Statistical Relational AI Meets Deep Learning

Gautam Kunapuli
Research Associate Professor
Department of Computer Science

UT DALLAS
Erik Jonsson School of Engineering and Computer Science

http://starling.utdallas.edu
UT DALLAS STARLinG LAB
STATISTICAL RELATIONAL LEARNING

Navdeep Kaur, Gautam Kunapuli, Tushar Khot, Kristian Kersting, William Cohen and Sriraam Natarajan (2018). **Relational Restricted Boltzmann Machines: A Probabilistic Logic Learning Approach**. In: Lachiche N., Vrain C. (eds) *Inductive Logic Programming* (ILP'17). Lecture Notes in Computer Science, v. 10759. Springer, Cham

Navdeep Kaur
Indiana University / The University of Texas at Dallas

Tushar Khot
Allen Institute for Artificial Intelligence

Kristian Kersting
Technische Universität Darmstadt

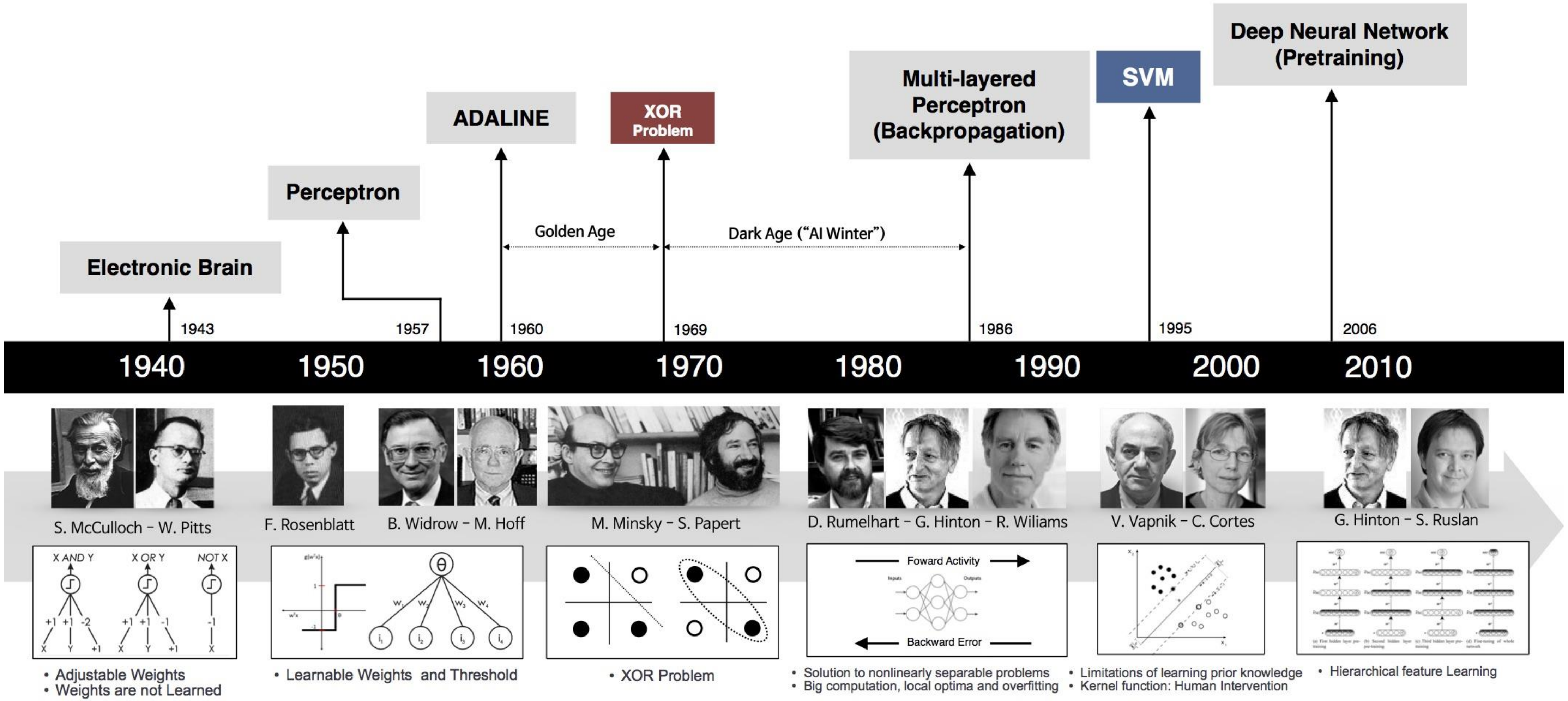
William Cohen
Google

Sriraam Natarajan
The University of Texas at Dallas

Statistical Relational AI meets Deep Learning
The Big Takeaway

- **Neural networks and deep learning** seeing an extraordinary resurgence
  - *widely applied to image, audio and video processing in diverse domains and problems*
- Deep learning inputs are **flat representations**: vectors, matrices, tensors
  - *limits applicability to data with rich relational structure such as graphs and networks*
- **Statistical relational learning** emerging as a powerful framework
  - combines logic (for representing structure) and probability (to capture uncertainty)
  - *widely applied to knowledge bases, social networks, large structured data sets*
- Combine the two frameworks: **augment RBMs with relational features**
  - *qualitative relationships (structure): **relational random walks***
  - *quantitative influences (parameters): **restricted Boltzmann machines***
- **Relational Restricted Boltzmann Machines (R^{2}BM)**
  - *expressive and interpretable deep models*

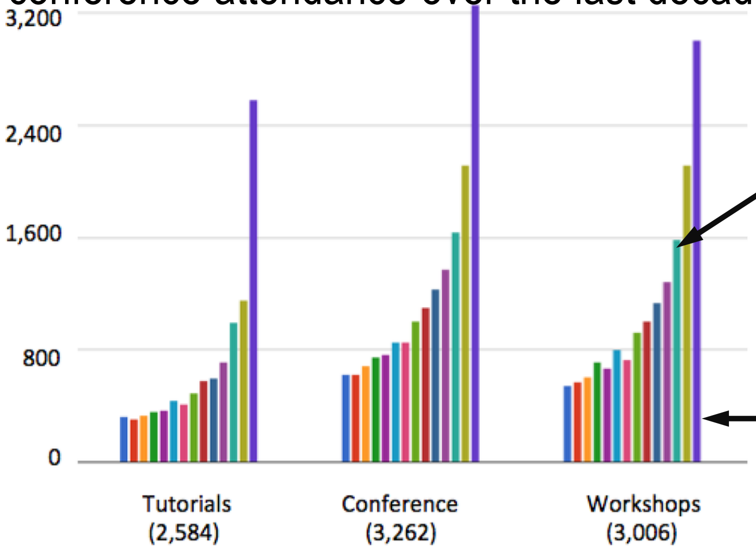
Neural Networks to Deep Learning
Changing Fortunes in the 20th Century



Source: Unknown

The Second Golden Age
Deep Learning in the 21st Century

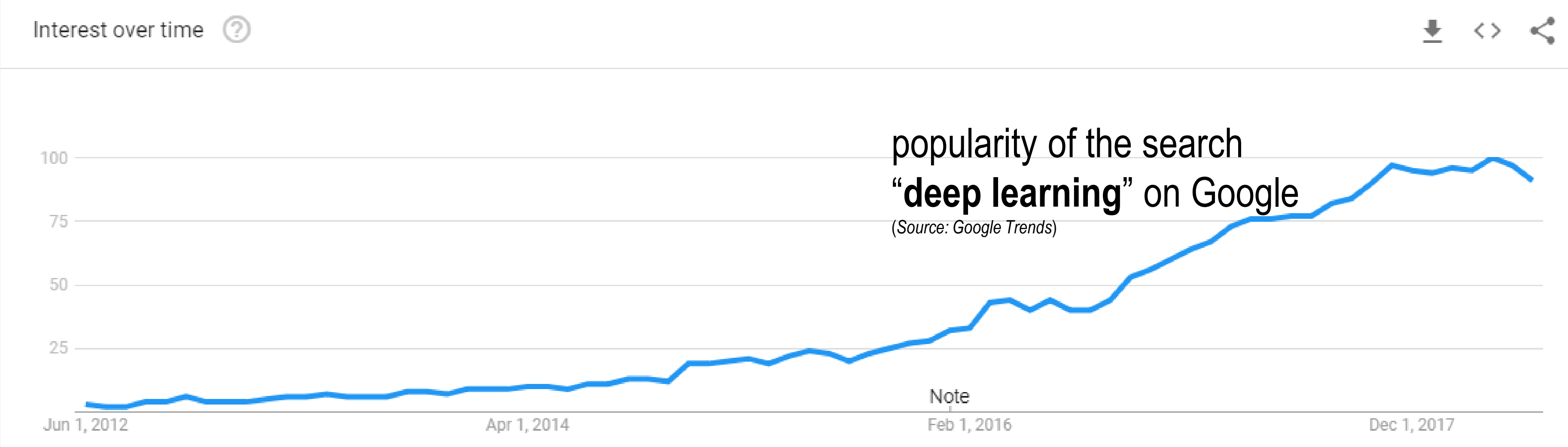
Neural Information Processing Systems (NIPS)
conference attendance over the last decade



Source: Andrew Beam

Mark Zuckerberg attends NIPS 2013 and hires **Yann LeCun** to lead **Facebook AI Research**

This figure only tracks attendance till **2015**; NIPS **2017** drew over **8000** attendees



popularity of the search "**deep learning**" on Google
(Source: Google Trends)

The Second Golden Age
Why Deep Learning Now?

Significant Technological Advances:

- **Availability of massive, powerful computing resources:** *More GPUs means more layers*
- **Availability of massive, high-quality labeled data sets:** *More layers means more labeled data*

Significant Technical Advances:

- **Optimization-friendly activation functions**: *Rather than using neurocognition-inspired activation functions (logistic, hyperbolic tan), use activation function such as RelU to handle* ***vanishing gradients***
- **Robust optimizers**: *Newer variants of* ***stochastic gradient descent*** *(momentum, RMSprop, and ADAM) produce better weights, faster*
- **Improved architectures:** *U-nets, Highway networks, Siamese networks, Resnets enable deep learning for different types of problems and domains*
- **Effective regularization:** *Techniques like batch normalization and data-augmentation reduce* ***overfitting***

Significant Accessibility:

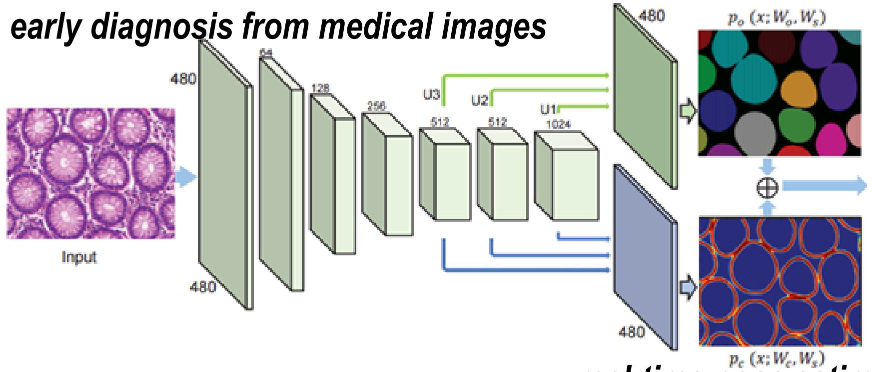
- **Widely-accessible software platforms** *like TensorFlow, Theano, Mxnet, Chainer implement a variety of layers, activation types, and GPU-based optimization algorithms and make prototyping faster*

Adapted from Andrew Beam's blog post: "Deep Learning 101 - Part 1: History and Background"
https://beamandrew.github.io/deeplearning/2017/02/23/deep_learning_101_part1.html

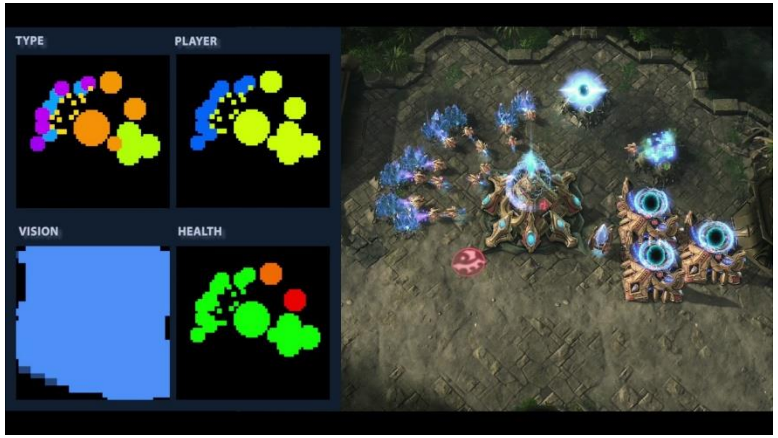
Deep Learning Applications

Deep Learning's greatest successes (arguably) are in **image, audio and video analysis** applications

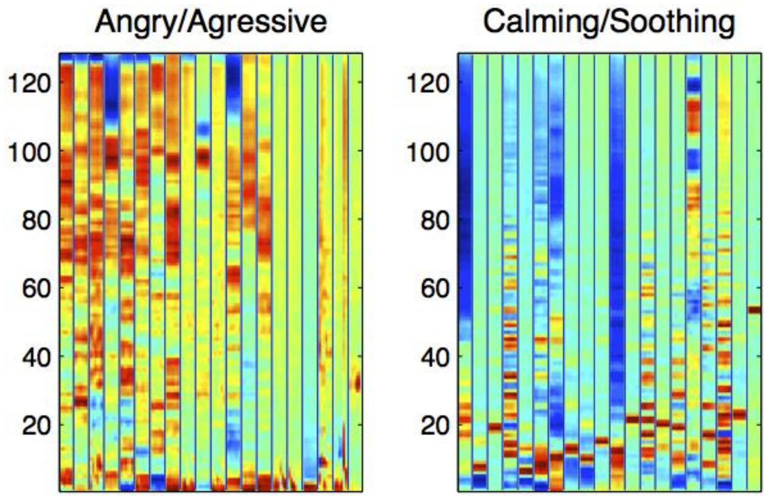
early diagnosis from medical images



autonomous agents for (video) games

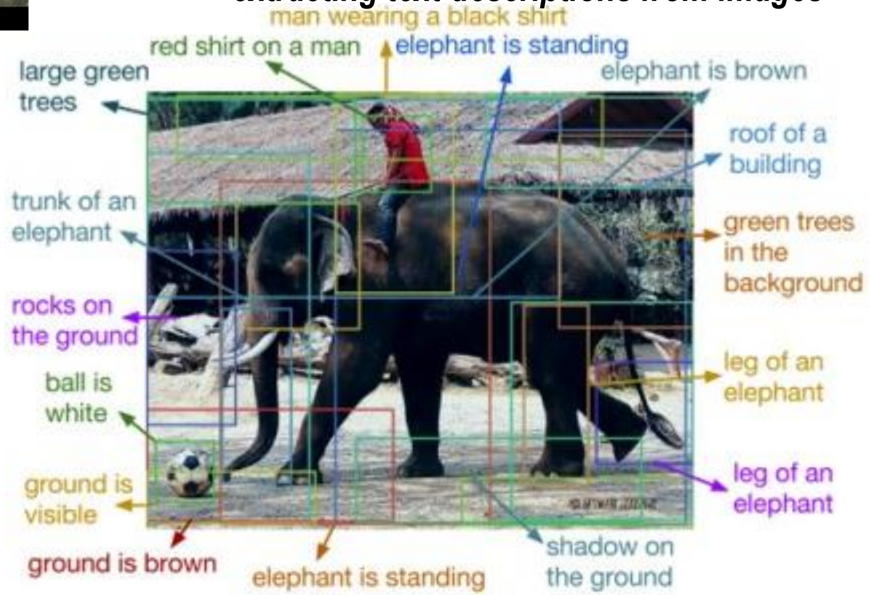


real-time pose estimation



audio analysis of music for genre classification

extracting text descriptions from images



lip reading

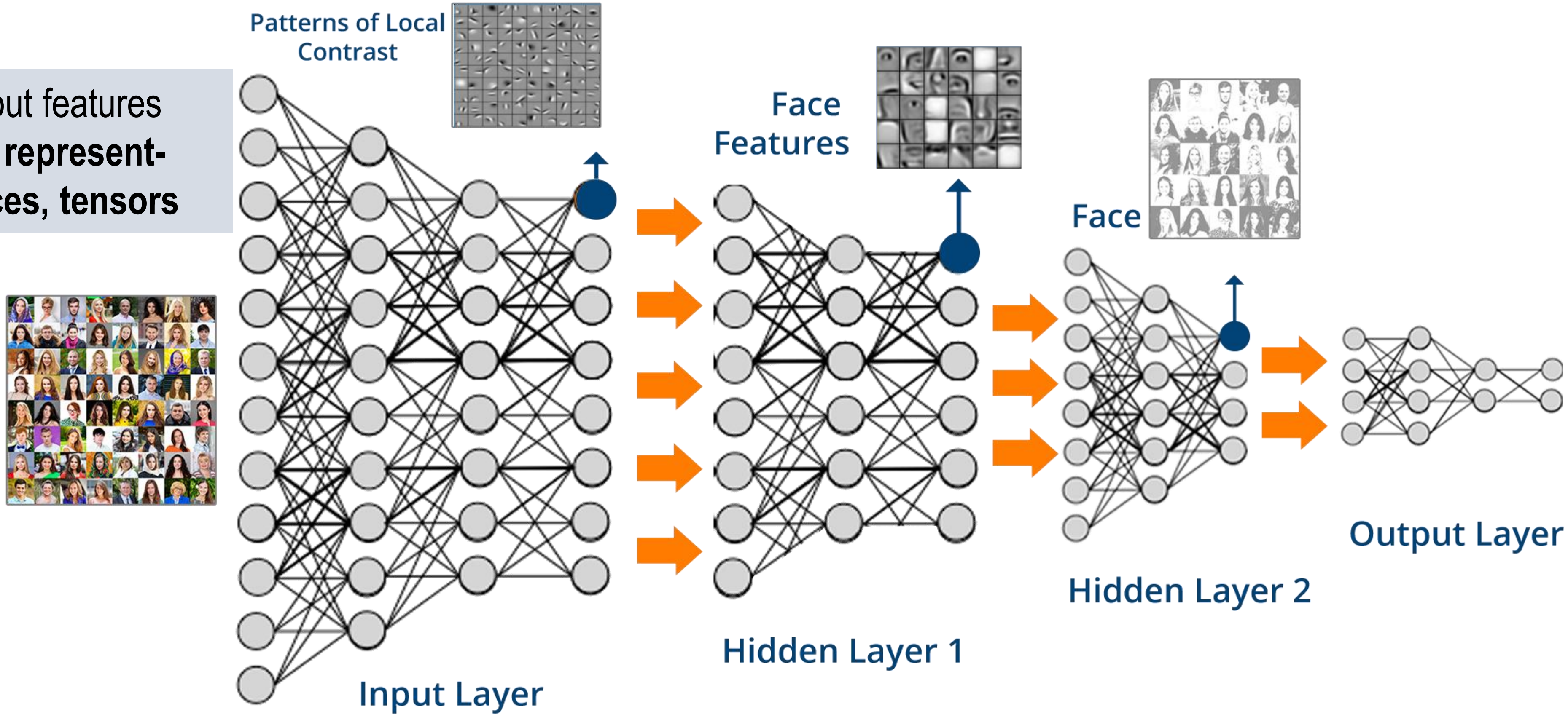
colorizing black and white images

Colorado National Park, 1941 | Textile Mill, June 1937 | Berry Field, June 1909 | Hamilton, 1936

Deep Learning Pros
… and Cons

pro: multiple layers discover and **generate** new **feature combinations**
con: intermediate layers are not always easily **interpretable**, especially by non-machine-learning domain experts

pro: can handle large number of input features
con: inputs are standardized to **flat representations** of features: **vectors, matrices, tensors**



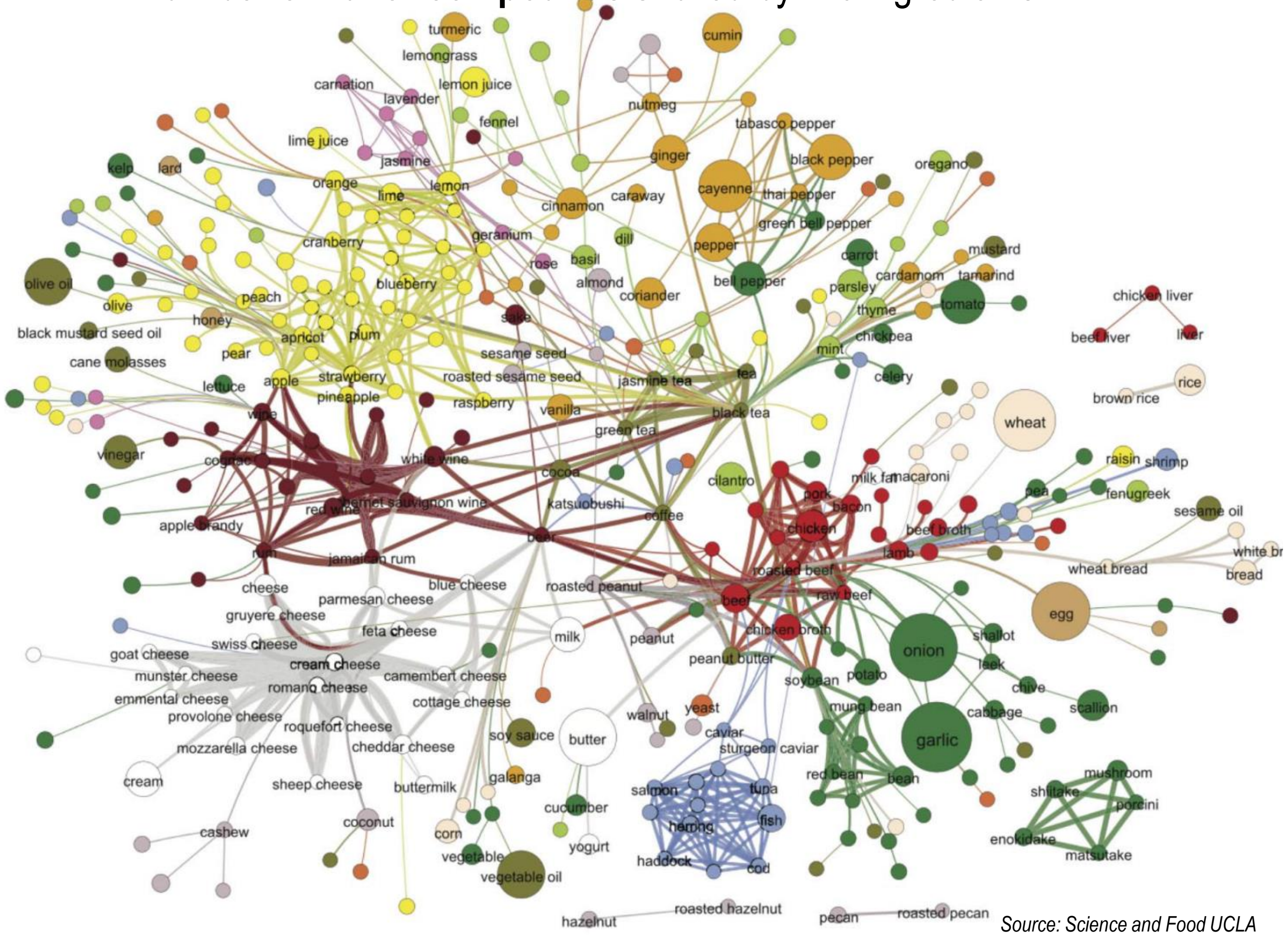
Source: Matthew Mayo, KDNuggets

Domains with Objects, Attributes and Relations

Flat Representations Cannot Handle Structure

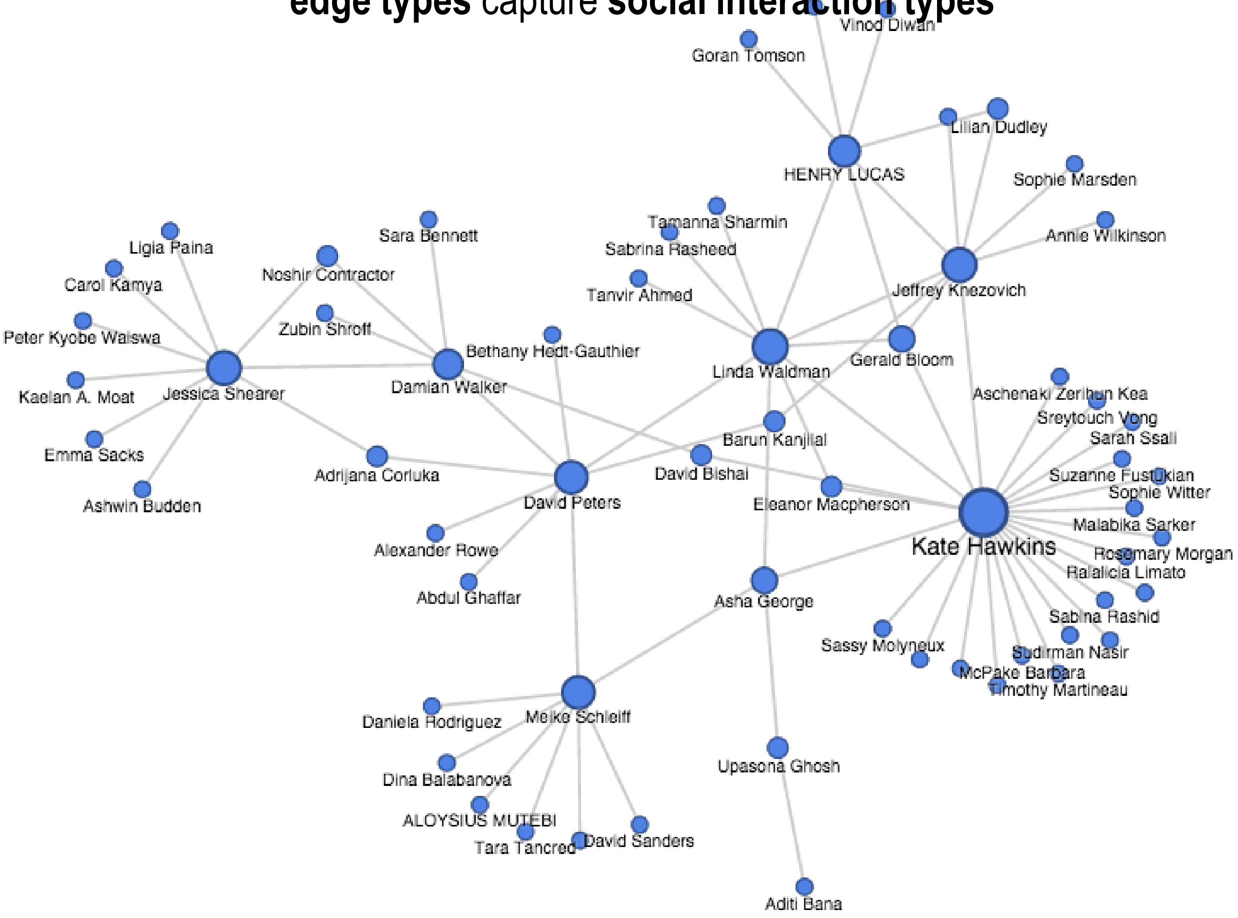
Most **data** is actually **stored in relational databases**, and contains **objects**, their **attributes** and **relationships** between them

Flavor network: **nodes** are **ingredients**, **node size** is the ingredient's **prevalence** in recipes, **edge thickness** is the **number of flavor compounds shared** by two ingredients



Source: Science and Food UCLA

Social network: **nodes** are **individuals**, **node size** is their social influence, **edges** are social connections between individuals, **edge types** capture **social interaction types**

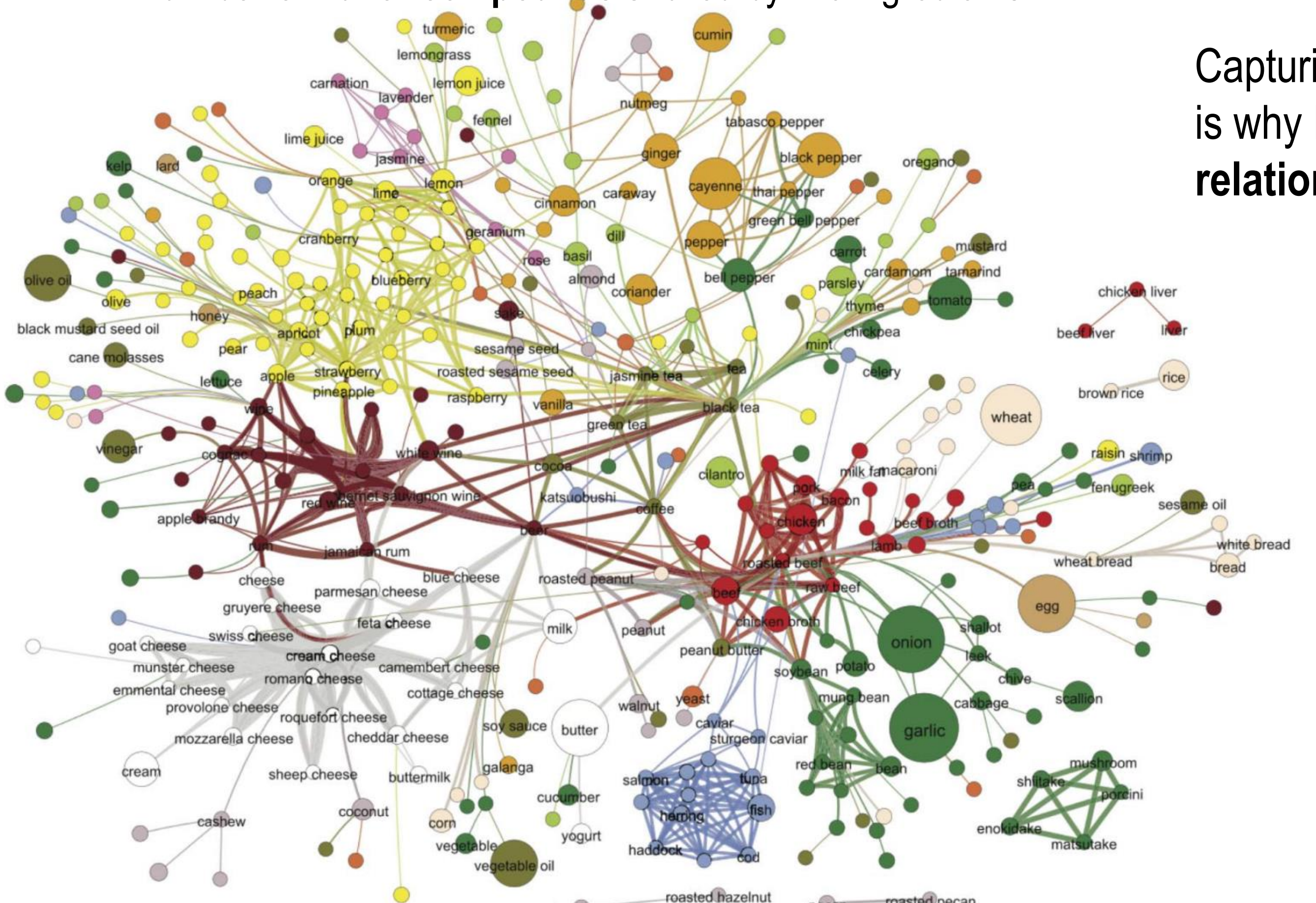


Source: Future Health Systems

Statistical Relational Learning
Flat Representations Cannot Handle Structure

Most **data** is actually **stored in relational databases**, and contains **objects**, their **attributes** and **relationships** between them

Flavor network[1]: **nodes** are **ingredients**, **node size** is the ingredient's **prevalence** in recipes, **edge thickness** is the **number of flavor compounds shared** by two ingredients



Source: Science and Food UCLA

Different **ingredients** may have **different numbers** of flavor ``**neighbors**"
e.g., *cayenne has 6 flavor neighbors, while blueberry has 16*

Capturing this (pairwise) information in a single table is not possible, which is why RDBMS use several tables and a **schema** describing the **relationships** between their columns

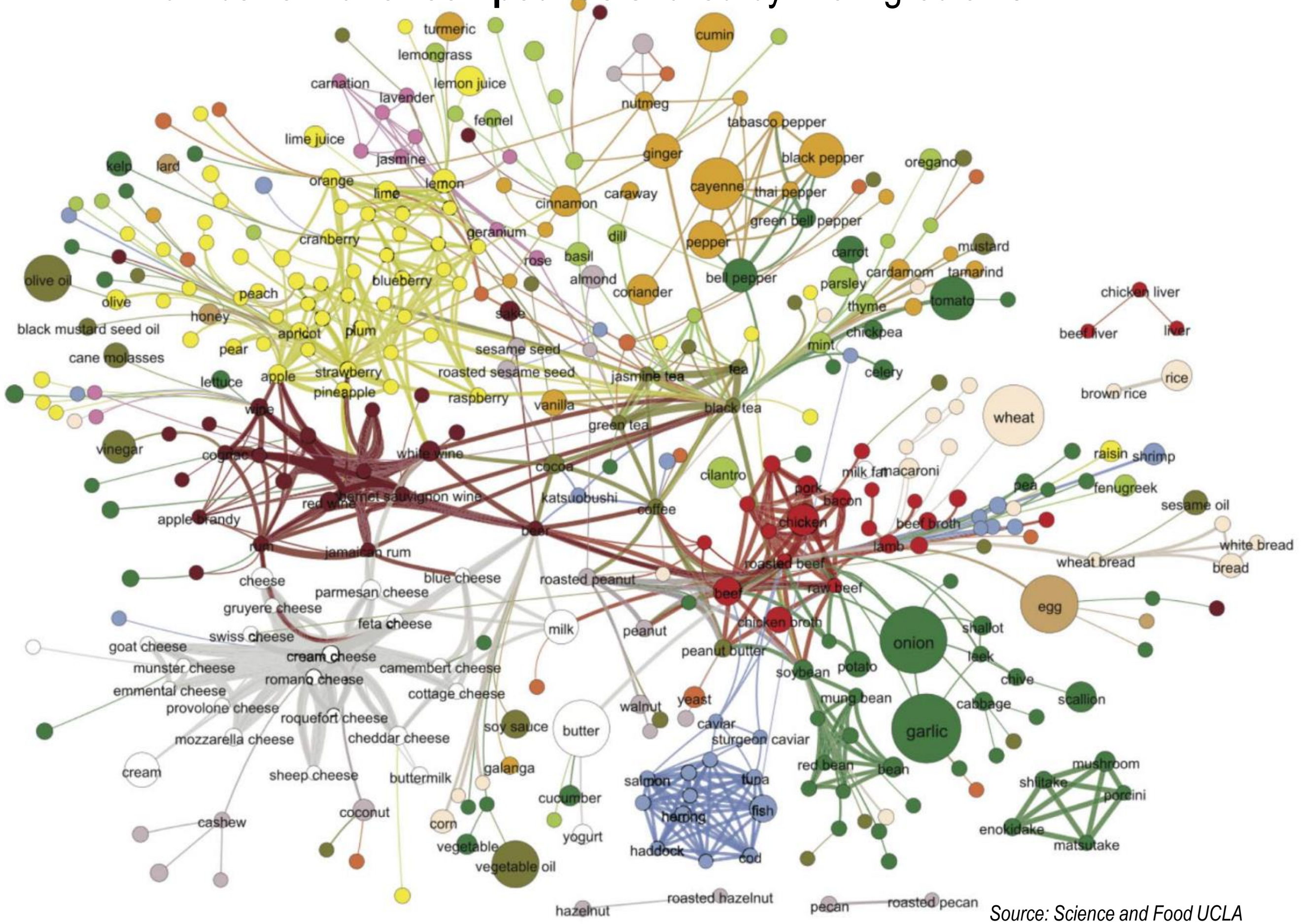
Many other data sets and applications:

- Social Networks, Customer Networks
- Collaborative Filtering
- Electronic Health Record data
- Gene Regulatory Networks
- Bibliographic data
- Communication data
- Trust Networks

[1]Ahn Y-Y, Ahnert SE, Bagrow JP, Barabási A-L (2011). **Flavor network and the principles of food pairing.** *Scientific Reports 1*, 196.

Statistical Relational Learning
First-Order Logic Can Capture Relationships

Flavor network: **nodes** are **ingredients**, **node size** is the ingredient's **prevalence** in recipes, **edge thickness** is the **number of flavor compounds shared** by two ingredients



Source: Science and Food UCLA

Entities, attributes and relationships can be expressed through **logical predicates**

```
IngredientOf(shrimpScampi, shrimp)
IngredientOf(shrimpScampi, garlic)
IngredientOf(shrimpScampi, oliveOil)
IngredientOf(seasonedMussels, garlic)
IngredientOf(seasonedMussels, mussel)
…
FlavorCompound(garlic, hexylAlcohol)
FlavorCompound(mussel, nonanoicAcid)
…
CanSubstitute(shrimp,mussel)
```

Complex interactions can be expressed through **logical clauses** (rules)

```
IngredientOf(?recipe, ?ingredient1) AND
FlavorCompound(?ingredient1, ?compound) AND
FlavorCompound(?ingredient2, ?compound) AND
                    ⇒
CanSubstitute(?ingredient1,?ingredient2)
```

Statistical Relational Learning
But What About Uncertainty?

Learning

Decision trees, Optimization, SVMs, …

Logic

Resolution, WalkSat, Prolog, description logics, …

Probability

Bayesian networks, Markov networks, Gaussian Processes…

Logic + Learning

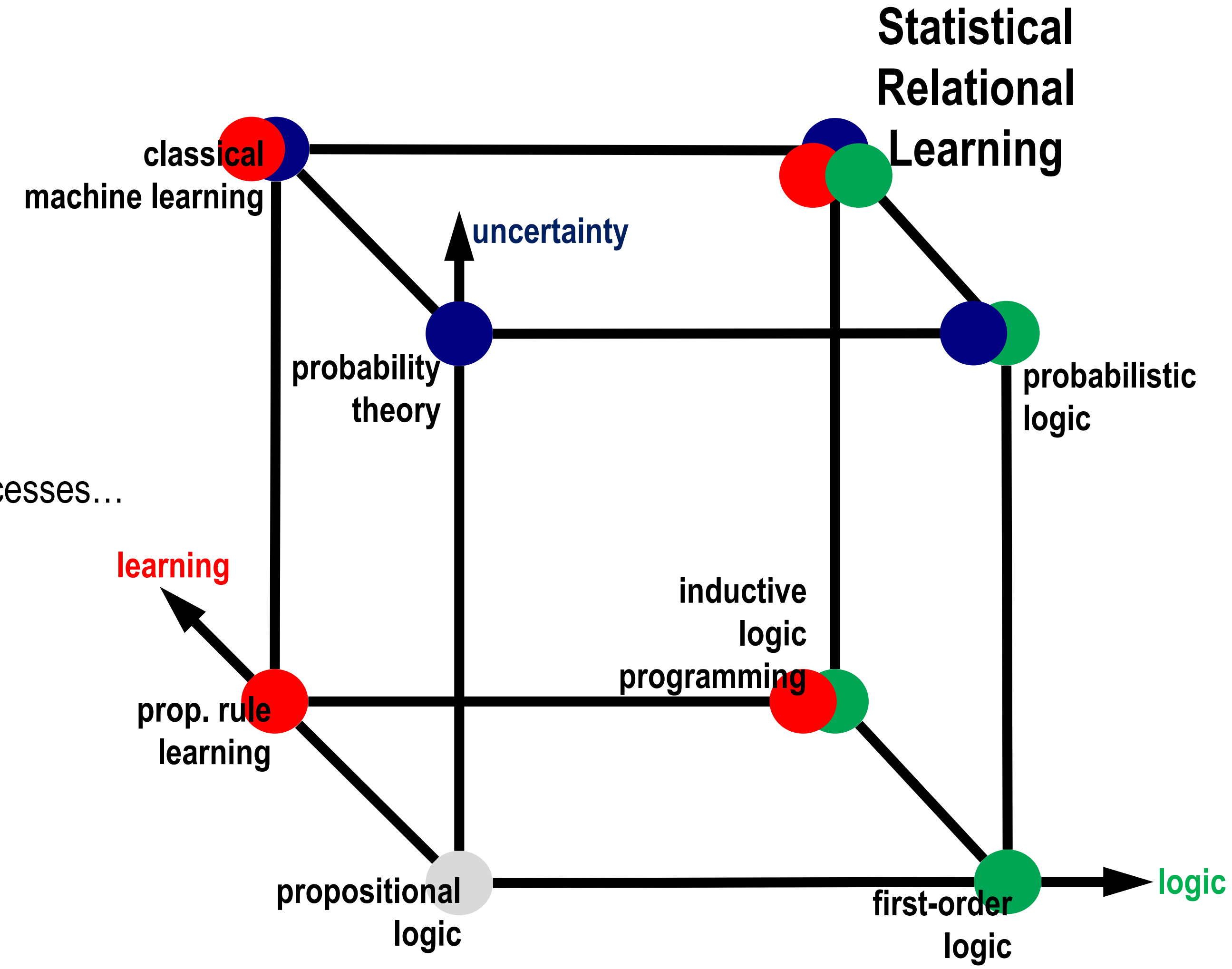
Inductive Logic Programming (ILP)

Learning + Probability

EM, Dynamic Programming, Active Learning, …

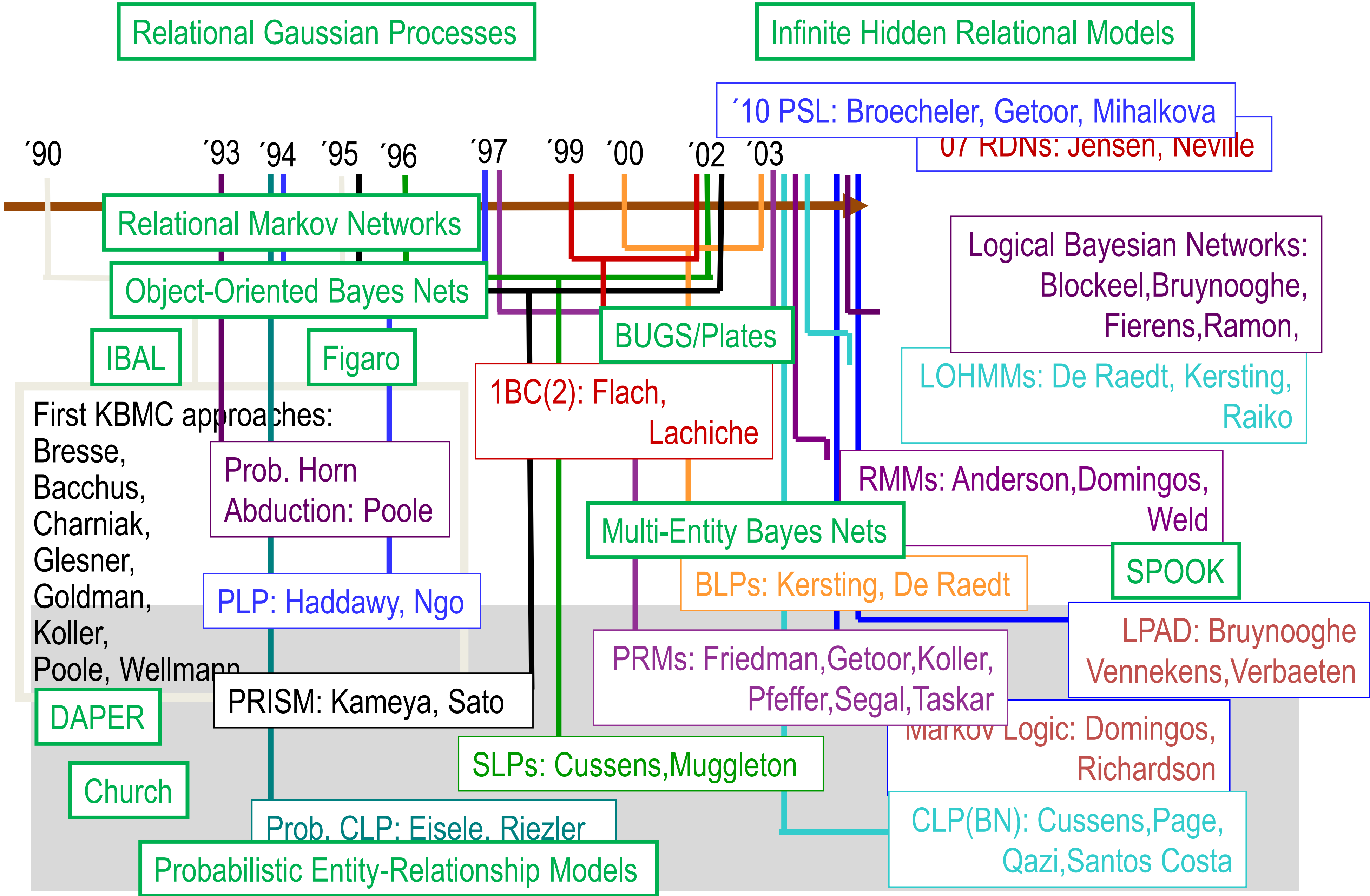
Logic + Probability

Nillson, Halpern, Bacchus, KBMC, ICL, …



Slide adapted from Sriraam Natarajan's tutorial "Probabilistic Logic Models: Past, Present & Future"

Statistical Relational Learning
A Brief History



Slide from Sriraam Natarajan's tutorial "Probabilistic Logic Models: Past, Present & Future"

Statistical Relational Learning
Markov Logic Networks

A Markov Logic Network[2] is specified by a set of **weighted rules** that incorporate domain knowledge **qualitatively** and **quantitatively**:

If two persons are friends, they either both smoke or both do not smoke

```
1.5 Friends(?x, ?y) ⇒ ( Smokes(?x) ⇔ Smokes(?y) )
```

Smoking causes cancer

```
1.2 Smokes(?x) ⇒ Cancer(?x)
```

We will write these as **weighted clauses** (in this example, Horn clauses):

```
1.5 !Friends(?x, ?y) OR !Smokes(?x) OR  Smokes(?y)
1.5 !Friends(?x, ?y) OR  Smokes(?x) OR !Smokes(?y)
1.2 !Smokes(?x) OR Cancer(?x)
```

Weights can be negative and/or infinite, and higher weight ⇒ likelier the constraint is to hold

```
Friends(Amy,Cal)
Friends(Ben,Cal)
Smokes(Amy)
Smokes(Cal)
Cancer(Amy)
```

Evidence is the data known to be true (or false). If we use the **closed-world assumption**, all facts not in evidence are assumed to be false.

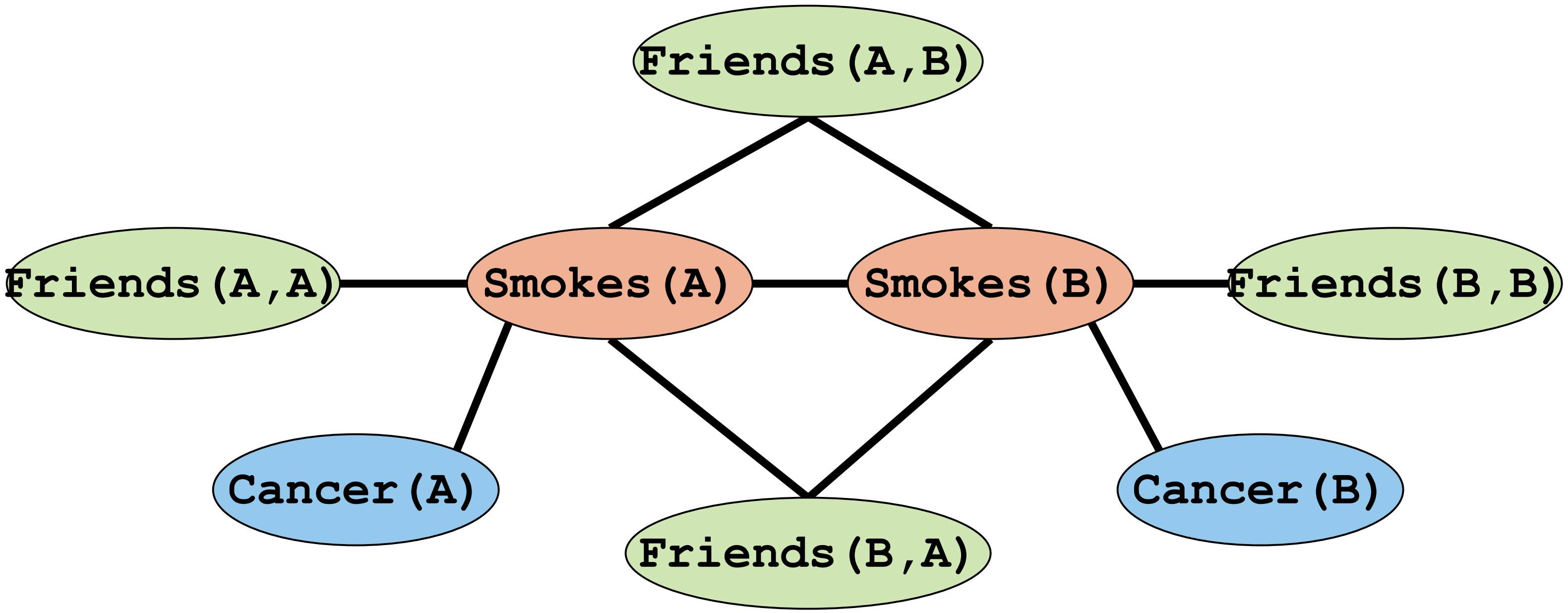
In our example, all facts not in evidence can be **queried**.

[2]M. Richardson and P. Domingos. 2006. **Markov logic networks**. *Machine Learning*, 62(1-2), pp. 107-136.

Statistical Relational Learning
Markov Logic Networks

```
1.5 !Friends(?x, ?y) OR !Smokes(?x) OR  Smokes(?y)
1.5 !Friends(?x, ?y) OR  Smokes(?x) OR !Smokes(?y)
1.2 !Smokes(?x) OR Cancer(?x)
```

Consider this MLN with two people: **Anna (A)** and **Bob (B)**
grounding: instantiating the rules with **all possible values** for the variables
graph structure: edge between two **ground** nodes they **appear together in a rule**
An MLN is **template** for (ground) **Markov networks**



Statistical Relational Learning
Markov Logic Networks

```
1.5  !Friends(?x, ?y) OR !Smokes(?x) OR  Smokes(?y)
1.5  !Friends(?x, ?y) OR  Smokes(?x) OR !Smokes(?y)
1.2  !Smokes(?x) OR Cancer(?x)
```

```
Friends(Amy,Cal)
Friends(Ben,Cal)
Smokes(Amy)
Smokes(Cal)
Cancer(Amy)
```

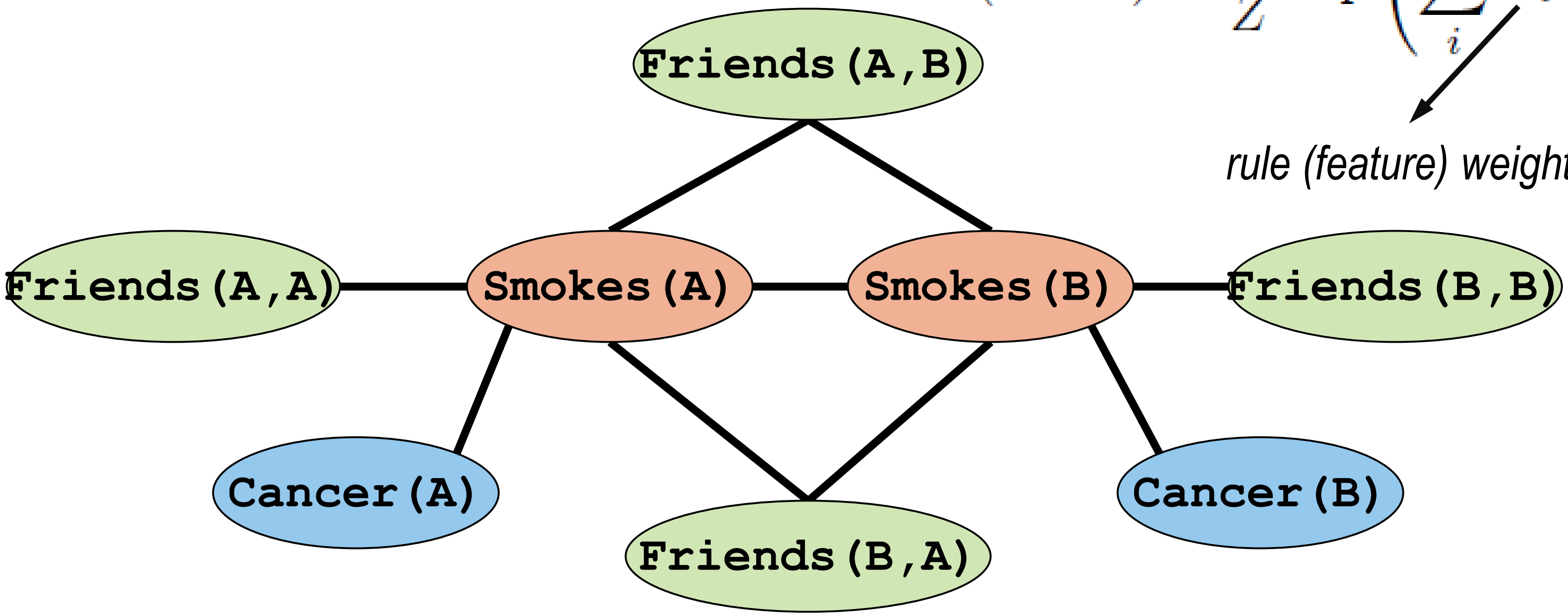
Evidence is the data known to be true (or false). If we use the **closed-world assumption**, all facts not in evidence are assumed to be false.

probability distribution over possible worlds specified by the ground Markov network

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

rule (feature) weights

*#count of the times this rule is satisfied in the world, **x***



Relational Restricted Boltzmann Machines (R^2BM)

SRL Meets Deep Learning

We consider **Restricted Boltzmann Machines (RBMs)**
variant of Boltzmann machines with restriction that neurons form a bipartite graph; restriction allows for more efficient training

Key intuition: Make the RBM features **relational and interpretable**

Construct the distributions similar to an SRL model using **aggregators**

Step 1: Relational Data Transformation

Bring relational data to lifted graphical form

Bring *n*-ary predicates to binary form by introducing Compound Value Type

Step 2: Relational Transformation Layer

Learn *m* Random Walks on Lifted Relational graph connecting argument type of target example

Two ways of transformation

- ***Existential Semantics*** **(RRBM-E)***: if there exists at least one instance of random walk satisfied for target example*
- ***Counts (RRBM-C):*** *# instances of random walk satisfied for target example*

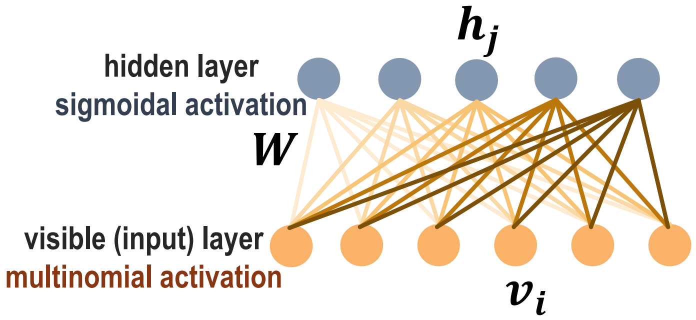
Step 3: Learning Relational RBM

Learn Discriminative RBM by utilizing the features learnt at Transformation layer

(Discriminative) Restricted Boltzmann Machines

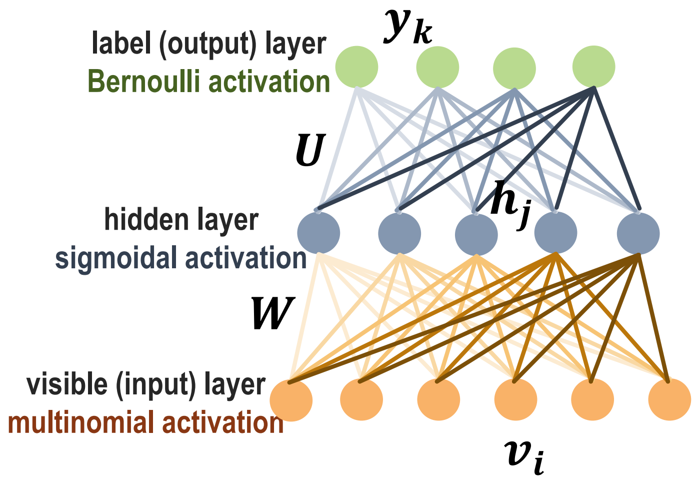
Background and Notation

A **restricted Boltzmann machine** (RBM) is a **generative** stochastic artificial neural network that **can learn a probability distribution** over its set of inputs



$$P(\boldsymbol{v}, \boldsymbol{h}) = \frac{1}{Z} e^{-(\boldsymbol{h}^T W \boldsymbol{v} + \mathbf{b}^T \boldsymbol{v} + \mathbf{c}^T \boldsymbol{h})}$$

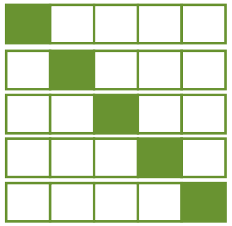
A **discriminative RBM**[3] is a modification that can also **model outputs** for classification problems



$$P(\boldsymbol{v}, \boldsymbol{h}, \boldsymbol{y}) = \frac{1}{Z} e^{-(\boldsymbol{h}^T W \boldsymbol{v} + \mathbf{b}^T \boldsymbol{v} + \mathbf{c}^T \boldsymbol{h} + \boldsymbol{h}^T U \boldsymbol{y} + \mathbf{d}^T \boldsymbol{y})}$$

Multiclass outputs are modeled using **one-hot vectorization**

(Class ID = 1) `person`
(Class ID = 2) `car`
(Class ID = 3) `tree`
(Class ID = 4) `road`
(Class ID = 5) `line`



[3]H. Larochelle and Y. Bengio (2008). **Classification using discriminative restricted Boltzmann machines**. In *Proceedings of the 25th ICML*, pp. 536-543.

Relational Random Walks
Lifted Relational Random Walks

Network architecture is determined by **domain structure**, the set of **relational rules** that describe how various relations, entities and attributes interact

Other approaches employ **carefully hand-crafted rules** or **learn them with inductive logic programming**. We learn structure through **relational random walks**[4]!

Random Walk: *A student* `S` *takes a course* `C` *taught by Professor* `P`

$$\mathtt{S} \xrightarrow{\mathtt{takes}} \mathtt{C} \xrightarrow{\mathtt{taughtBy}} \mathtt{P}$$

Clausal Form: `takes(S,C) AND taughtBy(C,P)`

A **relational random walk** through a domain's **schema** (lifted relational graph) is a chain of relations that identifies a **feature template**

Random Walk: *A student* `S` *is the author of two publications,* $\mathtt{T}_1$ *and* $\mathtt{T}_2$

$$\mathtt{T}_1 \xrightarrow{\mathtt{author}} \mathtt{S} \xrightarrow{\mathtt{author}^{-1}} \mathtt{T}_2$$

Clausal Form: $\mathtt{author}(\mathtt{T}_1,\mathtt{S})$ AND $\mathtt{author}^{-1}(\mathtt{S},\mathtt{T}_2)$

For **semantically sound** relational random walks, we need to define distinct **inverse predicates**, where the argument order (domain and range of binary predicates) is **reversed**

e.g., $\mathtt{author}^{-1}$`(Student, TitleOfPubl)` is the inverse of `author(TitleOfPubl, Student)`

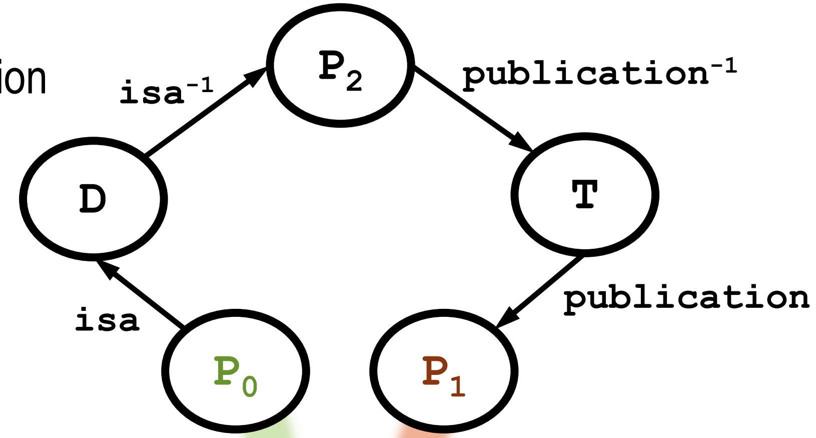
[4]N. Lao, T. Mitchell and W. W. Cohen (2011). **Random walk inference and learning in a large scale knowledge base**. In Proceedings of EMNLP '11. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 529-539.

Relational Random Walks
Lifted Relational Random Walks

Network architecture is determined by **domain structure**, the set of **relational rules** that describe how various relations, entities and attributes interact

Other approaches employ **carefully hand-crafted rules** or **learn them with inductive logic programming**. We learn structure through **relational random walks!**

P: person
D: designation
T: title



Every relational random walk is a **relational feature** that is constrained to **begin at the first argument** and **end at the second argument** of the **target predicate**

relational random walk: a feature template describing what we want to predict

$$\mathtt{advisedBy}(P_0, P_1) \Leftarrow \mathtt{isa}(P_0, D)\ \text{AND}\ \mathtt{isa}(D, P_2)^{-1}\ \text{AND}\ \mathtt{publication}(P_2, T)^{-1}\ \text{AND}\ \mathtt{publication}(T, P_1)$$

target predicate: what we want to predict

Relational Restricted Boltzmann Machines
Step 1: Data Transformation

Convert **predicate logic data** to probabilistic random walk form

Convert **n-ary predicates** to binary form by introducing a **Compound Value Type**

Freebase (a now defunct online knowledge base) used Compound Value Types (CVTs) to represent n-ary relations with n > 2, e.g., values like geographic coordinates, actors playing a character in a movie.

The ternary predicate
`taught(Prof, Course, Semester)`

becomes three binary predicates:
`taught1(t_id, Prof)`,
`taught2(t_id, Course)`,
`taught3(t_id, Semester)`

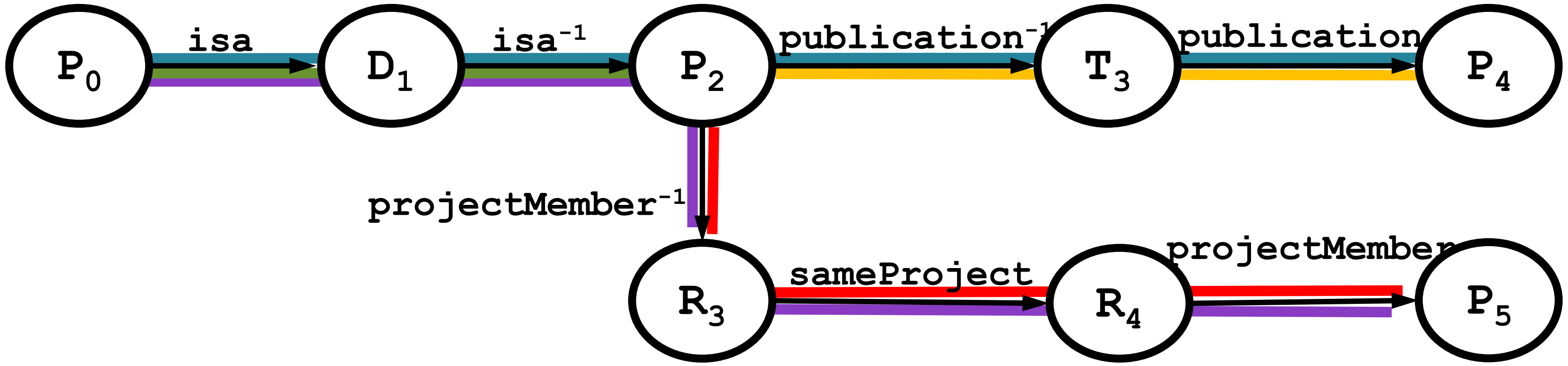
Convert **unary predicates** to binary form by introducing a new predicate **`isa`**

The unary predicate:
`student(Person)`

becomes a binary predicate:
`isa(Person,`student’)`

Relational Restricted Boltzmann Machines
Step 2a: Construct Relational Random Walks

Learn *m* **relational random walks** on the lifted relational graph connecting argument types of target example; each relational random walk represents **local structure** in the domain, or alternately, a **compound feature**



P: person
D: designation
T: title
R: project

RW1: advisedBy(P_0,P_2)$\Leftarrow$ isa(P_0,D_1) $\wedge$ isa^{-1}(D_1,P_2)

RW2: advisedBy(P_0,P_4)$\Leftarrow$ isa(P_0,D_1) $\wedge$ isa(D_1,P_2)$^{-1}$ $\wedge$ publication(P_2,T_3)$^{-1}$ $\wedge$ publication(T_3,P_4)

RW3: advisedBy(P_2,P_4)$\Leftarrow$ publication^{-1}(P_2,T_3) $\wedge$ publication(T_3,P_4)

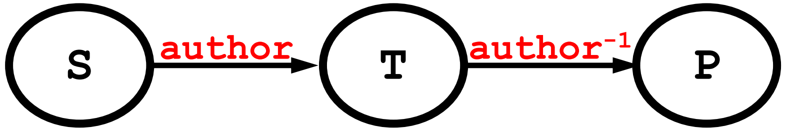
RW4: advisedBy(P_2,P_5)$\Leftarrow$ projectMember^{-1}(P_2,R_3) $\wedge$ sameProject(R_3,R_4) $\wedge$ projectMember(R_4,P_5)

RW5: advisedBy(P_0,P_5)$\Leftarrow$ isa(P_0,D_1) $\wedge$ isa^{-1}(D_1,P_2) $\wedge$ projectMember^{-1}(P_2,R_3) $\wedge$ SameProject(R_3,R_4) $\wedge$ projectMember(R_4, P_5)

Relational Restricted Boltzmann Machines
Step 2b: Create Aggregated Input Feature Vector

Convert each **relational example** into an **aggregate vector** of **random-walk-based features**

RW$_4$: *A student* **S** *and a Professor* **P** *write a paper titled* **T**



advisedBy(S,P) ⇐ **author(S,T)** AND **author^{-1}(T,P)**

not all **Professor-Student** training examples will have the same number of papers
(commonly referred to as **multiple-parent problem**)

Ana-Bob have 10 papers, while **Cal-Dan** have 3.

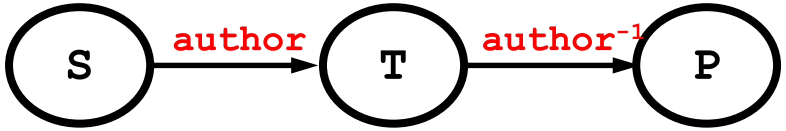
RRBM-E
aggregate using **existential semantics**: does there exist **at least** **<u>one</u>** instance of the random walk satisfied in a given training example?

RRBM-C
aggregate using **count semantics**: **how many instances** of the random walk are satisfied for by a given training example?

Relational Restricted Boltzmann Machines
Step 2b: Create Aggregated Input Feature Vector

Convert each **relational example** into an **aggregate vector** of **random-walk-based features**

RW_4: *A student* **S** *and a Professor* **P** *write a paper titled* **T**



advisedBy(S,P) ⇐ **author(S,T)** AND **author^{-1}(T,P)**

not all **Professor-Student** training examples will have the same number of papers
(commonly referred to as **multiple-parent problem**)

Ana-Bob have 10 papers, while **Cal-Dan** have 3.

RRBM-E

aggregate using **existential semantics**: does there exist **at least** **<u>one</u>** instance of the random walk satisfied in a given training example?

examples \ *features*	RW_1	RW_2	RW_3	RW_4	….	RW_m
advisedBy(Ana,Bob)	1	0	1	1	…	1
advisedBy(Cal,Dan)	1	1	0	1	…	1
advisedBy(Ena,Fen)	1	0	0	0	…	1

RRBM-C

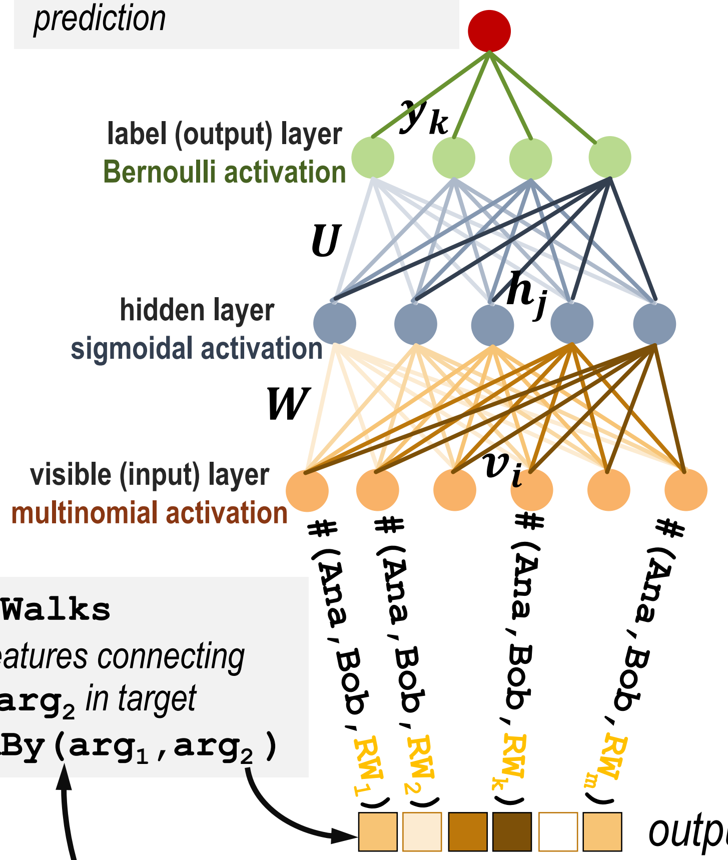
aggregate using **count semantics**: **how many instances** of the random walk are satisfied for by a given training example?

examples \ *features*	RW_1	RW_2	RW_3	RW_4	….	RW_m
advisedBy(Ana,Bob)	0	7	0	10	…	2
advisedBy(Cal,Dan)	3	17	4	3	…	13
advisedBy(Ena,Fen)	0	9	6	0	…	11

Relational Restricted Boltzmann Machines
Step 3: Discriminative Learning

Learn Discriminative RBM by utilizing the aggregated features from the **relational transformation layer**

advisedBy(Ana,Bob)
prediction



relational transformation layer ***stacked*** *on top of the DRBM forms the Relational RBM model*

stochastic gradient descent *is used to learn a regularized, non-linear, weighted combination of features; due to* ***non-linearity****, we can to learn a much more* ***expressive model***

$$p(\hat{y}|x) = \frac{e^{d_{\hat{y}}+\sum_{j=1}^{n}\sigma\left(c_j+U_{j\hat{y}}+\sum_{f=1}^{m}W_{jy}x_f\right)}}{\sum_{k=1}^{C} e^{d_k+\sum_{j=1}^{n}\sigma\left(c_j+U_{jk}+\sum_{f=1}^{m}W_{jf}v_f\right)}}$$

$$\sigma(z) = \log(1+e^z)$$

Random Walks
relational features connecting **arg$_1$** *and* **arg$_2$** *in target* **advisedBy(arg$_1$,arg$_2$)**

output of ***relational transformation layer*** *is fed into* ***multi-layered discriminative RBM***

?advisedBy(arg$_1$=Ana,arg$_2$=Bob)
relational training example with facts (ground instances) about **arg$_1$=Ana** *and* **arg$_2$=Bob**

Relational Restricted Boltzmann Machines
Experimental Setup

Domains:

Domain	Target Predicate
UW-CSE	`advisedBy(Person,Person)`
Cora Entity Resolution	`sameVenue(Venue,Venue)`
IMDB	`workedUnder(Person,Person)`
Yeast	`cites(Paper,Paper)`

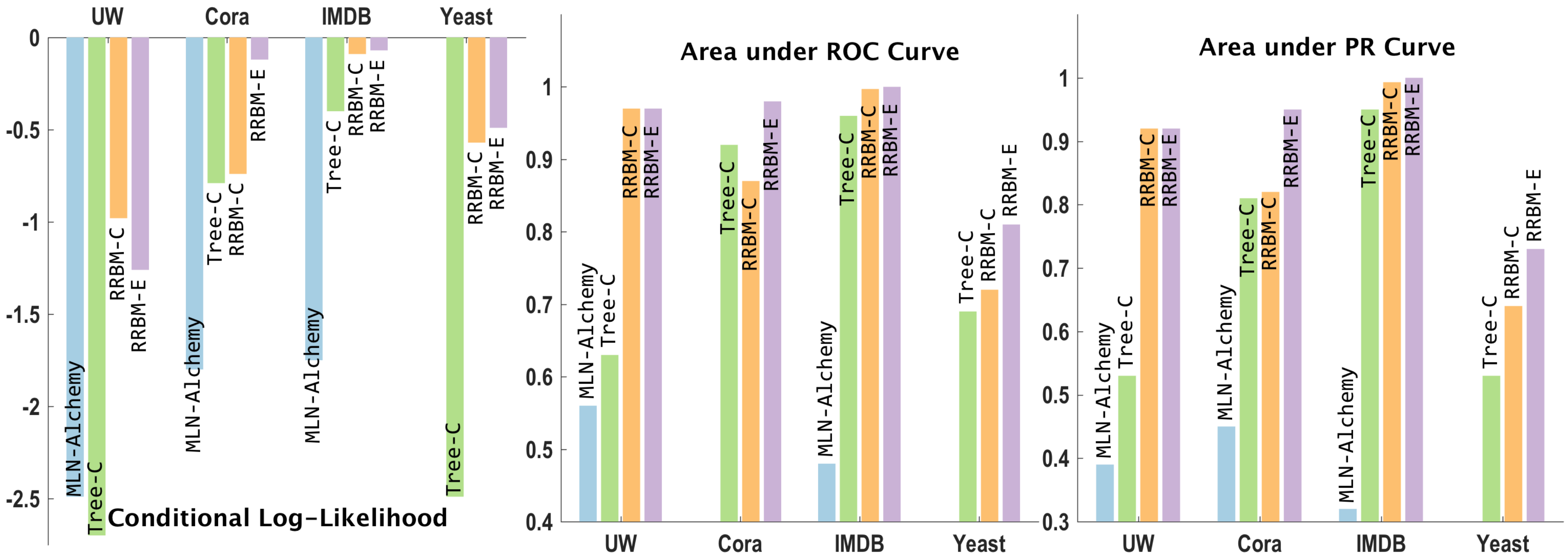
Comparative Algorithms:

- Baselines: *Tree-Count,* MLN (Alchemy[5])
- State-of-the-art SRL Methods[6]: RDN-Boost[7], MLN-Boost[8]

[5] https://alchemy.cs.washington.edu/
[6] https://starling.utdallas.edu/software/boostsrl/
[7] S. Natarajan, T. Khot, K. Kersting, B. Gutmann and J. W. Shavlik (2012). **Gradient-based Boosting for Statistical Relational Learning: The Relational Dependency Network Case**, *Special issue of Machine Learning Journal (MLJ)*, Volume 86, Number 1, pp. 25-56.
[8] T. Khot, S. Natarajan, K. Kersting, B. Gutmann and J. W. Shavlik (2015). **Gradient-based Boosting for Statistical Relational Learning: The Markov Logic Network and Missing Data Cases**, *Machine Learning Journal*, Volume 100, Issue 1, pp. 75-100.

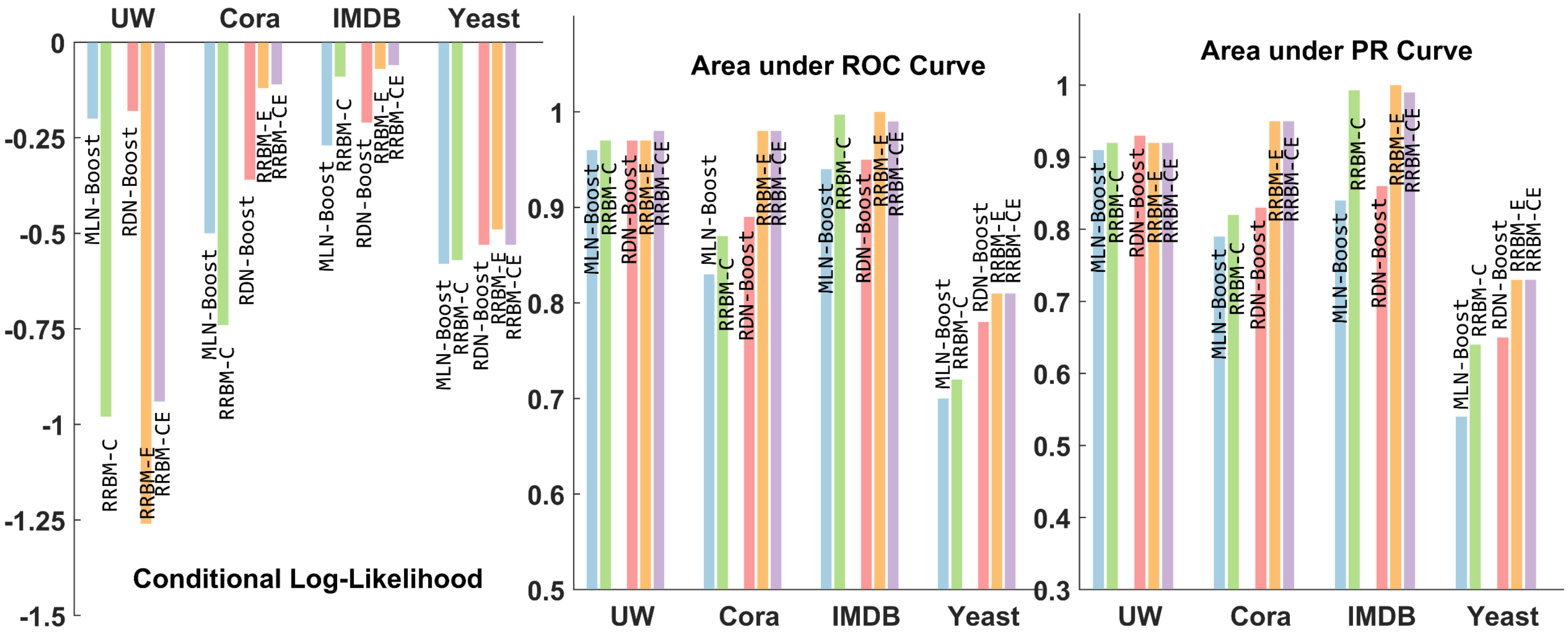
Relational Restricted Boltzmann Machines

RRBM Outperforms Baseline MLN and Decision-Tree Models



Relational Restricted Boltzmann Machines

RRBM Performs Similar To/BetterState-of-The-Art SRL Models



Relational Restricted Boltzmann Machines
Discussion

- Method to **augment RBMs with relational features**
- Connections to **existing SRL approaches**
- On par with **state-of-the-art SRL results**
- **Future work**
  - Multiple distributions
  - Predicate invention using RWs and RBMs
  - More interesting deep models
  - Exploring closing of loop – using deep features to improve log-linear model

Current and Future Work
Lifted Relational Neural Networks