Probabilistic reasoning about plants, animals, objects, and people

David Poole

Department of Computer Science,
University of British Columbia

Work with: David Buchman, Bahare Fatemi, Seyed Mehran Kazemi, Kristian Kersting, Sriraam Natarajan, Perouz Taslakian

February 11, 2021

"The mind is a neural computer, fitted by natural selection with combinatorial algorithms for causal and probabilistic reasoning about plants, animals, objects, and people.

. . .

"In a universe with any regularities at all, decisions informed about the past are better than decisions made at random. That has always been true, and we would expect organisms, especially informavores such as humans, to have evolved acute intuitions about probability. The founders of probability, like the founders of logic, assumed they were just formalizing common sense."

Steven Pinker, How the Mind Works, 1997, pp. 524, 343.

"The mind is a neural computer, fitted by natural selection with combinatorial algorithms for causal and probabilistic reasoning about plants, animals, objects, and people.

. . .

"In a universe with any regularities at all, decisions informed about the past are better than decisions made at random. That has always been true, and we would expect organisms, especially informavores such as humans, to have evolved acute intuitions about probability. The founders of probability, like the founders of logic, assumed they were just formalizing common sense."

Steven Pinker, How the Mind Works, 1997, pp. 524, 343.

"The mind is a neural computer, fitted by natural selection with combinatorial algorithms for causal and probabilistic reasoning about plants, animals, objects, and people. It is driven by goal states that served biological fitness in ancestral environments, such as food, sex, safety, parenthood, friendship, status and knowledge.

. . .

"In a universe with any regularities at all, decisions informed about the past are better than decisions made at random. That has always been true, and we would expect organisms, especially informavores such as humans, to have evolved acute intuitions about probability. The founders of probability, like the founders of logic, assumed they were just formalizing common sense."

Steven Pinker, How the Mind Works, 1997, pp. 524, 343.

Outline

1. What are relational probabilistic models and relational learning?
   - Relational Models
   - Knowledge Graphs
2. Learning Knowledge Graphs
3. Learning General Knowledge: Lifted Graphical Models
4. Bayesian $\Rightarrow$ Exchangeability $\Rightarrow$ Lifted Inference
5. Identity and Existence Uncertainty

Motivation

- AI studies what agents should do.
  - Acting is gambling: agents who don't use probabilities will lose to those who do.
  - No prediction is certain: never believe anyone who gives definitive predictions!

Motivation

- AI studies what agents should do.
    - Acting is gambling: agents who don't use probabilities will lose to those who do.
    - No prediction is certain: never believe anyone who gives definitive predictions!
- What is the world made up of?
    - ML: Features or random variables

Motivation

- AI studies what agents should do.
    - Acting is gambling: agents who don't use probabilities will lose to those who do.
    - No prediction is certain: never believe anyone who gives definitive predictions!
- What is the world made up of?
    - ML: Features or random variables
    - Everyone else: things (entities, individuals)

Motivation

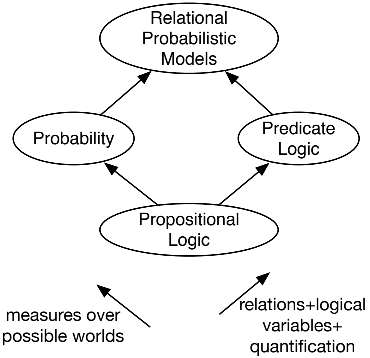
- AI studies what agents should do.
  - Acting is gambling: agents who don't use probabilities will lose to those who do.
  - No prediction is certain: never believe anyone who gives definitive predictions!
- What is the world made up of?
  - ML: Features or random variables
  - Everyone else: things (entities, individuals) that each have properties, and there are relationships among them

Motivation

- AI studies what agents should do.
  - Acting is gambling: agents who don't use probabilities will lose to those who do.
  - No prediction is certain: never believe anyone who gives definitive predictions!
- What is the world made up of?
  - ML: Features or random variables
  - Everyone else: things (entities, individuals) that each have properties, and there are relationships among them
- How can we reconcile these?

Motivation

- AI studies what agents should do.
  - Acting is gambling: agents who don't use probabilities will lose to those who do.
  - No prediction is certain: never believe anyone who gives definitive predictions!
- What is the world made up of?
  - ML: Features or random variables
  - Everyone else: things (entities, individuals) that each have properties, and there are relationships among them
- How can we reconcile these?



Outline

1. What are relational probabilistic models and relational learning?
   - Relational Models
   - Knowledge Graphs
2. Learning Knowledge Graphs
3. Learning General Knowledge: Lifted Graphical Models
4. Bayesian $\Rightarrow$ Exchangeability $\Rightarrow$ Lifted Inference
5. Identity and Existence Uncertainty

What is a relational models?

Introductions to AI and machine learning typically start with learning from relations, e.g.:

Example	*Author*	*Thread*	*Length*	*Where_read*	*User_action*
e_1	*known*	*new*	*long*	*home*	*skips*
e_2	*unknown*	*new*	*short*	*work*	*reads*
...	...	...	...	...	...

What is a relational models?

Introductions to AI and machine learning typically start with learning from relations, e.g.:

Example	*Author*	*Thread*	*Length*	*Where_read*	*User_action*
e_1	*known*	*new*	*long*	*home*	*skips*
e_2	*unknown*	*new*	*short*	*work*	*reads*
...	...	...	...	...	...

What makes relational models in ML special is that the values are meaningless names. E.g., student #, product id, user id, movie id:

User	*Movie*	*Rating*	*Timestamp*
196	242	3	881250949
186	302	3	891717742
...	...	...	...

(Movielens 100k)

Names can be changed or exchanged with exactly same meaning.

Choosing Entities and Relations in Logic

First-order logical languages allow many different ways of representing facts.

E.g., How to represent: "Pen #7 is red."

- $red(pen_7)$.
- $color(pen_7, red)$.
- $prop(pen_7, color, red)$.

Choosing Entities and Relations in Logic

First-order logical languages allow many different ways of representing facts.

E.g., How to represent: "Pen #7 is red."

- $red(pen_7)$.
- $color(pen_7, red)$.
- $prop(pen_7, color, red)$.
- a single relation can be implicit $\longrightarrow$ triples: $(pen_7, color, red)$.

Triples are universal representations of relations

All relations can be represented in terms of triples:

	...	P_j	...
	...	...	...
r_i	...	v_{ij}	...
	...	...	...

can be represented as

Triples are universal representations of relations

All relations can be represented in terms of triples:

	...	P_j	...
	...	...	...
r_i	...	v_{ij}	...
	...	...	...

can be represented as the triple (r_i, P_j, v_{ij}).

- r_i is either a primary key or a reified entity.

Triples are universal representations of relations

All relations can be represented in terms of triples:

	...	P_j	...
	...	...	...
r_i	...	v_{ij}	...
	...	...	...

can be represented as the triple (r_i, P_j, v_{ij}).

- r_i is either a primary key or a reified entity.
- Examples of reified entities: a booking, a marriage, a talk, a lab report, an event, a party, a meeting, a drink

Triples are universal representations of relations

All relations can be represented in terms of triples:

	...	P_j	...
	...	...	...
r_i	...	v_{ij}	...
	...	...	...

can be represented as the triple (r_i, P_j, v_{ij}).

- r_i is either a primary key or a reified entity.
- Examples of reified entities: a booking, a marriage, a talk, a lab report, an event, a party, a meeting, a drink

$prop(Entity, Property, Value)$ is the only relation needed:
$(Entity, Property, Value)$ triples, semantic network, entity relationship model, knowledge graphs, ...

Warning: Many knowledge graphs convert to triples naively

Projecting onto pairs loses information:

- For example:
  Air Canada flies from New York to Vancouver
  Air Canada flies from Vancouver to Los Angeles

Warning: Many knowledge graphs convert to triples naively

Projecting onto pairs loses information:

- For example:
  Air Canada flies from New York to Vancouver
  Air Canada flies from Vancouver to Los Angeles
- These are true triples:
  $(Air\ Canada, Flies\ From, New\ York)$
  $(Air\ Canada, Flies\ To, Los\ Angeles)$
- However, Air Canada does not fly from New York to Los Angeles.
  The information about flights is lost!

Warning: Many knowledge graphs convert to triples naively

FB15K, a knowledge base commonly used in research papers, contains test cases:

- (Jade North, /sports/pro_athlete/teams./soccer/football_roster_position/position, Defender (association football))
  "*Jade North plays position defender.*"
- (Real Zaragoza, /soccer/football_team/current_roster./sports/sports_team_roster/position, Defender (association football))
  "*Real Zaragoza football club has position defender.*"

Predicting one position in the tuple given two others varies widely in difficulty!

Please look at a knowledge graph before you use it!

Entities are not like words

- When representing words as vectors, interesting relations are learned:

$$king - man + woman = queen$$

Entities are not like words

- When representing words as vectors, interesting relations are learned:

$$king - man + woman = queen$$

- It is tempting to want (translational models):

$$Brussels = Belgium + capital\_of$$

$$Washington\_DC = USA + capital\_of$$

Entities are not like words

- When representing words as vectors, interesting relations are learned:

$$king - man + woman = queen$$

- It is tempting to want (translational models):

$$Brussels = Belgium + capital\_of$$

$$Washington\_DC = USA + capital\_of$$

- But this entails:

$$USA = Belgium - Brussels + Washington\_DC$$

Entities are not like words

- When representing words as vectors, interesting relations are learned:

$$king - man + woman = queen$$

- It is tempting to want (translational models):

$$Brussels = Belgium + capital\_of$$

$$Washington\_DC = USA + capital\_of$$

- But this entails:

$$USA = Belgium - Brussels + Washington\_DC$$

- Words can have simple meanings but (almost all) entities are multi-faceted and complex.

Outline

Aside: Probabilities $\leftrightarrow$ sigmoid

$$P(h \mid e) \quad = \quad \frac{P(h \wedge e)}{P(h \wedge e) + P(\neg h \wedge e)}$$

Aside: Probabilities $\leftrightarrow$ sigmoid

$$
\begin{aligned}
P(h \mid e) &= \frac{P(h \wedge e)}{P(h \wedge e) + P(\neg h \wedge e)} \\
&= \frac{1}{1 + P(\neg h \wedge e)/P(h \wedge e)}
\end{aligned}
$$

Aside: Probabilities $\leftrightarrow$ sigmoid

$$
\begin{aligned}
P(h \mid e) &= \frac{P(h \wedge e)}{P(h \wedge e) + P(\neg h \wedge e)} \\
&= \frac{1}{1 + P(\neg h \wedge e)/P(h \wedge e)} \\
&= \frac{1}{1 + e^{-(\log P(h \wedge e)/P(\neg h \wedge e))}}
\end{aligned}
$$

Aside: Probabilities $\leftrightarrow$ sigmoid

$$
\begin{aligned}
P(h \mid e) &= \frac{P(h \wedge e)}{P(h \wedge e) + P(\neg h \wedge e)} \\
&= \frac{1}{1 + P(\neg h \wedge e)/P(h \wedge e)} \\
&= \frac{1}{1 + e^{-(\log P(h \wedge e)/P(\neg h \wedge e))}} \\
&= \mathit{sigmoid}(\log \mathit{odds}(h \mid e))
\end{aligned}
$$

$$\mathit{sigmoid}(x) = 1/(1 + e^{-x})$$

$$\mathit{odds}(h \mid e) = \frac{P(h \wedge e)}{P(\neg h \wedge e)} = \frac{P(h \mid e)}{1 - P(h \mid e)}$$

Odds is a product $\Rightarrow$ sigmoid of a sum $\rightarrow$ logistic regression

Aside: Probabilities $\leftrightarrow$ sigmoid

$$\begin{aligned}
P(h \mid e) &= \frac{P(h \wedge e)}{P(h \wedge e) + P(\neg h \wedge e)} \\
&= \frac{1}{1 + P(\neg h \wedge e)/P(h \wedge e)} \\
&= \frac{1}{1 + e^{-(\log P(h \wedge e)/P(\neg h \wedge e))}} \\
&= \mathit{sigmoid}(\log \mathit{odds}(h \mid e))
\end{aligned}$$

$$\mathit{sigmoid}(x) = 1/(1 + e^{-x})$$

$$\mathit{odds}(h \mid e) = \frac{P(h \wedge e)}{P(\neg h \wedge e)} = \frac{P(h \mid e)}{1 - P(h \mid e)}$$

Odds is a product $\Rightarrow$ sigmoid of a sum $\rightarrow$ logistic regression

Typical: to learn probability of

- Boolean feature: sigmoid of a linear function
- discrete feature: softmax of a linear function

Vector & Tensor Representations of Entities & Relations

- To learn a binary relation, e.g., *likes*(*Person*, *Movie*) in pseudo Python:

$$P(likes(p, m)) = sigmoid(\sum_f E_0[p][f] * E_1[m][f])$$

Vector & Tensor Representations of Entities & Relations

- To learn a binary relation, e.g., *likes*(*Person*, *Movie*) in pseudo Python:

$$P(likes(p, m)) = sigmoid(\sum_f E_0[p][f] * E_1[m][f])$$

— matrix factorization.
Embedding = a vector of feature values
Embedding for each person ($E_0[p]$) and movie ($E_1[m]$)

Vector & Tensor Representations of Entities & Relations

- To learn a binary relation, e.g., *likes*(*Person*, *Movie*) in pseudo Python:

$$P(likes(p, m)) = sigmoid(\sum_f E_0[p][f] * E_1[m][f])$$

  — matrix factorization.
  Embedding = a vector of feature values
  Embedding for each person ($E_0[p]$) and movie ($E_1[m]$)
- To learn triple: (h, r, t)

Vector & Tensor Representations of Entities & Relations

- To learn a binary relation, e.g., $likes(Person, Movie)$ in pseudo Python:

$$P(likes(p, m)) = sigmoid(\sum_f E_0[p][f] * E_1[m][f])$$

— matrix factorization.
Embedding = a vector of feature values
Embedding for each person ($E_0[p]$) and movie ($E_1[m]$)

- To learn triple: (h, r, t)

$$P((h, r, t)) = sigmoid(\sum_f E_0[h][f] * E_1[r][f] * E_2[t][f])$$

Vector & Tensor Representations of Entities & Relations

- To learn a binary relation, e.g., $likes(Person, Movie)$ in pseudo Python:

$$P(likes(p, m)) = sigmoid(\sum_f E_0[p][f] * E_1[m][f])$$

— matrix factorization.
Embedding = a vector of feature values
Embedding for each person ($E_0[p]$) and movie ($E_1[m]$)

- To learn triple: (h, r, t)

$$P((h, r, t)) = sigmoid(\sum_f E_0[h][f] * E_1[r][f] * E_2[t][f])$$

— polyadic decomposition model (1927): two vector embeddings for each entity e ($E_0[e]$ and $E_2[e]$) and one for reach relation r ($E_1[r]$).

Polyadic decomposition variations

- Polyadic decomposition doesn't work very well...
  - Consider $(p123, \mathit{likes}, m53)$ and $(m53, \mathit{directed\_by}, p534)$.

Polyadic decomposition variations

- Polyadic decomposition doesn't work very well...
  - Consider $(p123, \mathit{likes}, m53)$ and $(m53, \mathit{directed\_by}, p534)$.
  - Requires two embeddings per entity, but head embeddings and tail embeddings do not interact.

Polyadic decomposition variations

- Polyadic decomposition doesn't work very well...
  - Consider $(p123, likes, m53)$ and $(m53, directed\_by, p534)$.
  - Requires two embeddings per entity, but head embeddings and tail embeddings do not interact.
- DistMult: share same embedding for head and tail.

Polyadic decomposition variations

- Polyadic decomposition doesn't work very well...
  - Consider $(p123, \mathit{likes}, m53)$ and $(m53, \mathit{directed\_by}, p534)$.
  - Requires two embeddings per entity, but head embeddings and tail embeddings do not interact.
- DistMult: share same embedding for head and tail.
  Problem: can only represent symmetric relations.

Polyadic decomposition variations

- Polyadic decomposition doesn't work very well...
  - Consider $(p123, likes, m53)$ and $(m53, directed\_by, p534)$.
  - Requires two embeddings per entity, but head embeddings and tail embeddings do not interact.
- DistMult: share same embedding for head and tail.
  Problem: can only represent symmetric relations.
- CompleX: like DistMult, but the embeddings are complex numbers, tail is the conjugate of the head embedding

Polyadic decomposition variations

- Polyadic decomposition doesn't work very well...
  - Consider $(p123, \mathit{likes}, m53)$ and $(m53, \mathit{directed\_by}, p534)$.
  - Requires two embeddings per entity, but head embeddings and tail embeddings do not interact.
- DistMult: share same embedding for head and tail.
  Problem: can only represent symmetric relations.
- CompleX: like DistMult, but the embeddings are complex numbers, tail is the conjugate of the head embedding
- SimpleE: have an embedding for r^{-1} and learn to predict both (h, r, t) and $\left(t, r^{-1}, h\right)$

Polyadic decomposition variations

- Polyadic decomposition doesn't work very well...
  - Consider $(p123, \textit{likes}, m53)$ and $(m53, \textit{directed\_by}, p534)$.
  - Requires two embeddings per entity, but head embeddings and tail embeddings do not interact.
- DistMult: share same embedding for head and tail.
  Problem: can only represent symmetric relations.
- CompleX: like DistMult, but the embeddings are complex numbers, tail is the conjugate of the head embedding
- SimpleE: have an embedding for r^{-1} and learn to predict both (h, r, t) and $\left(t, r^{-1}, h\right)$
- SimpleE^{+} = SimplE with non-negative entity embeddings
  - can represent arbitrary relations
  - pointwise $\leq$ corresponds to implication
  - easy to explain what it learns

What/how embedding-based models learn

PD$^+$:

$$P((h,r,t)) = sigmoid(\sum_f E_0[h][f] * E_1[r][f] * E_2[t][f])$$

$E_0[h][f] \geq 0$ $E_2[h][f] \geq 0$.
Assume all embedding values are bounded.

What/how embedding-based models learn

PD$^+$:

$$P((h,r,t)) = sigmoid(\sum_f E_0[h][f] * E_1[r][f] * E_2[t][f])$$

$E_0[h][f] \geq 0$ $E_2[h][f] \geq 0$.
Assume all embedding values are bounded.

- Consider feature i: $E_0[h][i] * E_1[r][i] * E_2[t][i] \approx 0$ if $E_0[h][i] \approx 0$ or $E_1[r][i] \approx 0$ or $E_2[r][i] \approx 0$.

What/how embedding-based models learn

PD^+:

$$P((h, r, t)) = sigmoid(\sum_f E_0[h][f] * E_1[r][f] * E_2[t][f])$$

$E_0[h][f] \geq 0$ $E_2[h][f] \geq 0$.
Assume all embedding values are bounded.

- Consider feature i: $E_0[h][i] * E_1[r][i] * E_2[t][i] \approx 0$ if $E_0[h][i] \approx 0$ or $E_1[r][i] \approx 0$ or $E_2[r][i] \approx 0$.
- Consider feature i: $E_0[h][i] * E_1[r][i] * E_2[t][i] \gg 0$ if $E_0[h][i] \gg 0$ and $E_1[r][i] \gg 0$ and $E_2[t][i] \gg 0$.

What/how embedding-based models learn

PD^+:

$$P((h,r,t)) = sigmoid(\sum_f E_0[h][f] * E_1[r][f] * E_2[t][f])$$

$E_0[h][f] \geq 0$ $E_2[h][f] \geq 0$.
Assume all embedding values are bounded.

- Consider feature i: $E_0[h][i] * E_1[r][i] * E_2[t][i] \approx 0$ if $E_0[h][i] \approx 0$ or $E_1[r][i] \approx 0$ or $E_2[r][i] \approx 0$.
- Consider feature i: $E_0[h][i] * E_1[r][i] * E_2[t][i] \gg 0$ if $E_0[h][i] \gg 0$ and $E_1[r][i] \gg 0$ and $E_2[t][i] \gg 0$.
- Feature i forms two soft clusterings of entities:
  - those e for which $E_0[e][i]$ is high
  - those e for which $E_2[e][i]$ is high

What/how embedding-based models learn

PD$^+$:

$$P((h,r,t)) = sigmoid(\sum_f E_0[h][f] * E_1[r][f] * E_2[t][f])$$

$E_0[h][f] \geq 0$ $E_2[h][f] \geq 0$.
Assume all embedding values are bounded.

- Consider feature i: $E_0[h][i] * E_1[r][i] * E_2[t][i] \approx 0$ if $E_0[h][i] \approx 0$ or $E_1[r][i] \approx 0$ or $E_2[r][i] \approx 0$.
- Consider feature i: $E_0[h][i] * E_1[r][i] * E_2[t][i] \gg 0$ if $E_0[h][i] \gg 0$ and $E_1[r][i] \gg 0$ and $E_2[t][i] \gg 0$.
- Feature i forms two soft clusterings of entities:
  - those e for which $E_0[e][i]$ is high
  - those e for which $E_2[e][i]$ is high

  The entities in the first cluster are related to the entities in the second cluster for any relations for which $E_1[r][i]$ is high.

What/how embedding-based models learn

PD$^+$:

$$P((h,r,t)) = sigmoid(\sum_f E_0[h][f] * E_1[r][f] * E_2[t][f])$$

$E_0[h][f] \geq 0$ $E_2[h][f] \geq 0$.
Assume all embedding values are bounded.

- Consider feature i: $E_0[h][i] * E_1[r][i] * E_2[t][i] \approx 0$
  if $E_0[h][i] \approx 0$ or $E_1[r][i] \approx 0$ or $E_2[r][i] \approx 0$.
- Consider feature i: $E_0[h][i] * E_1[r][i] * E_2[t][i] \gg 0$
  if $E_0[h][i] \gg 0$ and $E_1[r][i] \gg 0$ and $E_2[t][i] \gg 0$.
- Feature i forms two soft clusterings of entities:
  - those e for which $E_0[e][i]$ is high
  - those e for which $E_2[e][i]$ is high

  The entities in the first cluster are related to the entities in the second cluster for any relations for which $E_1[r][i]$ is high.
- Negative values of $E_1[r][i]$ provide exceptions.

Learning general knowledge vs learning about a data set

- Suppose you want to create a model of who is friends with whom.

Learning general knowledge vs learning about a data set

- Suppose you want to create a model of who is friends with whom. Options:
  - learn general knowledge, e.g., transitivity, how male and female friendships work, how location affect friendship...

Learning general knowledge vs learning about a data set

- Suppose you want to create a model of who is friends with whom. Options:
  - learn general knowledge, e.g., transitivity, how male and female friendships work, how location affect friendship...
  - learn specific knowledge about who is friends with who; e.g., which particular group of people are generally friends with each other.

Learning general knowledge vs learning about a data set

- Suppose you want to create a model of who is friends with whom. Options:
  - learn general knowledge, e.g., transitivity, how male and female friendships work, how location affect friendship...
  - learn specific knowledge about who is friends with who; e.g., which particular group of people are generally friends with each other.
- The specific knowledge will tend to be more accurate on that population, but doesn't generalize to different populations.
- The general knowledge will tend to transfer better.
- Which is better depends on the goals and how success is measured.

Learning general knowledge vs learning about a data set

- Suppose you want to create a model of who is friends with whom. Options:
  - learn general knowledge, e.g., transitivity, how male and female friendships work, how location affect friendship...
  - learn specific knowledge about who is friends with who; e.g., which particular group of people are generally friends with each other.
- The specific knowledge will tend to be more accurate on that population, but doesn't generalize to different populations.
- The general knowledge will tend to transfer better.
- Which is better depends on the goals and how success is measured.
- Ideally we would try to do both; learn about specific entities and general knowledge.

Challenges of learning knowledge graphs

- Evaluating predictions when only positive examples are provided
  Consider the following relations:
  - Married to
  - Friend of
  - Knows about
  - Would get along with

Challenges of learning knowledge graphs

- Evaluating predictions when only positive examples are provided
  Consider the following relations:
  - Married to — each person related to 0 or 1 other persons (with a few exceptions)
  - Friend of — each person related to tens or hundreds of others
  - Knows about — each person might know about hundreds or thousands of others. Some people my be known by millions or billions of others.
  - Would get along with — almost everyone gets along with almost everyone else, but with some exceptions.

Beware of ranking

- Most knowledge graphs only contain positive information.
- How can we evaluate a prediction?
  (?, Plays Position, Defender)
  (Jade North, Plays Position, ?)
  (Real Zaragoza, Has Position, ?)
  (?, Has Position, Defender)

Beware of ranking

- Most knowledge graphs only contain positive information.
- How can we evaluate a prediction?
  (?, Plays Position, Defender)
  (Jade North, Plays Position, ?)
  (Real Zaragoza, Has Position, ?)
  (?, Has Position, Defender)
- Common to use measures based on ranking such as
  mean reciprocal rank (MRR) or Hit@1 or Hit@10.
- Problem #1: is it not good for answers for which there is no answer or many answers:
  Who is the pope married to?

Beware of ranking

- Most knowledge graphs only contain positive information.
- How can we evaluate a prediction?
  (?, Plays Position, Defender)
  (Jade North, Plays Position, ?)
  (Real Zaragoza, Has Position, ?)
  (?, Has Position, Defender)
- Common to use measures based on ranking such as
  mean reciprocal rank (MRR) or Hit@1 or Hit@10.
- Problem #1: is it not good for answers for which there is no answer or many answers:
  Who is the pope married to? Who likes Drake's music?

Beware of ranking

- Most knowledge graphs only contain positive information.
- How can we evaluate a prediction?
  (?, Plays Position, Defender)
  (Jade North, Plays Position, ?)
  (Real Zaragoza, Has Position, ?)
  (?, Has Position, Defender)
- Common to use measures based on ranking such as
  mean reciprocal rank (MRR) or Hit@1 or Hit@10.
- Problem #1: is it not good for answers for which there is no answer or many answers:
  Who is the pope married to? Who likes Drake's music?
- Problem #2: an oracle that knows everything does poorly on ranking scores!

Beware of ranking

- Most knowledge graphs only contain positive information.
- How can we evaluate a prediction?
  (?, Plays Position, Defender)
  (Jade North, Plays Position, ?)
  (Real Zaragoza, Has Position, ?)
  (?, Has Position, Defender)
- Common to use measures based on ranking such as mean reciprocal rank (MRR) or Hit@1 or Hit@10.
- Problem #1: is it not good for answers for which there is no answer or many answers:
  Who is the pope married to? Who likes Drake's music?
- Problem #2: an oracle that knows everything does poorly on ranking scores!
- Challenge: design a good evaluation scheme. Log-likelihood seems reasonable, but requires knowledge of negations.

Predicting Properties

- Tensor factorization models work well for predicting relations, but not for predicting properties.
  Tensor factorization relies on lower-dimensional representations, and there isn't one for properties.

Predicting Properties

- Tensor factorization models work well for predicting relations, but not for predicting properties.
  Tensor factorization relies on lower-dimensional representations, and there isn't one for properties.
- Imagine trying to predict $age(P)$, the age of person P, and $rating(P, M)$ the rating of person P on movie M.
  One of the embeddings of each person can just memorize the age — no generalization!
  — there are too many parameters

Predicting Properties

- Tensor factorization models work well for predicting relations, but not for predicting properties.
  Tensor factorization relies on lower-dimensional representations, and there isn't one for properties.
- Imagine trying to predict $age(P)$, the age of person P, and $rating(P, M)$ the rating of person P on movie M.
  One of the embeddings of each person can just memorize the age — no generalization!
  — there are too many parameters
- We need to predict the age using properties of the movies and ratings.

Predicting Properties

- Tensor factorization models work well for predicting relations, but not for predicting properties.
  Tensor factorization relies on lower-dimensional representations, and there isn't one for properties.
- Imagine trying to predict $age(P)$, the age of person P, and $rating(P, M)$ the rating of person P on movie M.
  One of the embeddings of each person can just memorize the age — no generalization!
  — there are too many parameters
- We need to predict the age using properties of the movies and ratings.
- Requires aggregation: some models provide implicit aggregation, and some you can use whatever aggregation you want. We need better models of aggregation.

Beyond Triples

If we have relations with multiple arguments:

- We could convert them to triples by reifying

Beyond Triples

If we have relations with multiple arguments:

- We could convert them to triples by reifying ... but the reified entities have very few data points (number of arguments of original relations)

Beyond Triples

If we have relations with multiple arguments:

- We could convert them to triples by reifying ... but the reified entities have very few data points (number of arguments of original relations)
- Design embedding-based model that work directly with original relations
- Allow them to be inferred from other relations

Outline

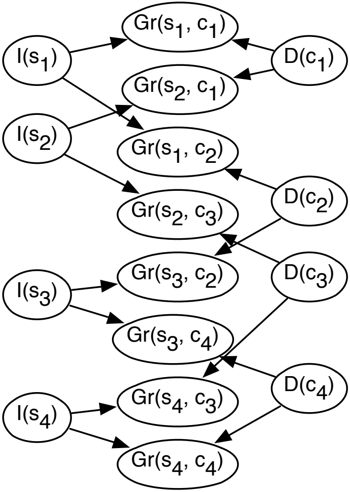
1. What are relational probabilistic models and relational learning?
   - Relational Models
   - Knowledge Graphs
2. Learning Knowledge Graphs
3. Learning General Knowledge: Lifted Graphical Models
4. Bayesian $\Rightarrow$ Exchangeability $\Rightarrow$ Lifted Inference
5. Identity and Existence Uncertainty

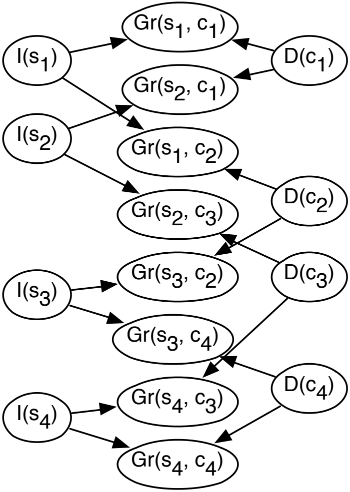
Example: Predicting Relations

Student	*Course*	*Grade*
s_1	c_1	A
s_2	c_1	C
s_1	c_2	B
s_2	c_3	B
s_3	c_2	B
s_4	c_3	B
s_3	c_4	?
s_4	c_4	?

- Students s_3 and s_4 have the same averages, on courses with the same averages.
- Which student would you expect to better?

From Relations to Bayesian Belief Networks



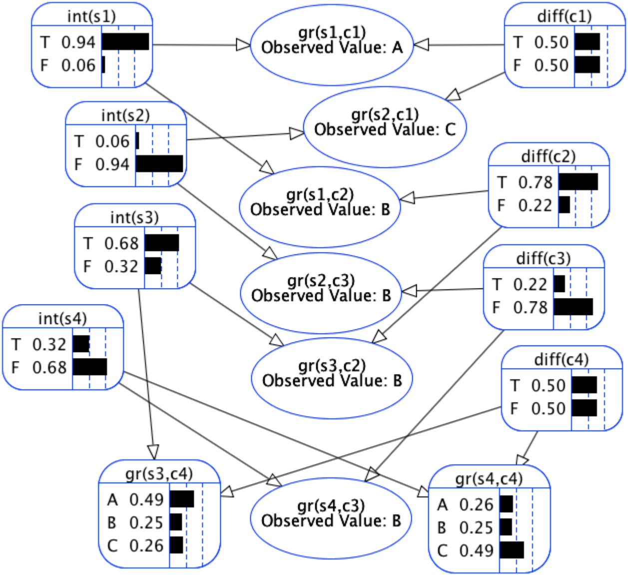


$I(S)$	$D(C)$	$Gr(S,C)$ A	B	C
true	*true*	0.5	0.4	0.1
true	*false*	0.9	0.09	0.01
false	*true*	0.01	0.09	0.9
false	*false*	0.1	0.4	0.5

$P(I(S)) = 0.5$
$P(D(C)) = 0.5$

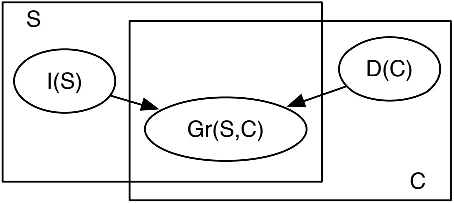
"parameter sharing"

Example: Predicting Relations



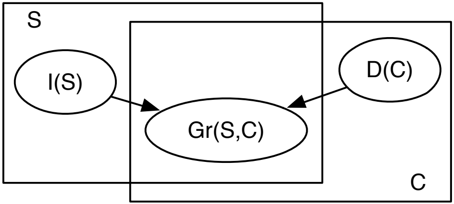
`http://artint.info/code/aispace/grades.xml`

Plate Notation



- S, C logical variable representing students, courses
- the set of entities of a type is called a population
- $I(S)$, $Gr(S, C)$, $D(C)$ are parametrized random variables

Plate Notation

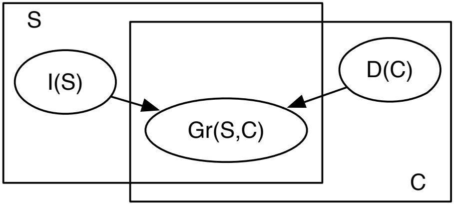


- S, C logical variable representing students, courses
- the set of entities of a type is called a population
- $I(S)$, $Gr(S, C)$, $D(C)$ are parametrized random variables

Grounding:

- for every student s, there is a random variable $I(s)$
- for every course c, there is a random variable $D(c)$

Plate Notation

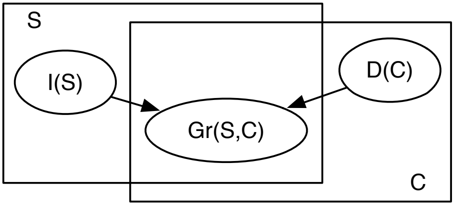


- S, C logical variable representing students, courses
- the set of entities of a type is called a population
- $I(S)$, $Gr(S, C)$, $D(C)$ are parametrized random variables

Grounding:

- for every student s, there is a random variable $I(s)$
- for every course c, there is a random variable $D(c)$
- for every s, c pair there is a random variable $Gr(s, c)$

Plate Notation

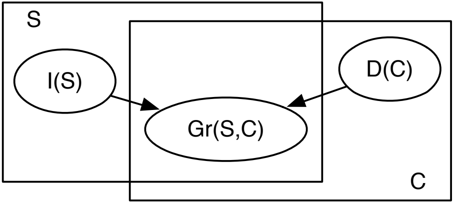


- S, C logical variable representing students, courses
- the set of entities of a type is called a population
- $I(S)$, $Gr(S, C)$, $D(C)$ are parametrized random variables

Grounding:

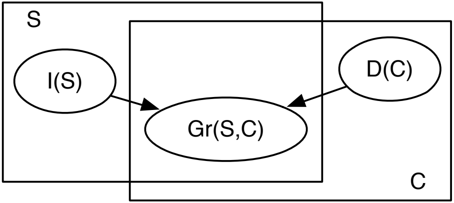
- for every student s, there is a random variable $I(s)$
- for every course c, there is a random variable $D(c)$
- for every s, c pair there is a random variable $Gr(s, c)$
- all instances share the same structure and parameters

Plate Notation



- If there were 1000 students and 100 courses:
  Grounding contains

Plate Notation



- If there were 1000 students and 100 courses:
  Grounding contains
  - 1000 $I(s)$ variables
  - 100 $D(c)$ variables
  - 100000 $Gr(s, c)$ variables

  total: 101100 variables
- To define the probabilities:
  1 for $I(S)$, 1 for $D(C)$, 8 for $Gr(S, C) = 10$ parameters.

Representations of Lifted Graphical models

How is a relation affected by other relationships? “aggregation”

Representations of Lifted Graphical models

How is a relation affected by other relationships? "aggregation"
Common representations:

	Undirected	Directed
Weighted Formulas	Markov Logic Networks (MLNs)	Relational Logistic Regression (RLR)

Representations of Lifted Graphical models

How is a relation affected by other relationships? "aggregation"
Common representations:

	Undirected	Directed
Weighted Formulas	Markov Logic Networks (MLNs)	Relational Logistic Regression (RLR)
Existential quantification (logic programs)		Independent Choice Logic / Problog
…	…	…

Representations of Lifted Graphical models

How is a relation affected by other relationships? “aggregation”
Common representations:

	Undirected	Directed
Weighted Formulas	Markov Logic Networks (MLNs)	Relational Logistic Regression (RLR)
Existential quantification (logic programs)		Independent Choice Logic / Problog
...	...	...

- MLNs and RLR are identical when “everything else” is observed.

Representations of Lifted Graphical models

How is a relation affected by other relationships? "aggregation"
Common representations:

	Undirected	Directed
Weighted Formulas	Markov Logic Networks (MLNs)	Relational Logistic Regression (RLR)
Existential quantification (logic programs)		Independent Choice Logic / Problog
...	...	...

- MLNs and RLR are identical when "everything else" is observed.
- Also: relational dependency networks: directed models that induce a Markov chain.

Example of polynomial dependence of population

$\alpha_0 : q$
$\alpha_2 : q \wedge r(X)$
$\alpha_4 : r(X)$
$\alpha_7 : q \wedge r(X) \wedge r(Y)$

Example of polynomial dependence of population

$$\alpha_0 : q$$
$$\alpha_2 : q \wedge r(X)$$
$$\alpha_4 : r(X)$$
$$\alpha_7 : q \wedge r(X) \wedge r(Y)$$

In RLR and in MLN, if all $r(a_i)$ are observed:

Example of polynomial dependence of population

$$\begin{aligned}
&\alpha_0 : q \\
&\alpha_2 : q \wedge r(X) \\
&\alpha_4 : r(X) \\
&\alpha_7 : q \wedge r(X) \wedge r(Y)
\end{aligned}$$

In RLR and in MLN, if all $r(a_i)$ are observed:

$$P(q \mid obs) = sigmoid(\alpha_0 + n_1\alpha_2 + n_1^2\alpha_7)$$

$r(X)$ is true for n_1 individuals

Danger of fitting to data without understanding the model

- Consider sigmoid of polynomials of degree 2:

  $$sigmoid(-0.01n^2 - 0.2n + 8)$$

  $$sigmoid(0.01n^2 - n + 16)$$

  Both go from ≈ 1 at $n = 10$ to ≈ 0 at $n = 30$.
  What happens as $n \to \infty$?

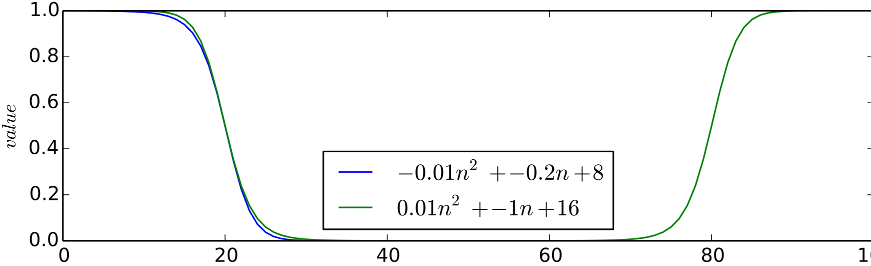
Danger of fitting to data without understanding the model

- Consider sigmoid of polynomials of degree 2:

$$sigmoid(-0.01n^2 - 0.2n + 8)$$

$$sigmoid(0.01n^2 - n + 16)$$

Both go from ≈ 1 at $n = 10$ to ≈ 0 at $n = 30$.
What happens as $n \to \infty$?



Outline

Exchangeability

- Before we know anything about entities, they are indistinguishable, and so should be treated identically.
  exchangeability — names can be exchanged and the model doesn't change.

Exchangeability

- Before we know anything about entities, they are indistinguishable, and so should be treated identically.
  exchangeability — names can be exchanged and the model doesn't change.
- Bayesianism: probability depends on what is known (conditioning)

Exchangeability

- Before we know anything about entities, they are indistinguishable, and so should be treated identically.
  exchangeability — names can be exchanged and the model doesn't change.
- Bayesianism: probability depends on what is known (conditioning)
- Entities we have the same information about must have same probability.

Exchangeability

- Before we know anything about entities, they are indistinguishable, and so should be treated identically.
  exchangeability — names can be exchanged and the model doesn't change.
- Bayesianism: probability depends on what is known (conditioning)
- Entities we have the same information about must have same probability.
- This provides a symmetry that can be exploited in Lifted Inference.

Exchangeability

- Before we know anything about entities, they are indistinguishable, and so should be treated identically.
  exchangeability — names can be exchanged and the model doesn't change.
- Bayesianism: probability depends on what is known (conditioning)
- Entities we have the same information about must have same probability.
- This provides a symmetry that can be exploited in Lifted Inference.
- See Guy Van den Broeck's *Computers and Thought* lecture from IJCAI 2019.

Outline

Identity

- (Lifted) inference requires counting.

Identity

- (Lifted) inference requires counting.
- Example: in the room was
  - Sam's mother
  - Chris's football coach
  - a brilliant mathematician

  How many people were in the room?

Identity

- (Lifted) inference requires counting.
- Example: in the room was
  - Sam's mother
  - Chris's football coach
  - a brilliant mathematician

  How many people were in the room?

  Answer: at least one

Identity

- (Lifted) inference requires counting.
- Example: in the room was
  - Sam's mother
  - Chris's football coach
  - a brilliant mathematician

  How many people were in the room?

  Answer: at least one
- If we also specified that there was no one else: there are between 1 and 3 people.

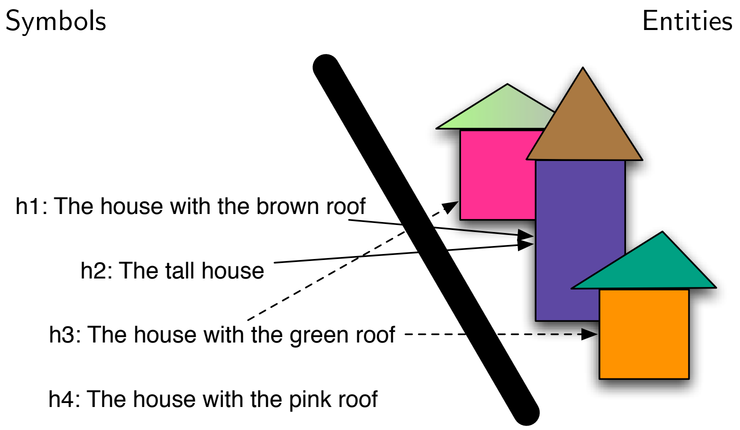
Identity

- (Lifted) inference requires counting.
- Example: in the room was
  - Sam's mother
  - Chris's football coach
  - a brilliant mathematician

  How many people were in the room?

  Answer: at least one
- If we also specified that there was no one else: there are between 1 and 3 people.
- We need knowledge graphs to (be able to) state "there are no more ..."

Correspondence Problem



c symbols and i entities $\longrightarrow c^{i+1}$ correspondences

Clarity Principle

Clarity principle: probabilities must be over well-defined propositions.

- What if an entity doesn't exist?
  - $house(h4) \land roof\_colour(h4, pink) \land \neg exists(h4)$

Clarity Principle

Clarity principle: probabilities must be over well-defined propositions.

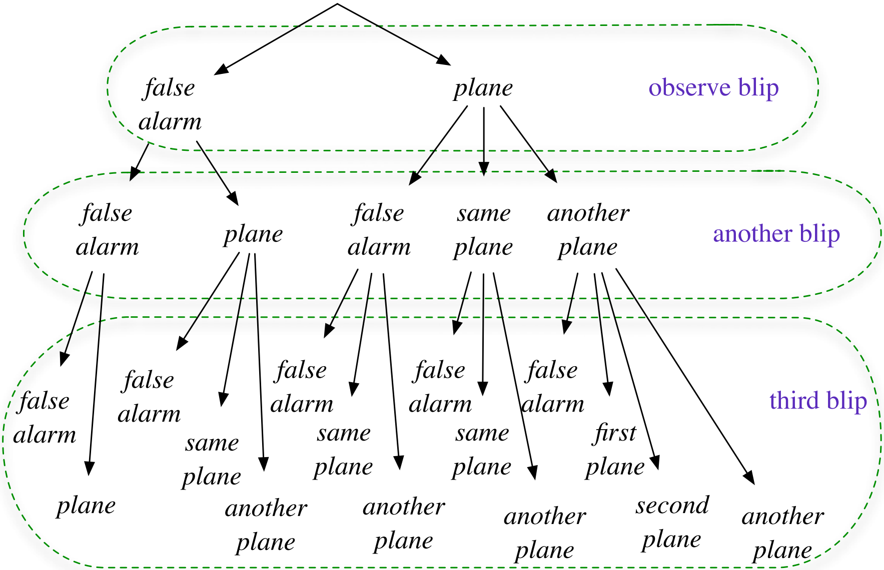
- What if an entity doesn't exist?
  - $house(h4) \wedge roof\_colour(h4, pink) \wedge \neg exists(h4)$

- What if more than one entity exists? Which one are we referring to?
  —In a house with three bedrooms, which is the second bedroom?

Handling Number and Existence Uncertainty

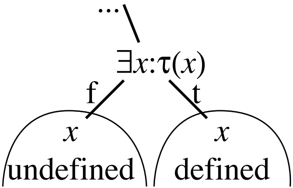
- distribution over the number of entities. For each number, reason about the correspondence.
- For each observation in sequence, hypothesize its correspondance e.g., if you observe a radar blip, there are three hypotheses:
  - the blip was produced by plane you already hypothesized
  - the blip was produced by another plane
  - the blip wasn't produced by a plane

Existence Example



First-order Semantic Trees

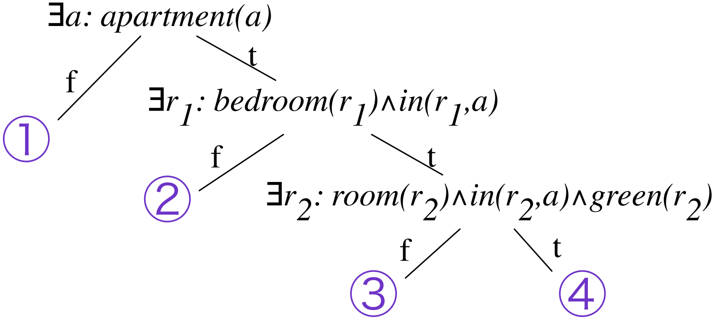
Split on quantified first-order formulae:



- The “true” sub-tree is in the scope of x
- The “false” sub-tree is not in the scope of x

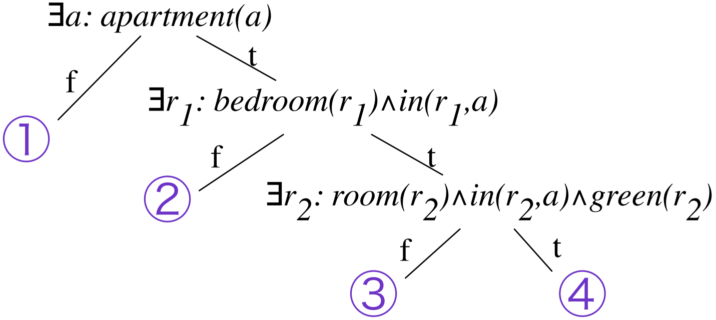
A logical generative model generates a first-order semantic tree.

First-order Semantic Tree (cont)



1. there is no apartment
2. there is no bedroom in the apartment
3. there is a bedroom but no green room
4. there is a bedroom and a green room

First-order Semantic Tree (cont)



① there is no apartment

② there is no bedroom in the apartment

③ there is a bedroom but no green room

④ there is a bedroom and a green room

All probabilities are over well-defined first-order formulae.

Conclusion

Challenge model and learn uncertainty about:

- Properties of entities
- Relationships among entities
- How properties and relations interrelate
- Identity (equality) of entities
- Existence (and number) of entities
- Interactions with time, ontologies, causality ...

Will you step up to this challenge? There is still lots to do!

What is now required is to give the greatest possible development to mathematical logic, to allow to the full the importance of relations, and then to found upon this secure basis a new philosophical logic, which may hope to borrow some of the exactitude and certainty of its mathematical foundation. If this can be successfully accomplished, there is every reason to hope that the near future will be as great an epoch in pure philosophy as the immediate past has been in the principles of mathematics. Great triumphs inspire great hopes; and pure thought may achieve, within our generation, such results as will place our time, in this respect, on a level with the greatest age of Greece.

– Bertrand Russell, Mysticism and Logic and Other Essays (1917)