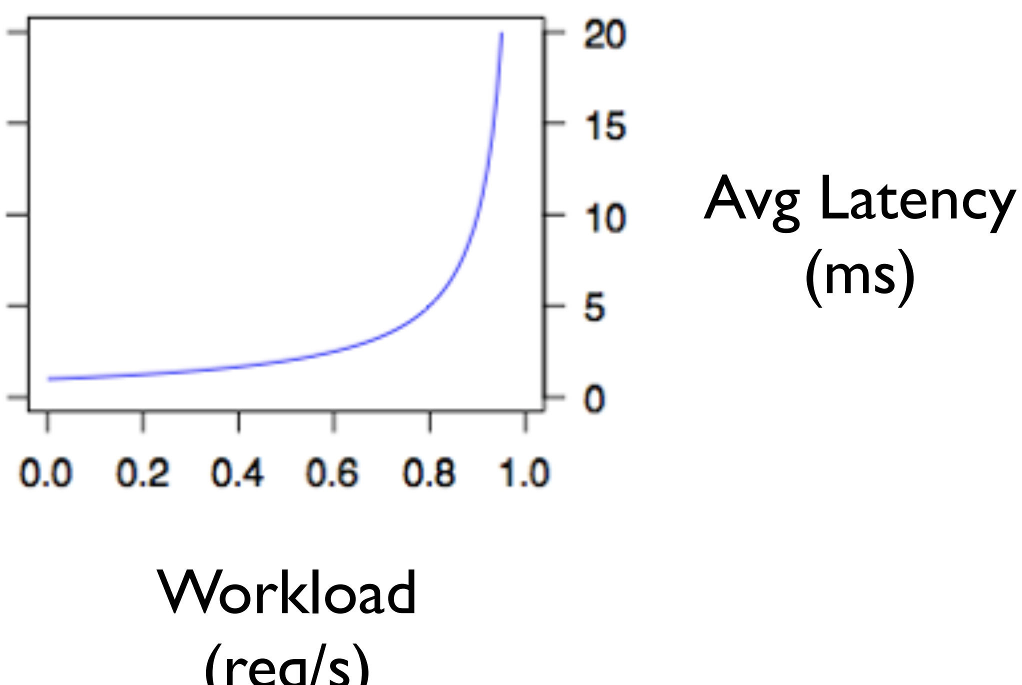
120ms →

213ms →

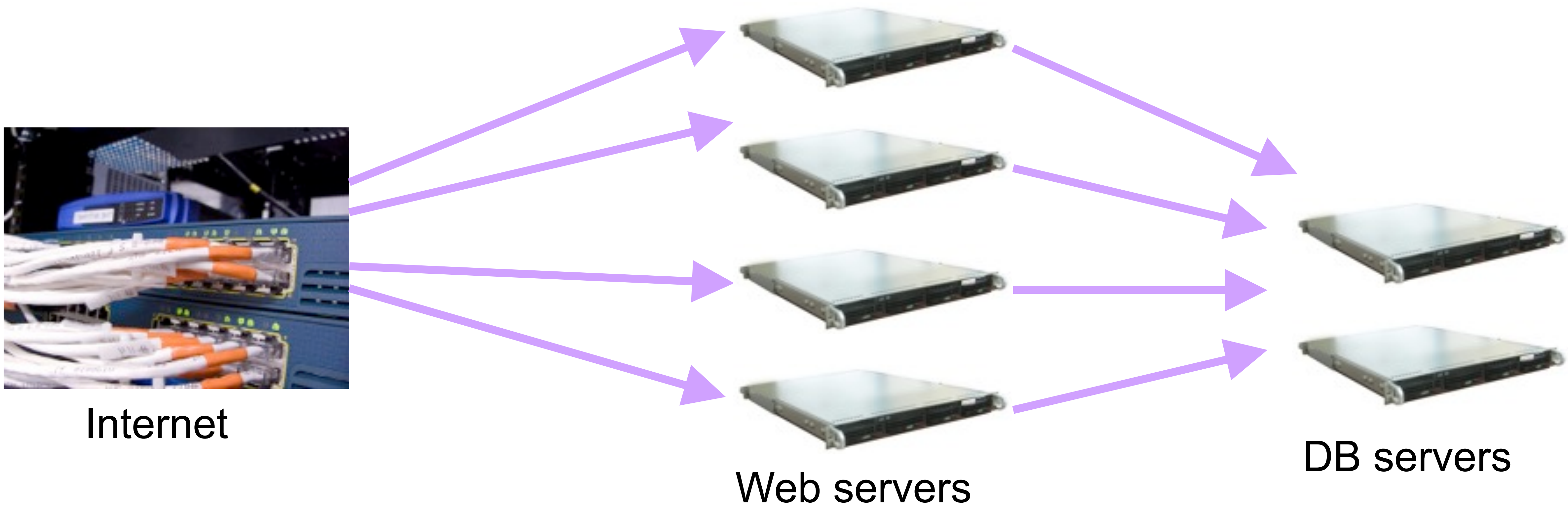
175ms →

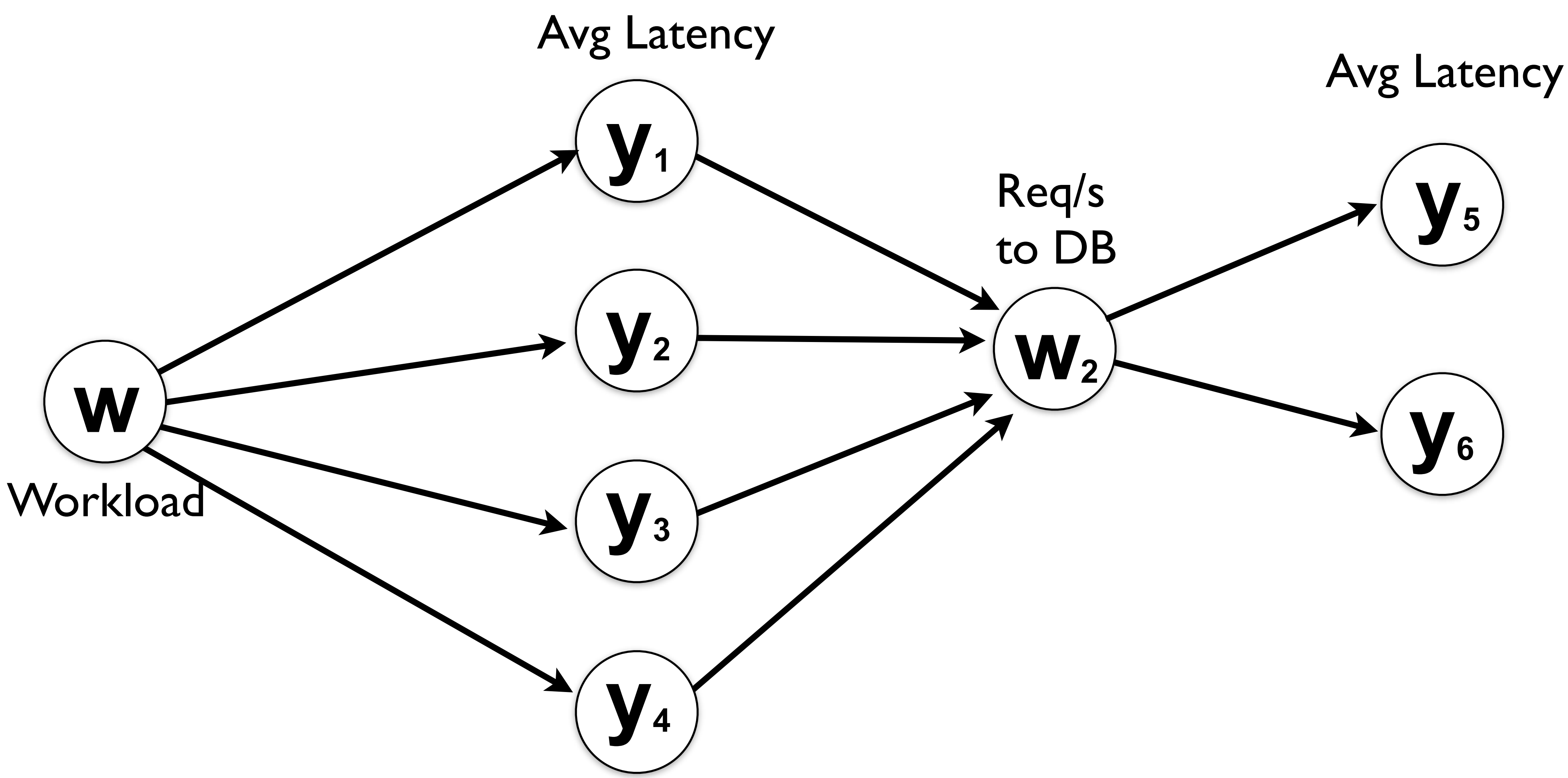
Data

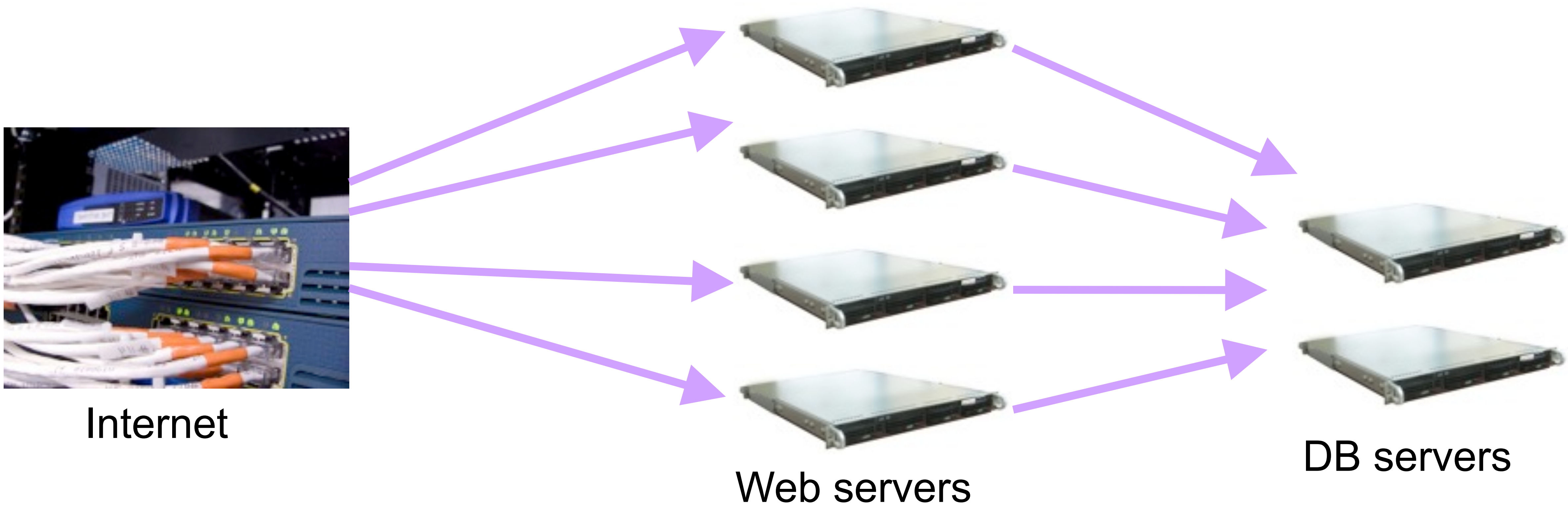
Model

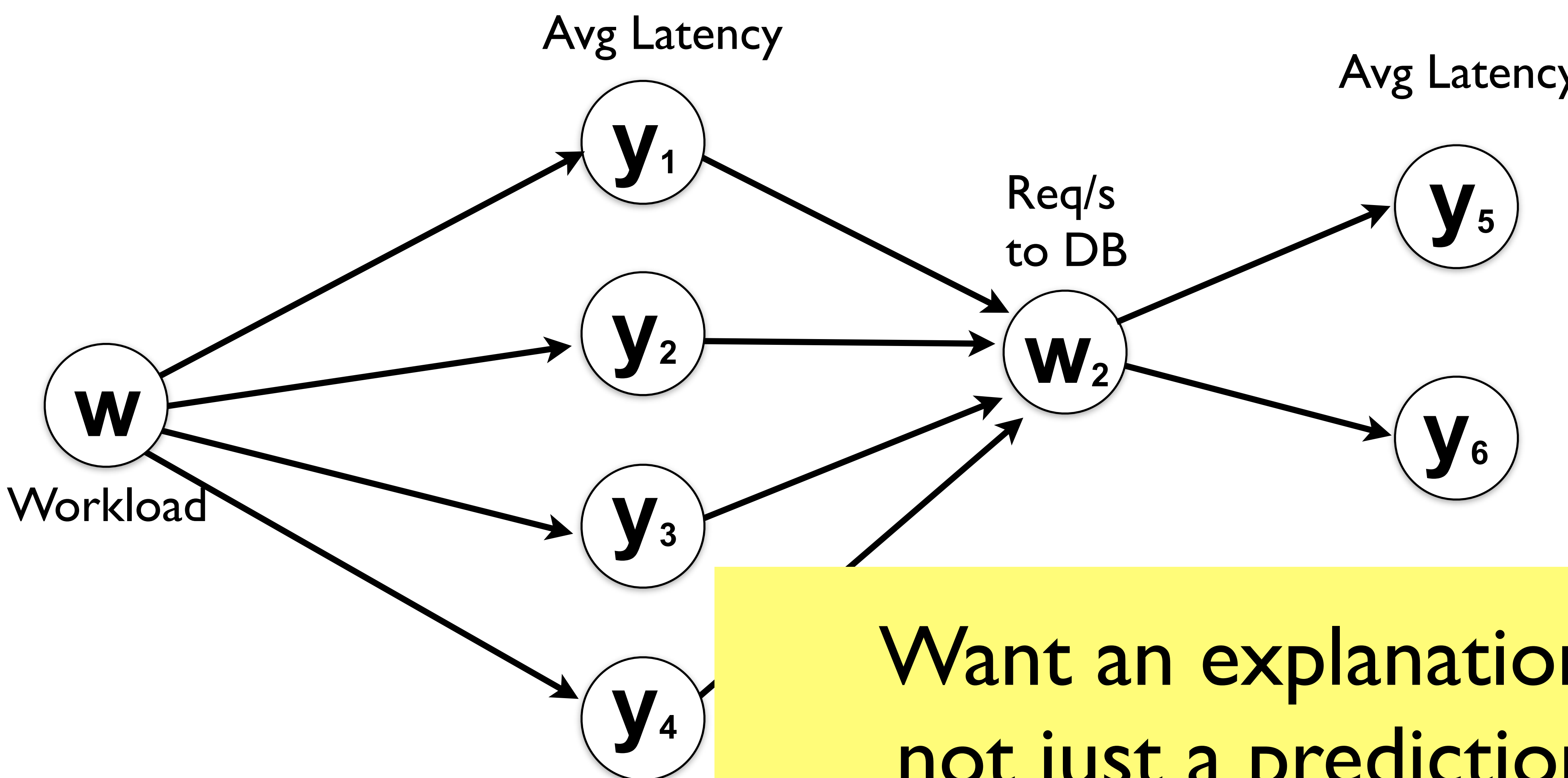


Can't pull apart distributed system

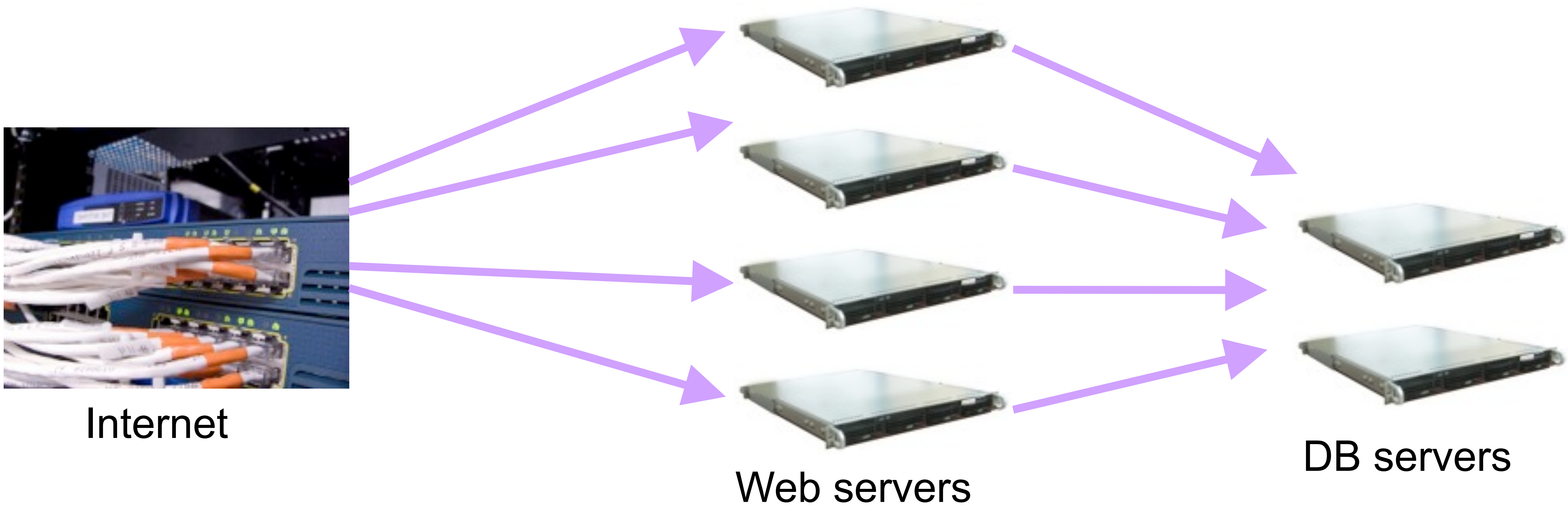

Internet
Web servers
DB servers


Avg Latency
Avg Latency
Req/s
to DB
w
Workload
y_1
y_2
y_3
y_4
w_2
y_5
y_6

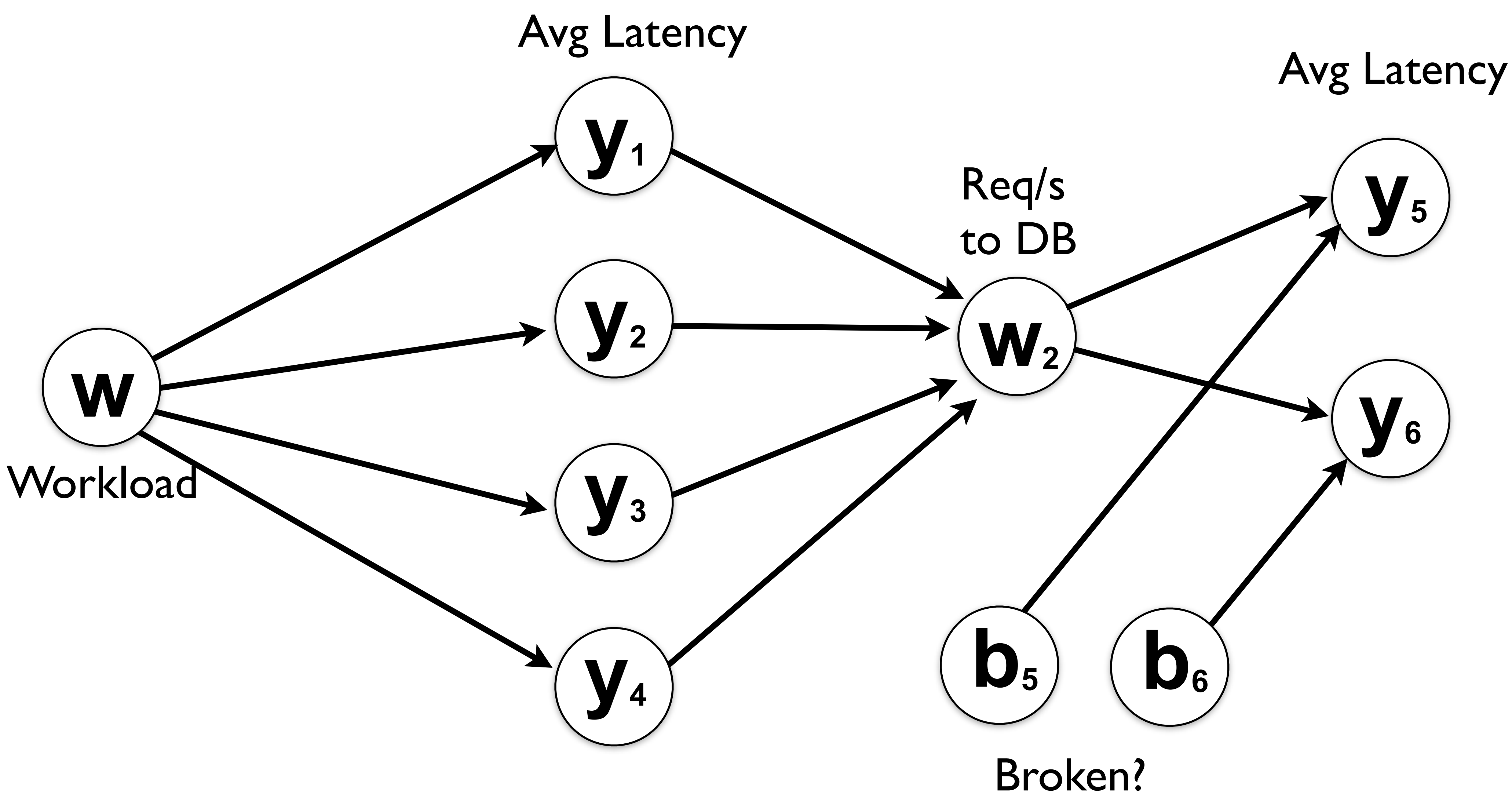




Want an explanation,
not just a prediction



Internet
Web servers
DB servers



Avg Latency
Avg Latency
Req/s
to DB
Workload
Broken?
w
y_1
y_2
y_3
y_4
w_2
y_5
y_6
b_5
b_6

Our Desiderata

- Predict many variables that depend on each other
- Predict "hidden explanations" that are never measured directly

Learning from **uncertain**, **indirect** information

Solution: *Probabilistic models*

Use conditioning to add new information.

Model is

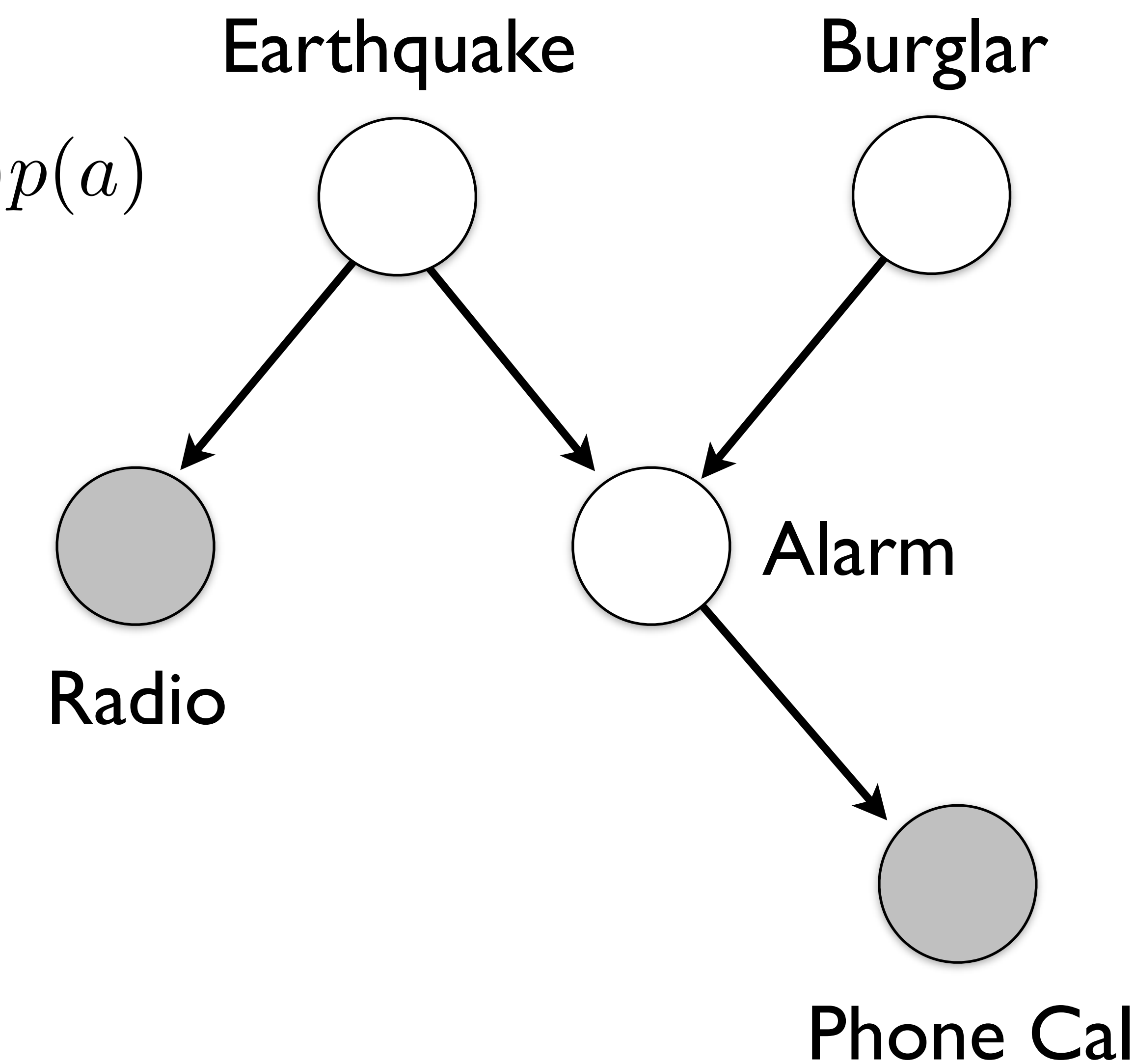
$$p(b, e, a, r, c)$$
$$= p(c|a)p(r|e)p(a|e, b)p(b)p(a)$$

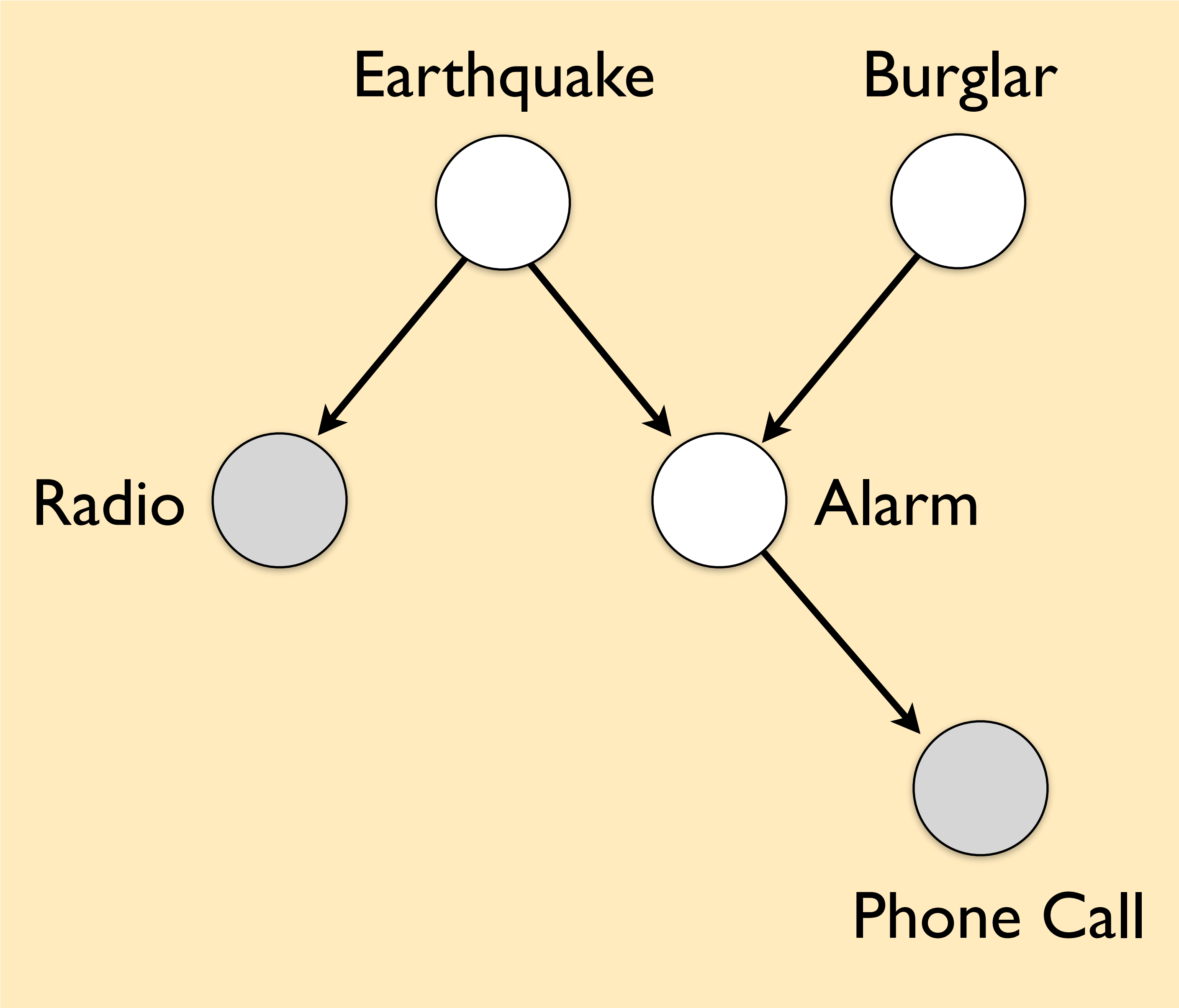
Infer hidden variables using

$$p(b, e, a|r = 1, c = 1)$$

or, for burglary,

$$p(b|r = 1, c = 1)$$





Measurements are

$$c = 1, r = 1$$

We wish to infer values

$$b, e, a$$

Problem: Measurements are noisy, indirect.

Solution: Use posterior distribution

$$p(b, e, a \,|\, c = 1, r = 1)$$

Inference is the problem of computing marginal distributions.

Example

$$p(b|r=1,c=1) \;\; = \frac{p(b|r=1,c=1)}{p(r=1,c=1)}$$

$$= \sum_{a,e} p(a,b,e,r=1,c=1)$$

Exponential time in worst case

Probabilistic model how-to

1. Choose structure of model
2. Choose parameters (learn from data)
3. Observe values of a subset of variables
4. Compute posterior distribution over others using inference
5. Use inference results to answer question of interest

Probabilistic model how-to

1. **Choose structure of model**
2. **Choose parameters (learn from data)**
3. Observe values of a subset of variables
4. Compute posterior distribution over others using inference
5. Use inference results to answer question of interest

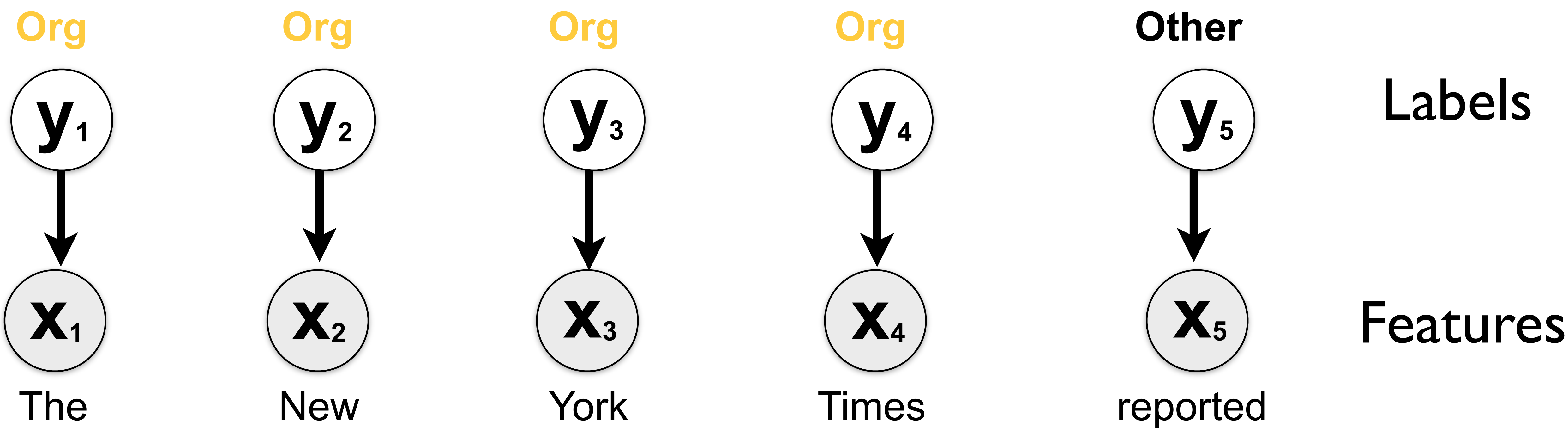
Combines prior knowledge and data

Probabilistic model how-to

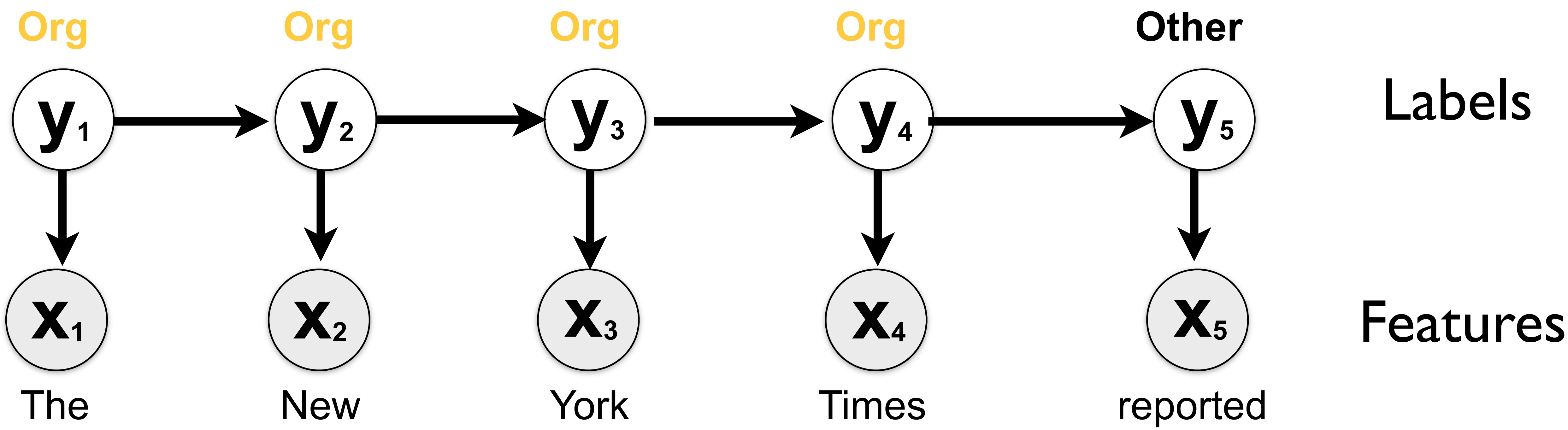
1. Choose structure of model
2. Choose parameters (learn from data)
3. Observe values of a subset of variables
4. **Compute posterior distribution over others using inference**
5. **Use inference results to answer question of interest**

Model forward,
Reason backward

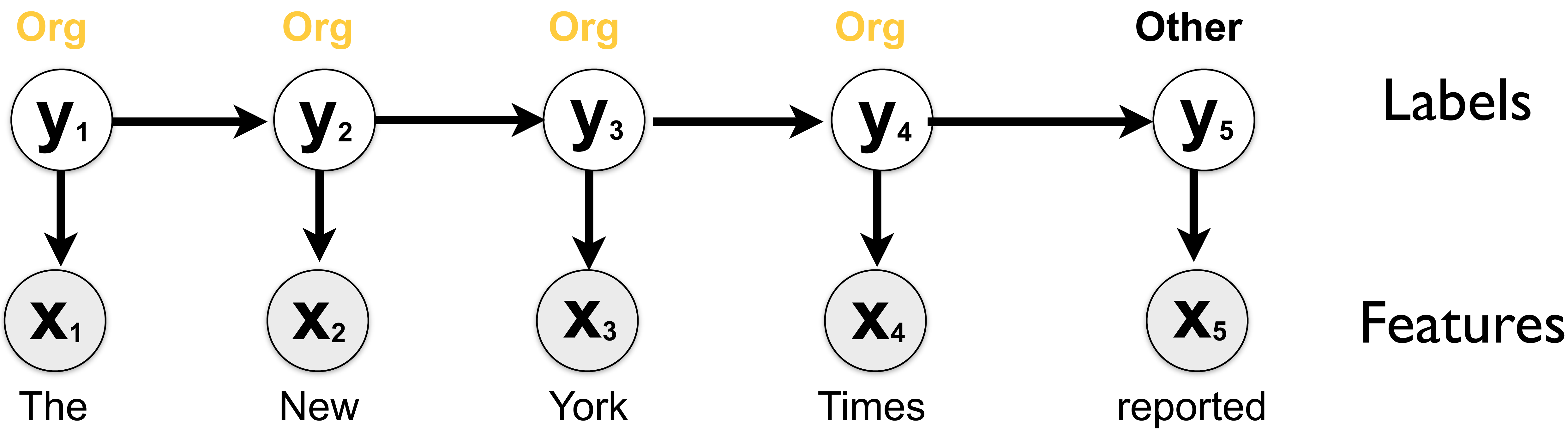
Variables that depend on each other



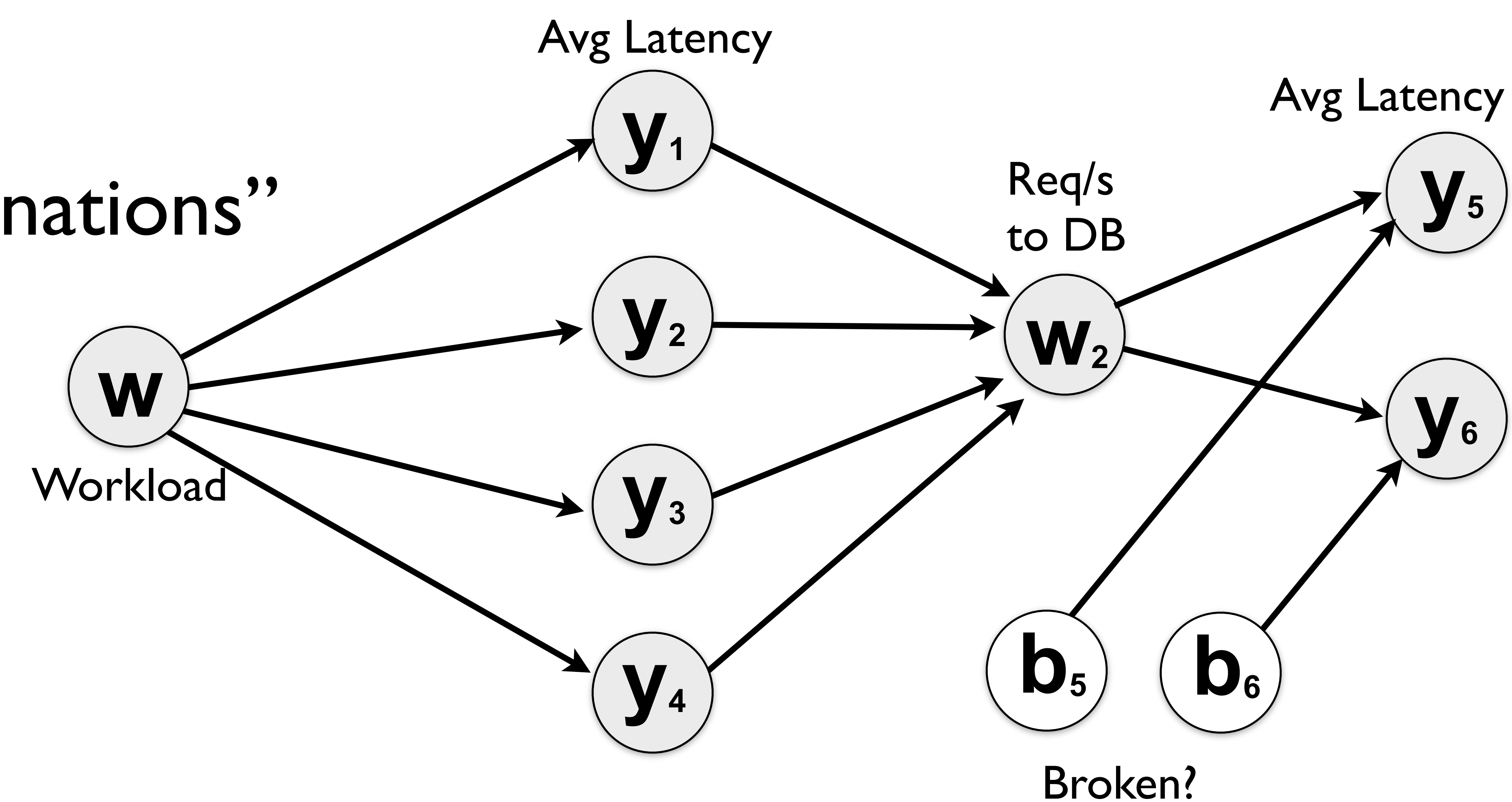
Variables that depend on each other



Variables that depend on each other



“Hidden explanations”



The Probabilistic Modelling Viewpoint

What we want

- Learn from **uncertain**, **indirect** information
- Predict many variables that depend on each other
- Predict "hidden explanations" that are never measured directly

How we get it

- Model forward from explanations to effects (using prior knowledge)
- Refine model by matching it to data
- Reason backward to explanations

So what *are* the main open problems in machine learning?

Four Open Areas

I. Learning at Scale

II. Exploiting Synergy In Learning

III. Learning Structures

IV. Our Insidious Inability to Divide and Conquer

Scale

Much modern data is *streaming*

Blog data (e.g., Twitter), online advertising, AI

Why hard? Consider classification

x	y

(New, **Org**)

(York, **Org**)

(Times, **Org**)

(reported, **Other**)

Progress: 500 Mfeatures/s on 1000 machines

[Agarwal, Chappelle, Dudik, Langford, 2012]

Scale

Much modern data is *streaming*

Blog data (e.g., Twitter), online advertising, AI

Why hard? Consider classification

	x	y
Machine A	New	**Org**
	York	**Org**
Machine B	Times	**Org**
	reported	**Other**

Progress: 500 Mfeatures/s on 1000 machines

[Agarwal, Chappelle, Dudik, Langford, 2012]

Scale

Much modern data is *streaming*

Blog data (e.g., Twitter), online advertising, AI

Why hard? Consider classification

	x	y		
Machine A	(New,	Org)	→	**Classifier A**
	(York,	Org)		
Machine B	(Times,	Org)	→	**Classifier B**
	(reported,	**Other**)		

Progress: 500 Mfeatures/s on 1000 machines

[Agarwal, Chappelle, Dudik, Langford, 2012]

Scale

Much modern data is *streaming*

Blog data (e.g., Twitter), online advertising, AI

Why hard? Consider classification

	x	y		
Machine A	(New, **Org**)		**Classifier A**	Combine & retrain
	(York, **Org**)			
Machine B	(Times, **Org**)		**Classifier B**	
	(reported, **Other**)			

Progress: 500 Mfeatures/s on 1000 machines

[Agarwal, Chappelle, Dudik, Langford, 2012]

Scale

Much modern data is *streaming*

Blog data (e.g., Twitter), online advertising, AI

Why hard? Consider classification

Bottleneck.
Destroys scalability

	x	**y**		
Machine A	(New, **Org**)		**Classifier A**	Combine & retrain
	(York, **Org**)			
Machine B	(Times, **Org**)		**Classifier B**	
	(reported, **Other**)			

Progress: 500 Mfeatures/s on 1000 machines

[Agarwal, Chappelle, Dudik, Langford, 2012]

Scale

Much modern data is *streaming*

Blog data (e.g., Twitter), online advertising, AI

Why hard? Consider classification

Bottleneck.
Destroys scalability

	x	y
Machine A	(New, Org)	
	(York, Org)	
Mach	(Times, Org)	

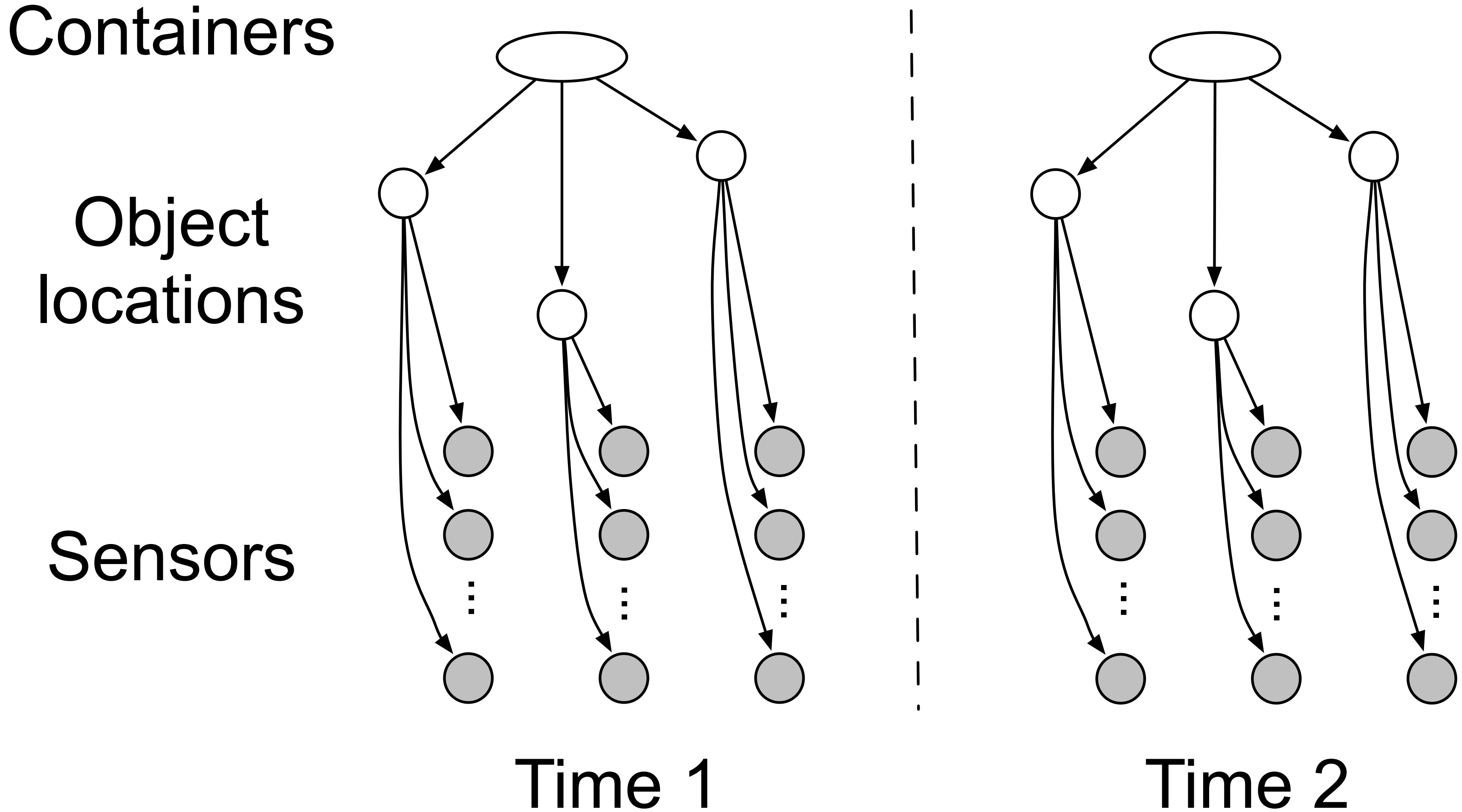
Classifier A → Combine & retrain

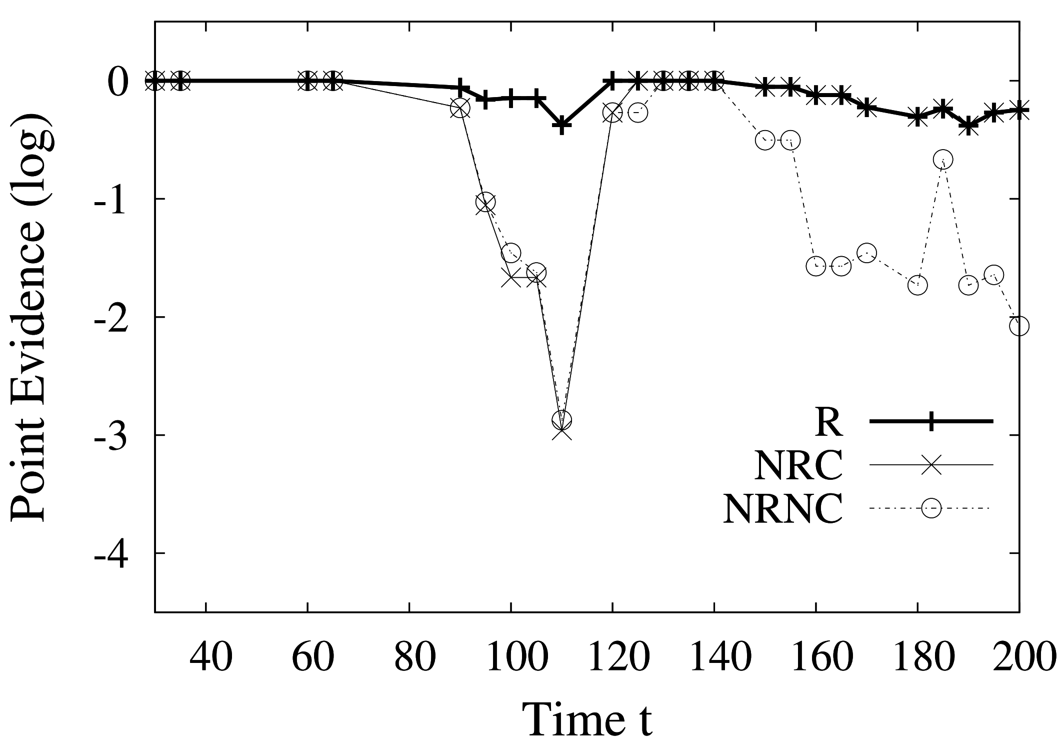
Classifier

Probabilistic inference doesn't scale this well!

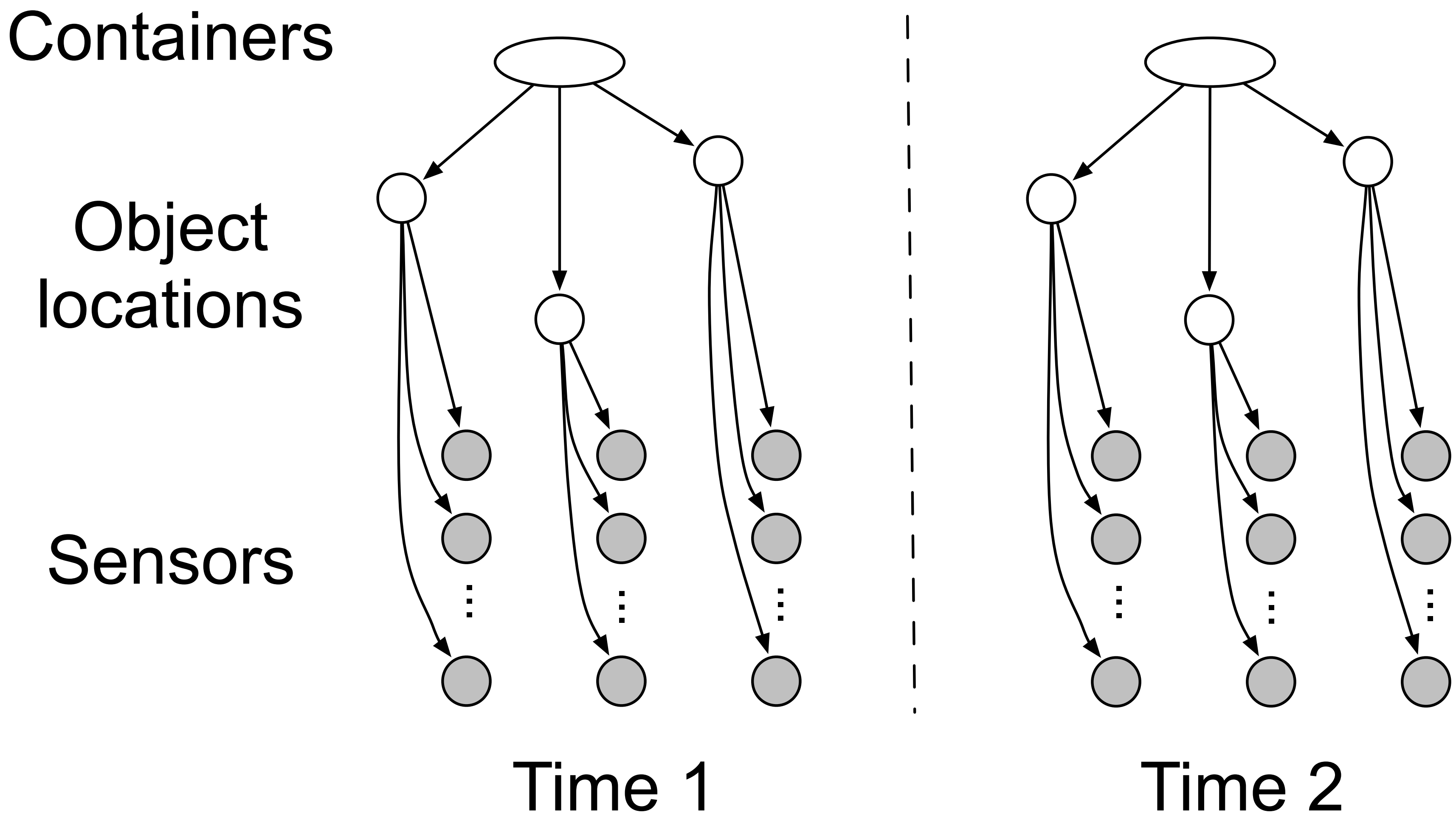
Progress: 500 Mfeatures/s on 1000 machines

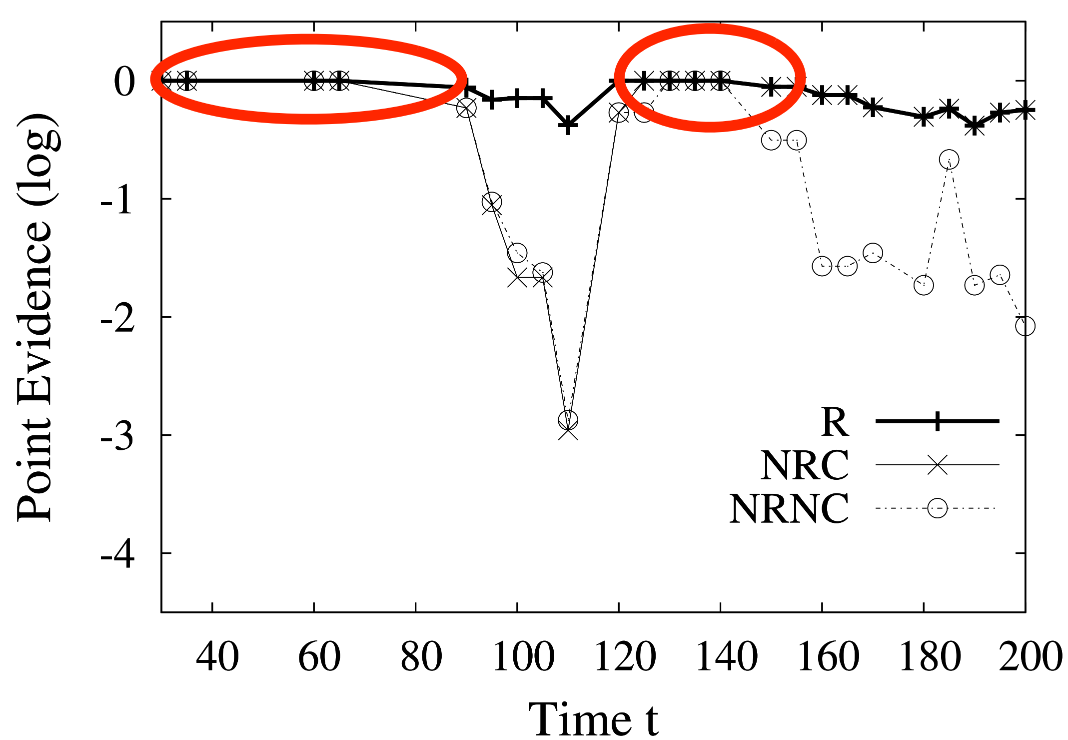
[Agarwal, Chappelle, Dudik, Langford, 2012]





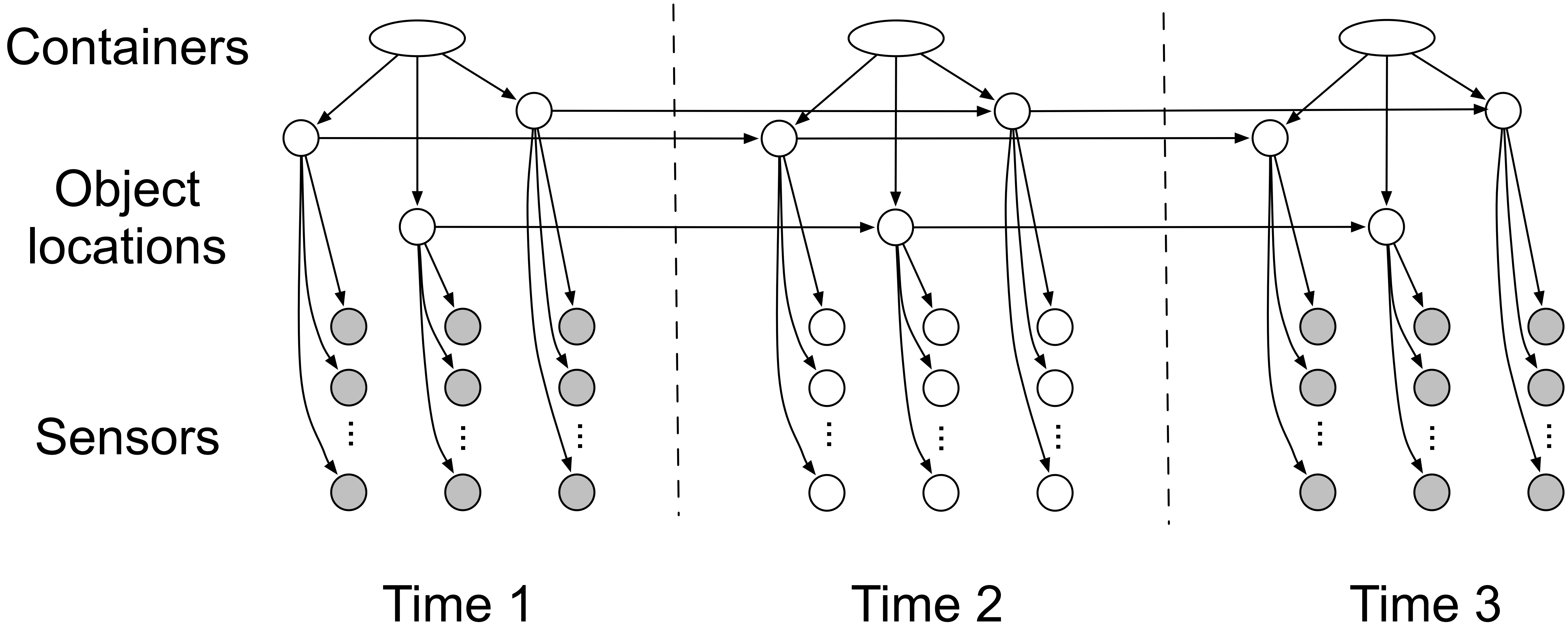
[Cao, Sutton, Diao, Shenoy, PVLDB 2011]





Optimization:
Don't process uninformative observations

[Cao, Sutton, Diao, Shenoy, PVLDB 2011]

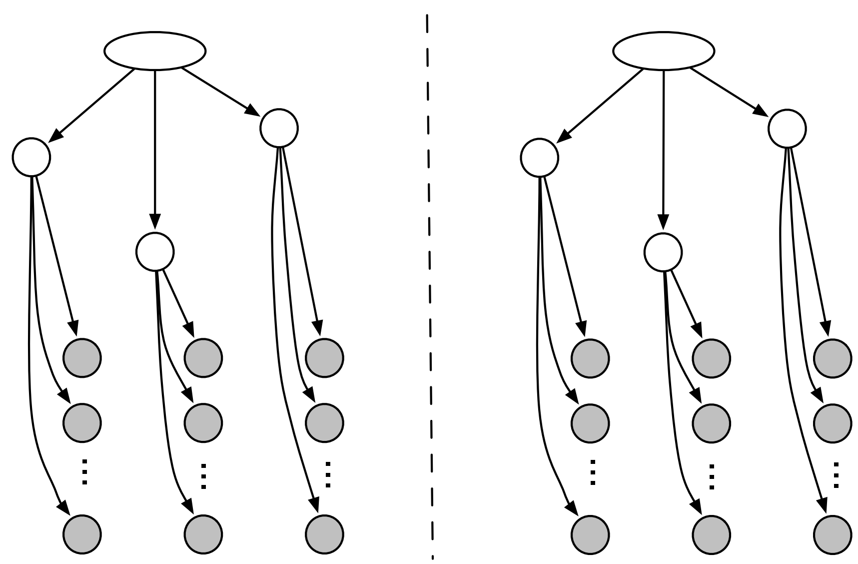


Question:

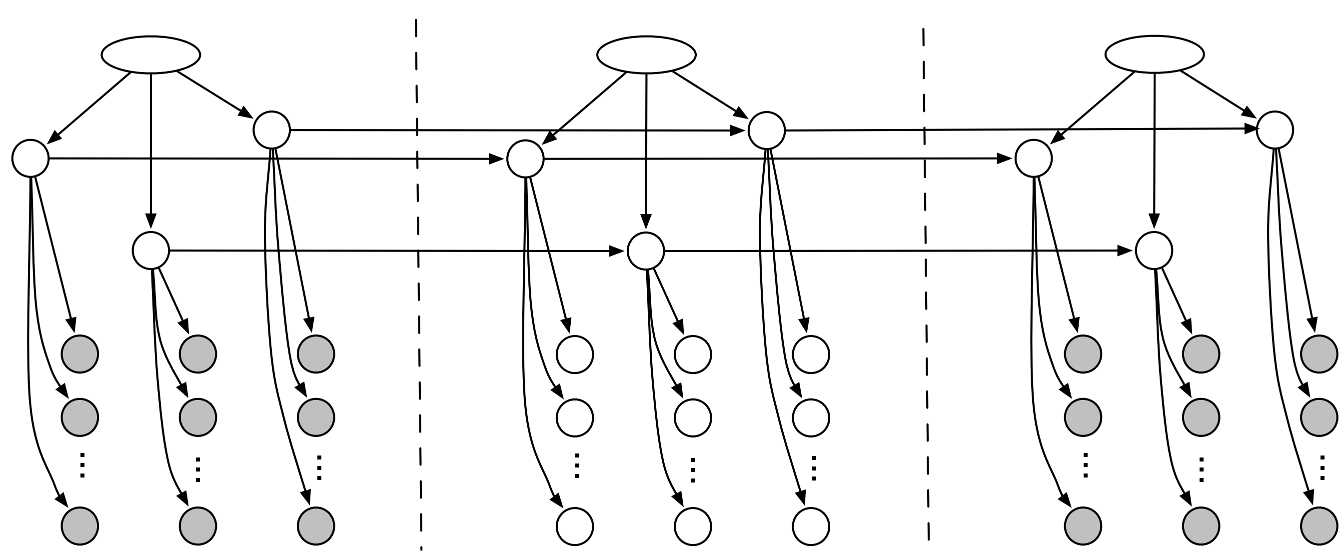
A general principled method
for ignoring uninformative data?

Question:

How to combine reflection and reaction for learning?



Real-time version
(Runs at stream speed)



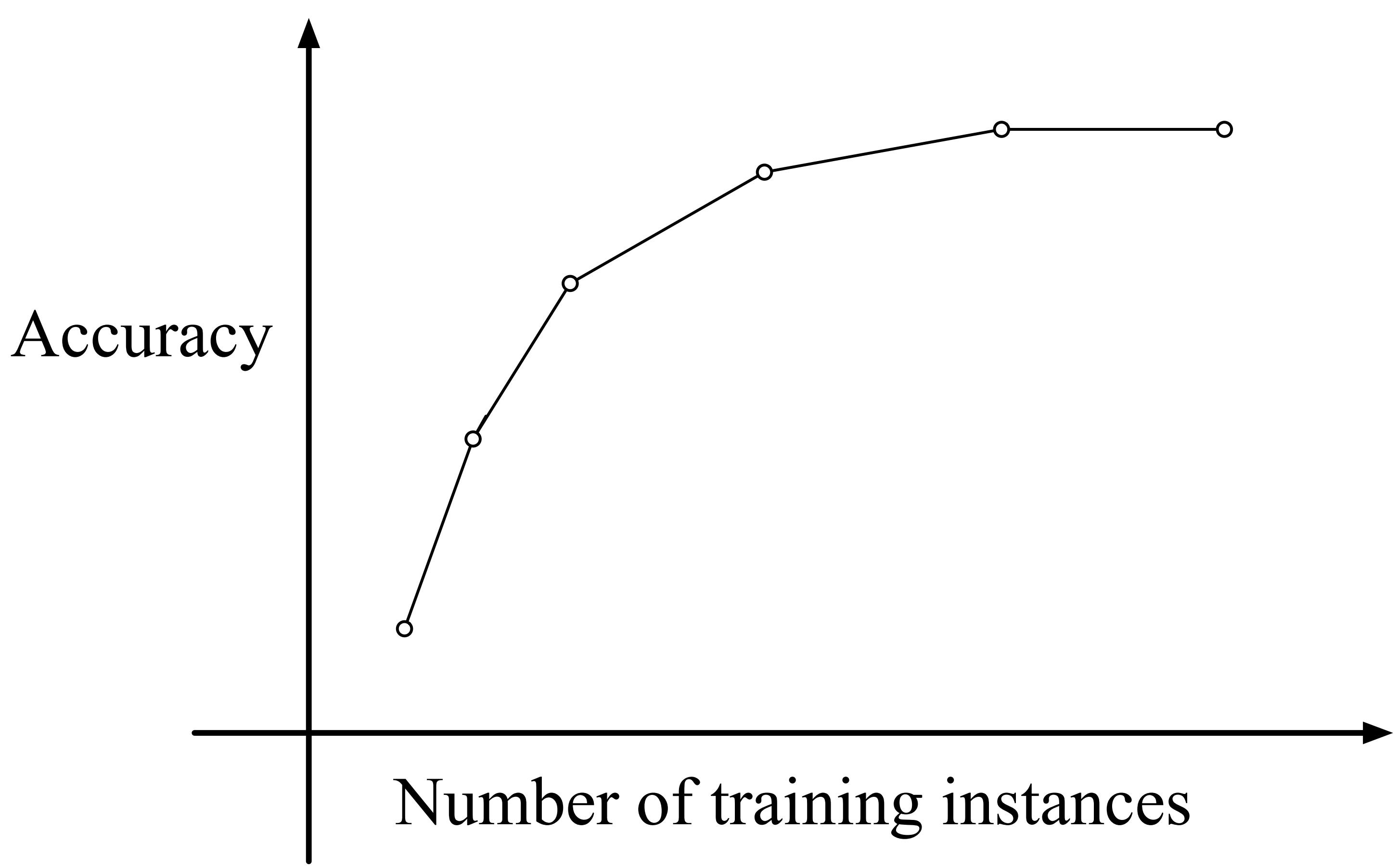
Reflective version
(Takes its time)

Synergy

Transfer Learning, Domain Adaption, Lifelong Learning, Learning to learn, Multitask Learning

Humans: The more we learn, the more we can learn

Machines:



The more they learn, the less they have left to learn

Customer 1: “Spacebook”

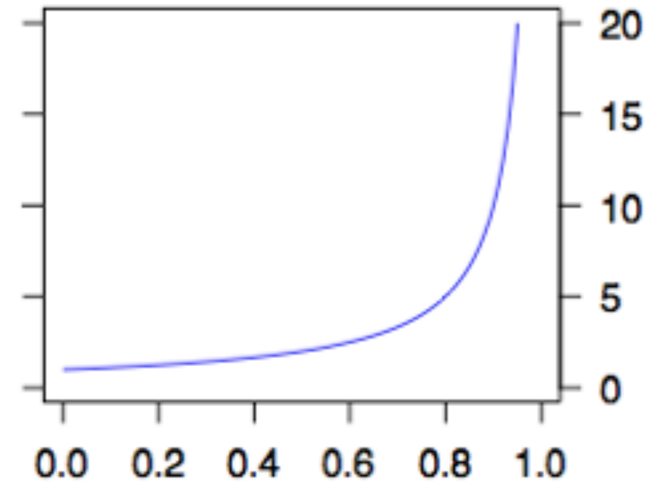
Model

Req1 10:33.10am

Req2 10:33.43am

120ms

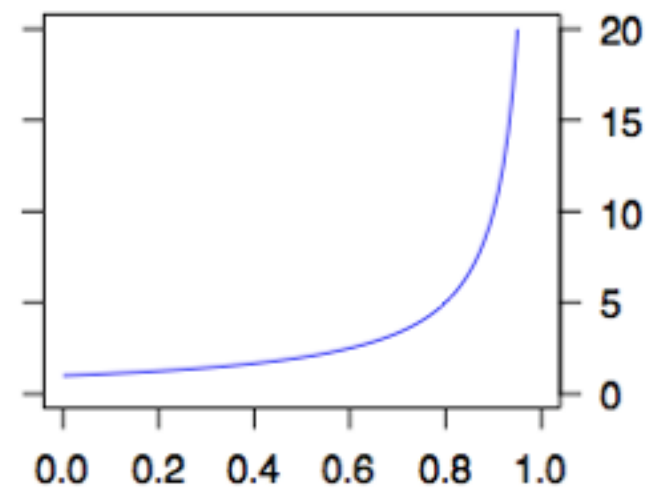
213ms



Customer 2: “Big Batch Job”

100% CPU

2 weeks



Customer 3: “Search Start-Up”

????

Transfer Learning

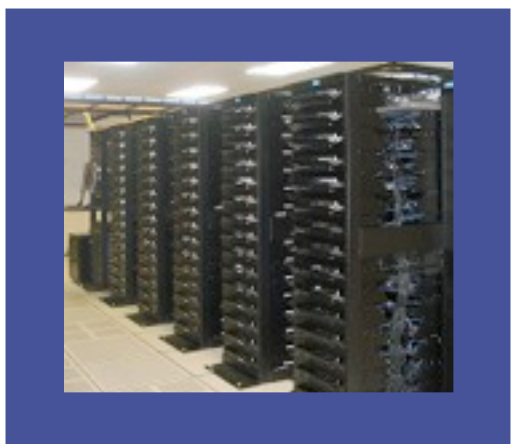
Main ideas out there:

- Reweight instances
- Reweight features
- Couple parameters
- Shared feature representation

Approach 1: Couple parameters

y running time
x features of workload
w parameters of model (to learn)

Task 1



$$y_1 = \sum_{k=1}^{K} w_k^{(1)} x_k$$

Task 2

$$y_2 = \sum_{k=1}^{K} w_k^{(2)} x_k$$

Approach 1: Couple parameters

y running time
x features of workload
w parameters of model
(to learn)

Task 1

$$y_1 = \sum_{k=1}^{K} w_k^{(1)} x_k$$

Main idea: Choose

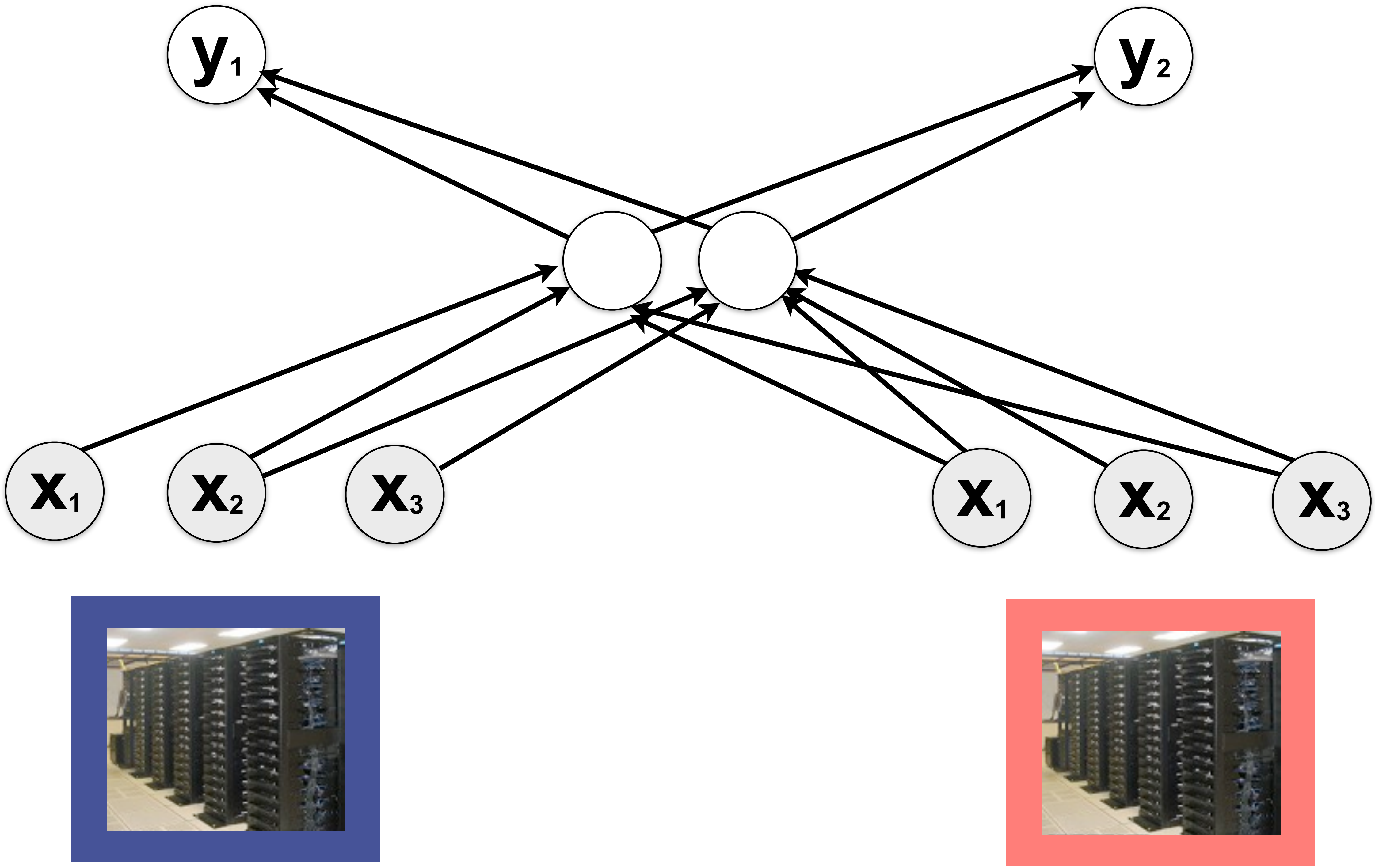
$$w_1^{(1)} \ldots w_K^{(1)}, w_1^{(2)} \ldots w_K^{(2)}$$

good at predicting training data
and
$$w_k^{(1)}, w_k^{(2)}$$
not far apart

Task 2

$$y_2 = \sum_{k=1}^{K} w_k^{(2)} x_k$$

Approach 2: Learn Subtasks



Task 1

Task 2

[Caruana, 1997]

Synergy

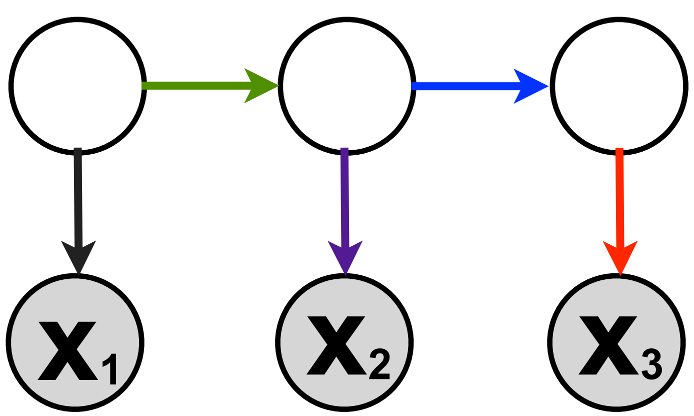
Can work with relatively homogenous task
Not in general use.

- What other sorts of information can be transferred between learning problems?
- Can this be done at large scale with a diverse set of learning problems?
- What would it take to have transfer learning usable by dummies? e.g., in Weka?

Divide and Conquer

In complex domains, all parameters interact.
Learning does not have a divide-and-conquer principle.

Example:



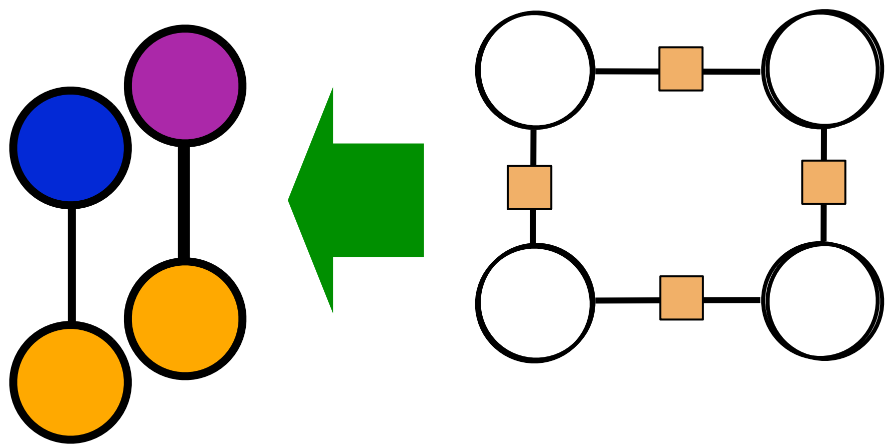
Any change to **red distribution** affects all three predictions

May need to change **green** to compensate

"Learning" means match $\mathbf{x}_1\,\mathbf{x}_2\,\mathbf{x}_3$ from training set

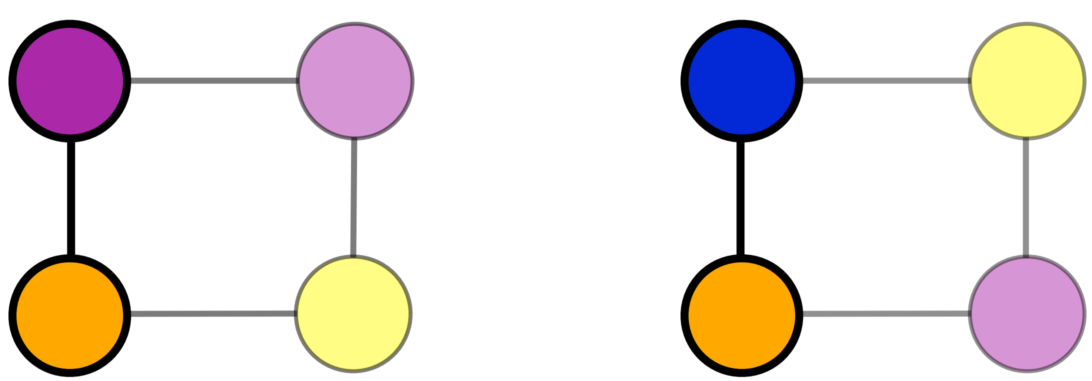
Piecewise Training

A First Approach at Divide-and-Conquer

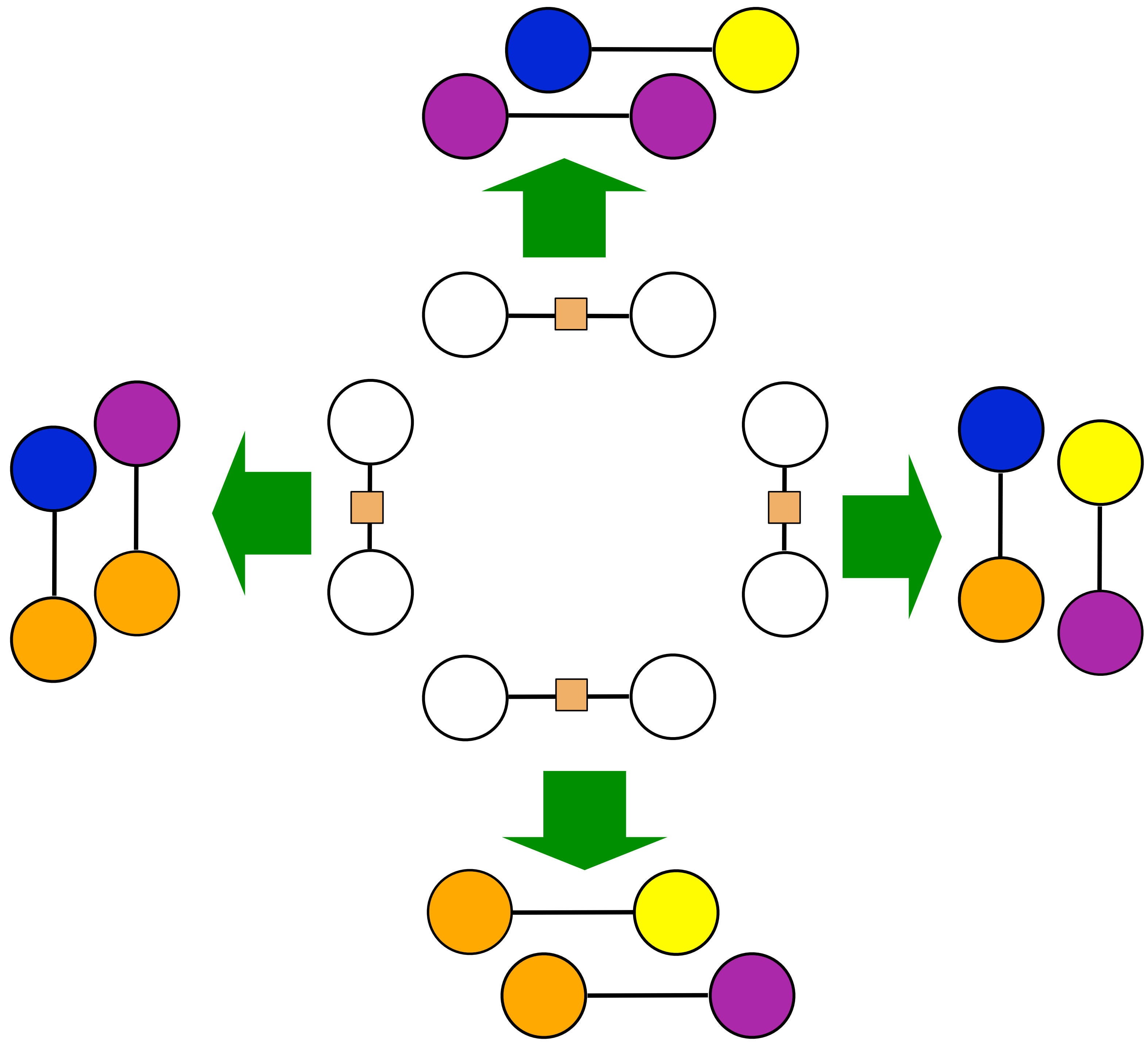


$$p(y_1, y_2, y_3, y_4) = Z^{-1} t(y_1, y_2) t(y_2, y_3) t(y_3, y_4) t(y_4, y_1)$$

TRAINING DATA:

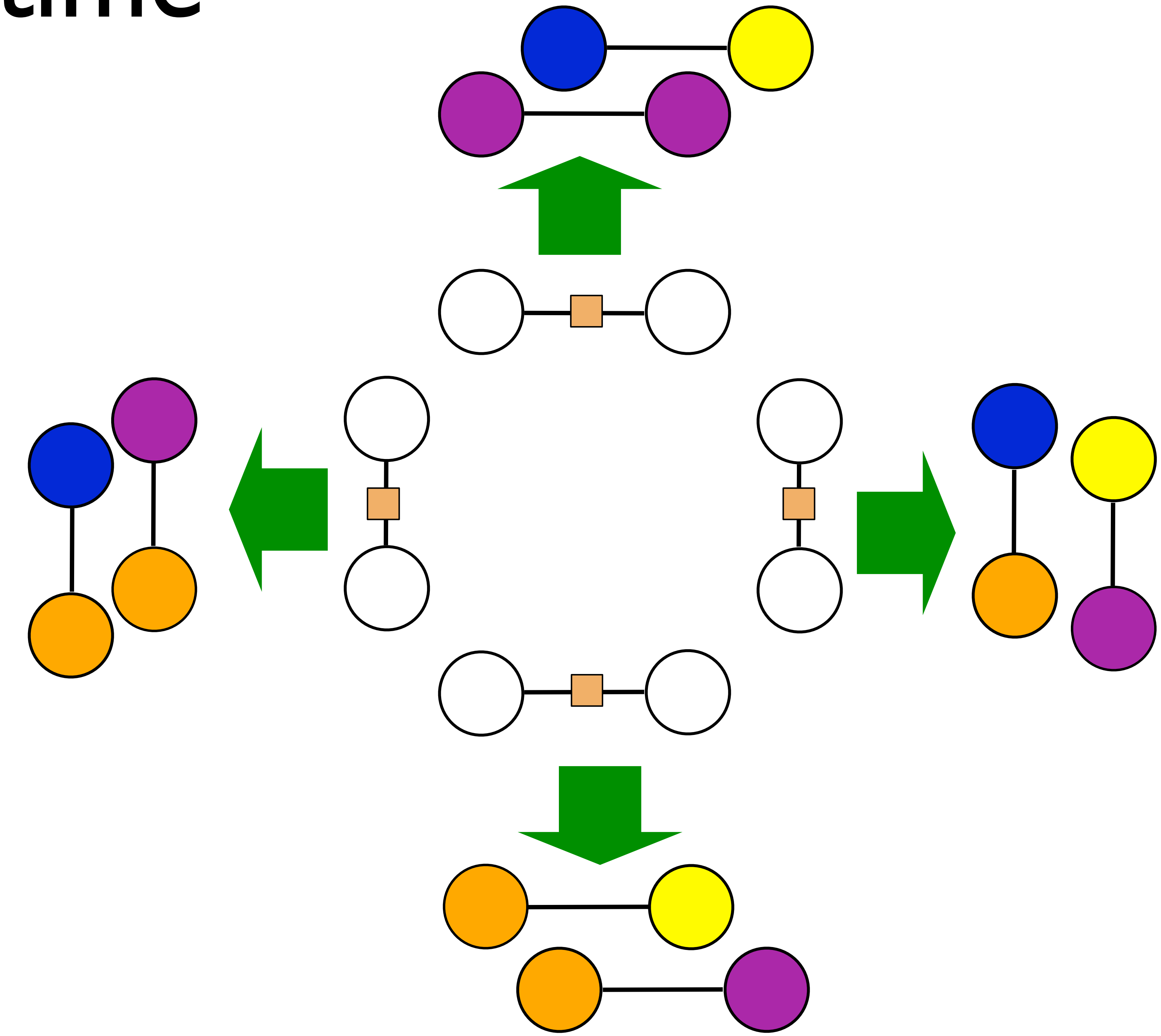


[Sutton and McCallum, 2005]



[Sutton and McCallum, 2005]

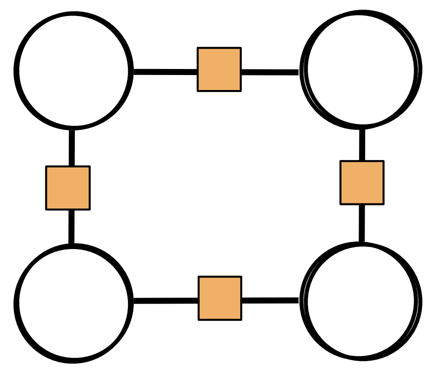
Training time



$$L_{\mathrm{PW}}(\theta) = \prod_a \frac{t_a(y_a)}{\sum_{y'_a} t_a(y'_a)}$$

[Sutton and McCallum, 2005]

Test time



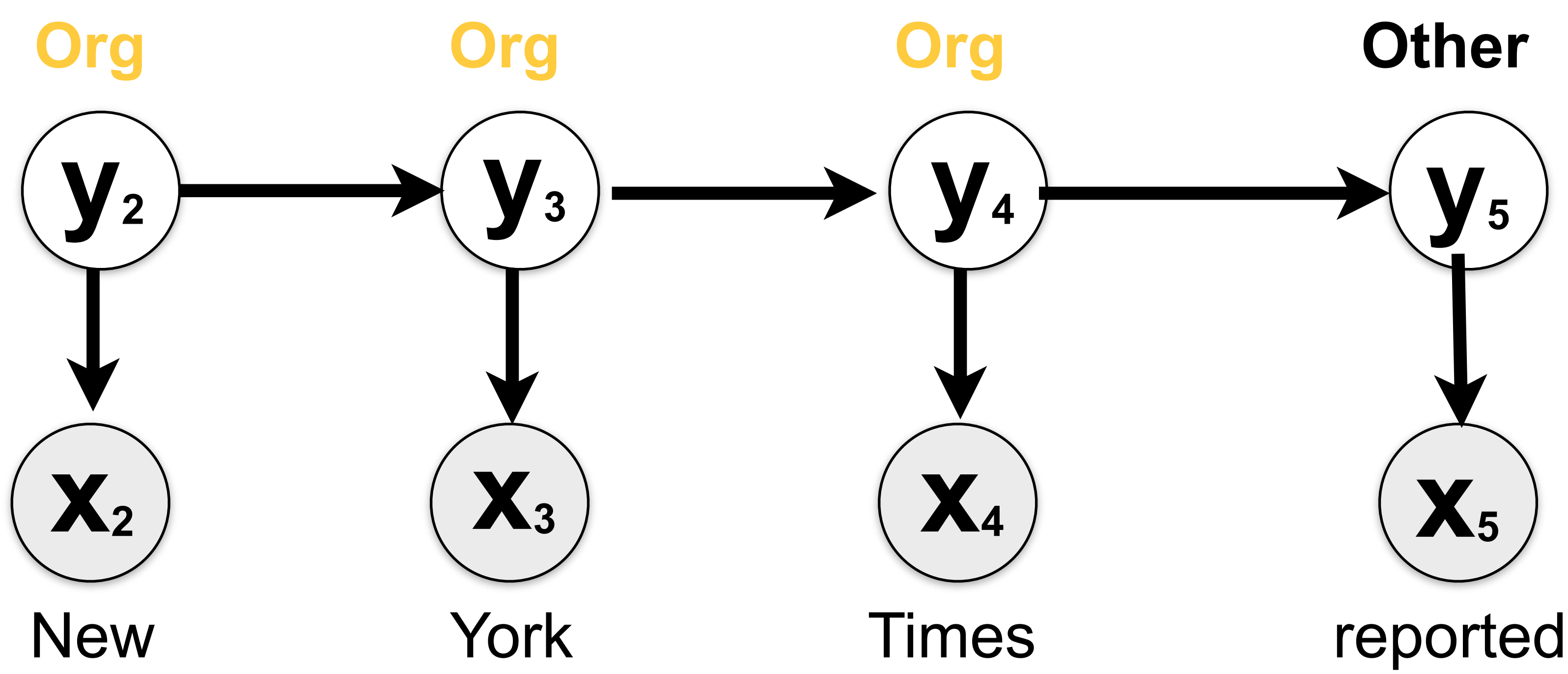
Put model back together, predict via

$$\max_{y_1,y_2,y_3,y_4} p(y_1,y_2,y_3,y_4) = Z^{-1} t(y_1,y_2) t(y_2,y_3) t(y_3,y_4) t(y_4,y_1)$$

joint max over y rather than independent

[Sutton and McCallum, 2005]

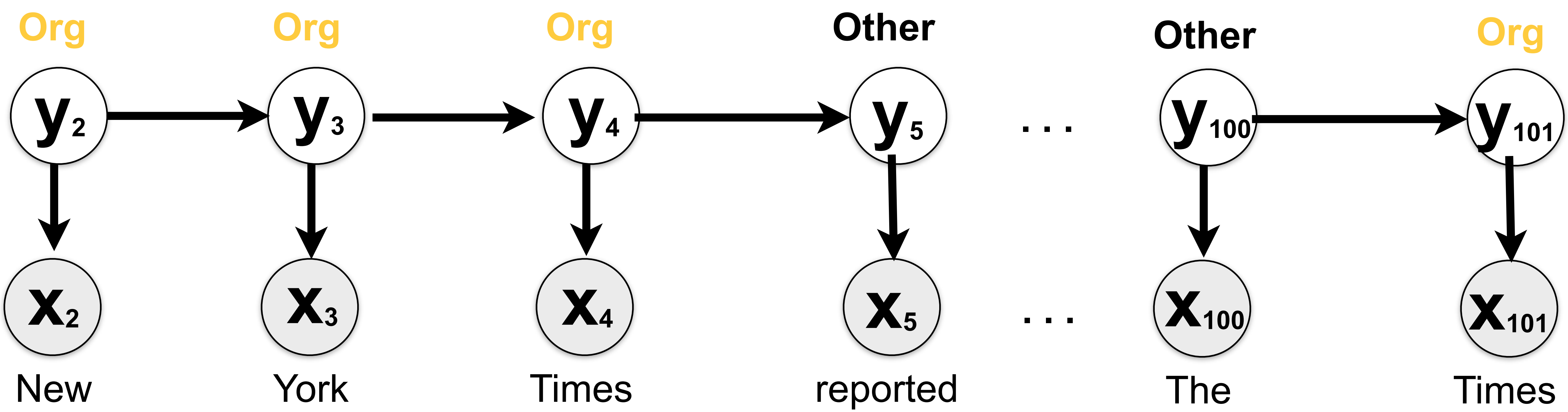
Learning Structure

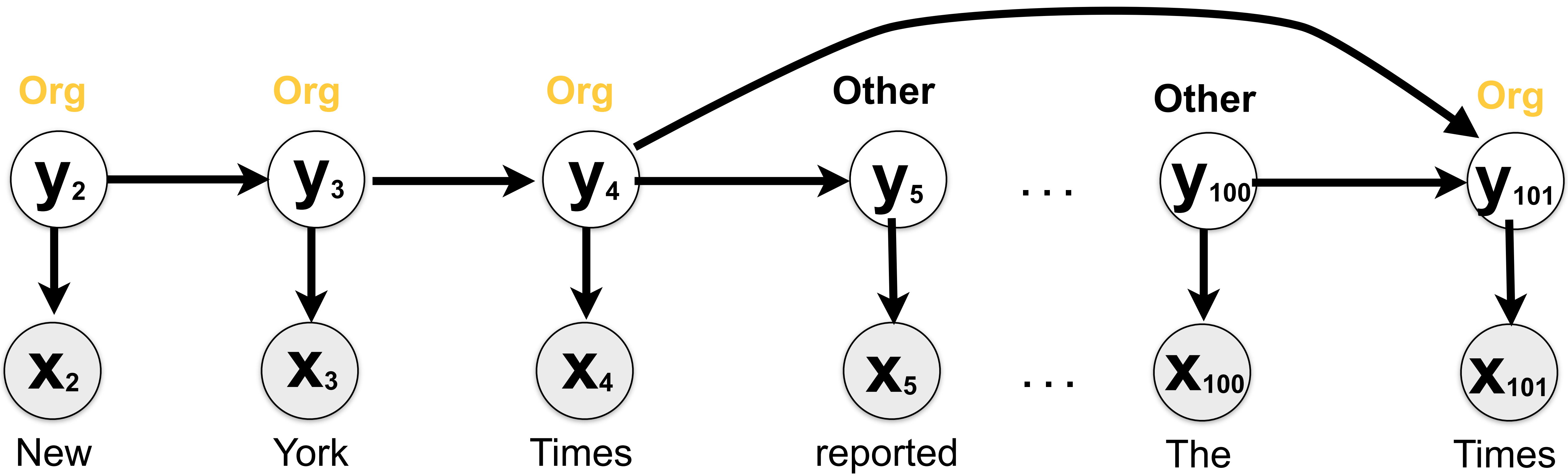


Parameters on each edge: Learned from data

Structure: You have to pick

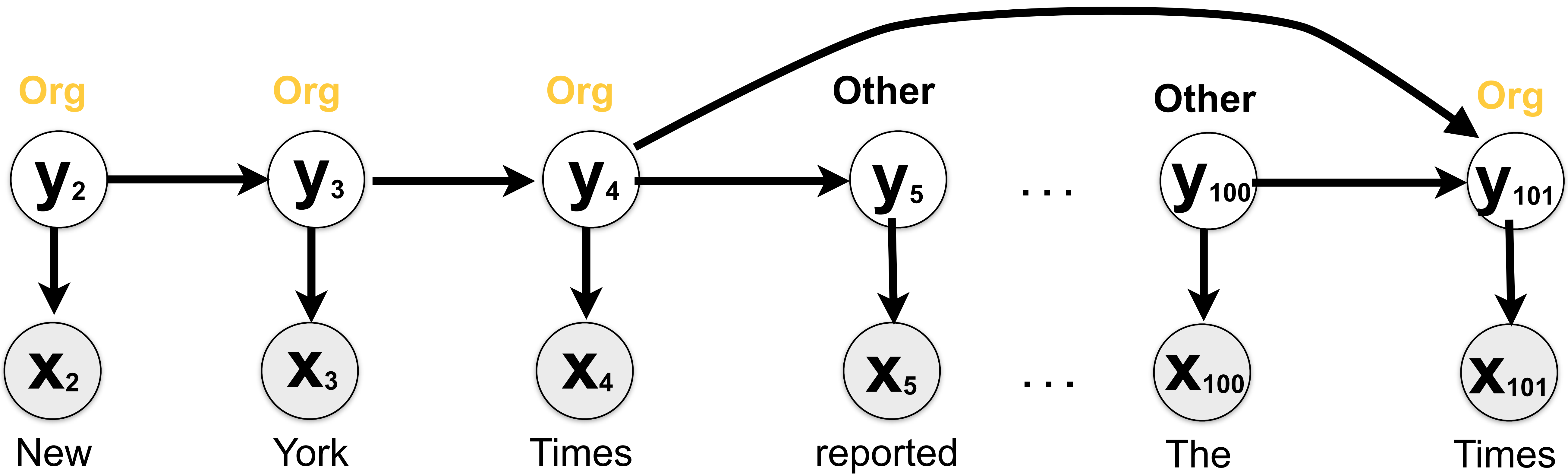
Keeps learned prediction "sane"


Org
Org
Org
Other
Other
Org
y_2
y_3
y_4
y_5
. . .
y_{100}
y_{101}
x_2
x_3
x_4
x_5
. . .
x_{100}
x_{101}
New
York
Times
reported
The
Times



"Skip-chain CRF" *[Sutton and McCallum, 2004; Finkel, Grenager, and Manning, 2005]*

[Rosenberg, Klein, and Taskar, 2007]



"Skip-chain CRF" *[Sutton and McCallum, 2004; Finkel, Grenager, and Manning, 2005]*

[Rosenberg, Klein, and Taskar, 2007]

How to automate this?

Learning Structure

Why hard?

Computational

Need to search exponential number of graphs

Statistical

Some graphs very complex, will overfit

Others won't

Learning Structure

Possible avenues:

- Adding inductive bias to structure learning?
- Sensible way of structure learning with latent variables?

Four Open Areas

All active areas of research:

I. Learning at Scale

II. Exploiting Synergy In Learning

III. Learning Structures

IV. Divide and Conquer

Four Open Areas

All active areas of research:

I. **Learning at Scale** (intense current interest)

II. **Exploiting Synergy In Learning**

III. Learning Structures

IV. Divide and Conquer

Four Open Areas

All active areas of research:

I. Learning at Scale

II. Exploiting Synergy In Learning

III. **Learning Structures** (long history, less now)

IV. Our Insidious Inability to Divide and Conquer

Four Open Areas

All active areas of research:

I. Learning at Scale

II. Exploiting Synergy In Learning

III. Learning Structures

IV. **Divide and Conquer**

(less work here)

Four Open Areas

All active areas of research: (this can be bad)

I. Learning at Scale

II. Exploiting Synergy In Learning

III. Learning Structures

IV. Divide and Conquer

The three outstanding problems in physics, in a certain sense, were never worked on while I was at Bell Labs...

1. time travel,
2. teleportation
3. antigravity

They are not important problems because we do not have an attack. It's not the consequence that makes a problem important, it is that you have a reasonable attack.

— Richard Hamming, *You and Your Research*

ICML 2012

Edinburgh, Scotland
June 26 - July 1, 2012
Paper Deadline: 24 Feb 2012
http://icml.cc/

Four Open Areas

I. Learning at Scale

II. Exploiting Synergy In Learning

III. Learning Structures

IV. Divide and Conquer