Learning and reasoning about entities and relations under uncertainty: a story of romance and disappointment

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

September 19, 2024

"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.

What is the real world made of?

A Features or random variables

B Words, pixels, phonemes . . .

C Entities and events (e.g., plants, people, diseases, lectures, university courses)

D Huh? There is a real world?

Outline

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

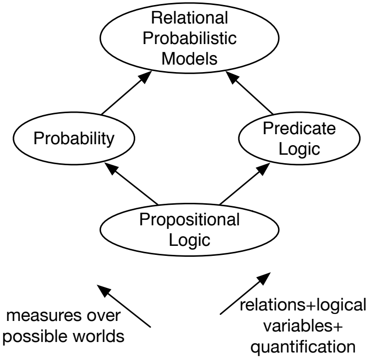
- AI studies what agents should do.
  - Acting is gambling: agents who don't use probabilities will lose to those who do.
  - No prediction is certain: never believe anyone who gives definitive predictions!
- What is the world made up of?
  - ML: Features or random variables
  - Everyone else: things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

What are 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 are 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 include meaningless names. E.g., student number, 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: $(pen_7, color, red)$.

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:

$$\begin{array}{c|c|c|c|} \cline{2-4} & \dots & P_j & \dots \\ \cline{2-4} & \dots & \dots & \dots \\ r_i & \dots & v_{ij} & \dots \\ & \dots & \dots & \dots \\ \cline{2-4} \end{array}$$

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, flight number, transaction number, FIFA Wold Cup Final 2026.

Triples are universal representations of relations

All relations can be represented in terms of triples:

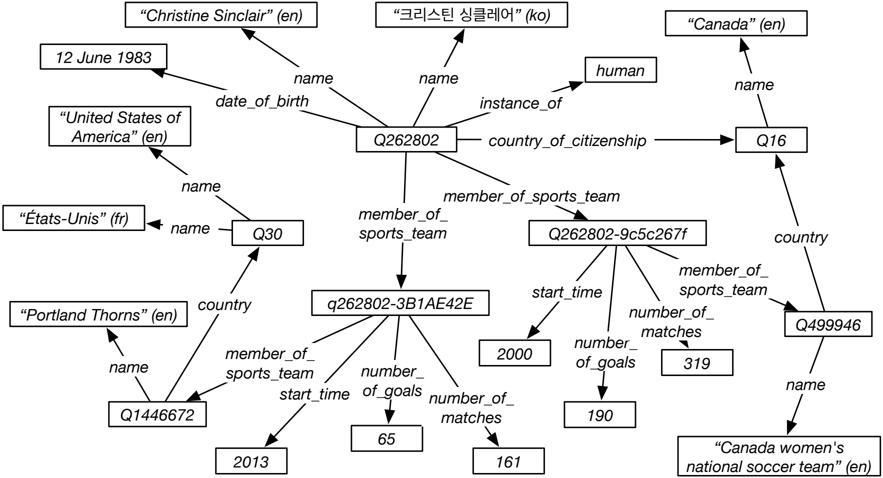
	...	P_j	...
	...	...	...
r_i	...	v_{ij}	...
	...	...	...

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, flight number, transaction number, FIFA Wold Cup Final 2026.

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

Wikidata example: Christine Sinclair



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 based on FreeBase, contains test triples:

- (Jade North,
  /sports/pro_athlete/teams./soccer/football_roster_position/position,
  Defender (association football))
  "*Jade North plays position defender.*"
- (Derby County F.C.,
  /soccer/football_team/current_roster./sports/sports_team_roster/positio
  Defender (association football))
  "*Derby County football club has position defender.*"

But use URIs (meaningless unique name) for Jade North, Derby County F.C., etc

Warning: Many knowledge graphs convert to triples naively

FB15K, a knowledge base commonly used in research papers based on FreeBase, contains test triples:

- (Jade North,
  /sports/pro_athlete/teams./soccer/football_roster_position/position,
  Defender (association football))
  "*Jade North plays position defender.*"
- (Derby County F.C.,
  /soccer/football_team/current_roster./sports/sports_team_roster/positio
  Defender (association football))
  "*Derby County football club has position defender.*"

But use URIs (meaningless unique name) for Jade North, Derby County F.C., etc

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):

  $$Dublin = Ireland + capital\_of$$

  $$Ottawa = Canada + capital\_of$$

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):

  $$Dublin = Ireland + capital\_of$$

  $$Ottawa = Canada + capital\_of$$

- But this entails:

  $$Canada = Ireland - Dublin + Ottawa$$

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):

$$Dublin = Ireland + capital\_of$$

$$Ottawa = Canada + capital\_of$$

- But this entails:

$$Canada = Ireland - Dublin + Ottawa$$

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

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):

$$Dublin = Ireland + capital\_of$$

$$Ottawa = Canada + capital\_of$$

- But this entails:

$$Canada = Ireland - Dublin + Ottawa$$

- Words can have simple meanings but (almost all) entities are multi-faceted and complex.
- Should we use the same sized vector for Canada as Q262802-3B1AE42E (the reified relation between Christine Sinclair and Portland Thorns)?

Outline

Aside: Probabilities $\leftrightarrow$ sigmoid

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

Aside: Probabilities $\leftrightarrow$ sigmoid

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

Aside: Probabilities $\leftrightarrow$ sigmoid

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

Aside: Probabilities $\leftrightarrow$ sigmoid

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

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

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

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

Aside: Probabilities $\leftrightarrow$ sigmoid

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

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

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

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:

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

Vector & Tensor Representations of Entities & Relations

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

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

— 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:

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

  — 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:

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

  — 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)

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

Vector & Tensor Representations of Entities & Relations

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

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

— 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)

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

— 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 $(p123, likes, m53)$ and $(m53, directed\_by, p534)$.

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.

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 $(p123, likes, m53)$ 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.

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

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
- 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.
- ComplеX: 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
  - point-wise $\leq$ corresponds to implication
  - easy to explain what it learns

What/how embedding-based models learn

PD^+:

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

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

What/how embedding-based models learn

PD^+:

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

$E_0[h][f] \geq 0$ $E_2[h][f] \geq 0$.
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^+:

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

$E_0[h][f] \geq 0$ $E_2[h][f] \geq 0$.
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$^+$:

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

$E_0[h][f] \geq 0$ $E_2[h][f] \geq 0$.

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$^+$:

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

$E_0[h][f] \geq 0$ $E_2[h][f] \geq 0$.
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$^+$:

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

$E_0[h][f] \geq 0$ $E_2[h][f] \geq 0$.

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.

Beware of ranking

- Most knowledge graphs only contain positive information.
- How can we evaluate a prediction?

Beware of ranking

- Most knowledge graphs only contain positive information.
- How can we evaluate a prediction? Test cases from FB15K:
  (Jade North, Plays Position, ?)
  (?, Plays Position, Defender)
  (Derby County F.C., Has Position, ?)
  (?, Has Position, Defender)

Beware of ranking

- Most knowledge graphs only contain positive information.
- How can we evaluate a prediction? Test cases from FB15K:
  (Jade North, Plays Position, ?)
  (?, Plays Position, Defender)
  (Derby County F.C., Has Position, ?)
  (?, Has Position, Defender)
- Common to use measures based on ranking such as
  mean reciprocal rank (MRR), Hit@1, Hit@10.
  (And then the sigmoid/softmax can be ignored).

Beware of ranking

- Most knowledge graphs only contain positive information.
- How can we evaluate a prediction? Test cases from FB15K:
  (Jade North, Plays Position, ?)
  (?, Plays Position, Defender)
  (Derby County F.C., Has Position, ?)
  (?, Has Position, Defender)
- Common to use measures based on ranking such as mean reciprocal rank (MRR), Hit@1, Hit@10.
  (And then the sigmoid/softmax can be ignored).
- 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? Test cases from FB15K:
  (Jade North, Plays Position, ?)
  (?, Plays Position, Defender)
  (Derby County F.C., Has Position, ?)
  (?, Has Position, Defender)
- Common to use measures based on ranking such as mean reciprocal rank (MRR), Hit@1, Hit@10.
  (And then the sigmoid/softmax can be ignored).
- Problem #1: is it not good for answers for which there is no answer or many answers:
  Who is the pope married to? Who has streamed Drake's music?

Beware of ranking

- Most knowledge graphs only contain positive information.
- How can we evaluate a prediction? Test cases from FB15K:
  (Jade North, Plays Position, ?)
  (?, Plays Position, Defender)
  (Derby County F.C., Has Position, ?)
  (?, Has Position, Defender)
- Common to use measures based on ranking such as
  mean reciprocal rank (MRR), Hit@1, Hit@10.
  (And then the sigmoid/softmax can be ignored).
- Problem #1: is it not good for answers for which there is no answer or many answers:
  Who is the pope married to? Who has streamed Drake's music?
- Problem #2: an omniscient agent does poorly on ranking scores!

Beware of ranking

- Most knowledge graphs only contain positive information.
- How can we evaluate a prediction? Test cases from FB15K:
  (Jade North, Plays Position, ?)
  (?, Plays Position, Defender)
  (Derby County F.C., Has Position, ?)
  (?, Has Position, Defender)
- Common to use measures based on ranking such as mean reciprocal rank (MRR), Hit@1, Hit@10.
  (And then the sigmoid/softmax can be ignored).
- Problem #1: is it not good for answers for which there is no answer or many answers:
  Who is the pope married to? Who has streamed Drake's music?
- Problem #2: an omniscient agent does poorly on ranking scores!
- Challenge: design a good evaluation scheme. Log-likelihood seems reasonable, but requires knowledge of negations.

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)

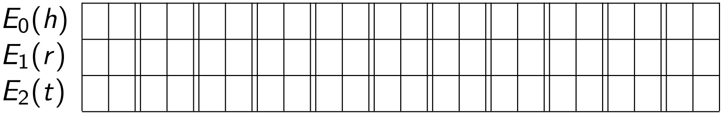
Beyond Triples

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

Beyond Triples

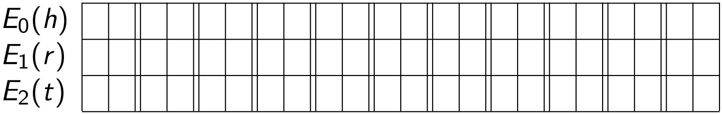
Distmult, Complex, Simple, Simple+... $P((h, r, t))$ is function of



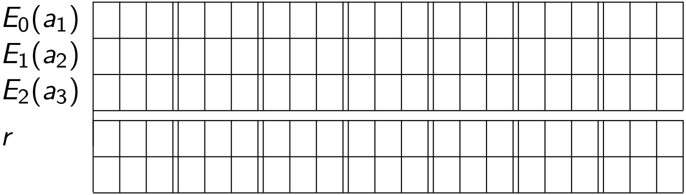
where each window (of size 2) is treated separately and then summed.

Beyond Triples

Distmult, Complex, Simple, Simple+... $P((h, r, t))$ is function of



where each window (of size 2) is treated separately and then summed.
$P(r(a_1, a_2, a_3))$ is function of



window size varies; each window independent

Outline

What can't any of the previous models do?

- The embedding gives a measure of similarity not identity.

What can't any of the previous models do?

- The embedding gives a measure of similarity not identity.
- Consider a flight with stopover:

$$\mathit{flight\_with\_stopover}(A, B) \leftrightarrow \exists C\ \mathit{flight}(A, C) \land \mathit{flight}(C, B)$$

The airport the second flight leaves from must be the same – not just similar – as the airport the first flight arrived at.

Identity

- Example: in the room was
  - Sam's mother
  - Chris's football coach
  - a brilliant computer scientist

  How many people were in the room?

Identity

- Example: in the room was
  - Sam's mother
  - Chris's football coach
  - a brilliant computer scientist

  How many people were in the room?

  Answer: at least one

Identity

- Example: in the room was
  - Sam's mother
  - Chris's football coach
  - a brilliant computer scientist

  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

- Example: in the room was
  - Sam's mother
  - Chris's football coach
  - a brilliant computer scientist

  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.
- Aside: We need knowledge graphs to (be able to) state "there are no more . . ."

Identity $\neq$ similarity

Similar (same make, shape and color); not the same chair:

Not similar, but the same person:

Existence and Identity are tricky

- Is this reference to the same paper as this other reference?
- Was there a burglar here last night?
- Was the burglar the same person as one of the people in the line up?

Existence and Identity are tricky

- Is this reference to the same paper as this other reference?
- Was there a burglar here last night?
- Was the burglar the same person as one of the people in the line up?

- Existence isn't a property of an entity!
  When existence is false, there is no entity.

Existence and Identity are tricky

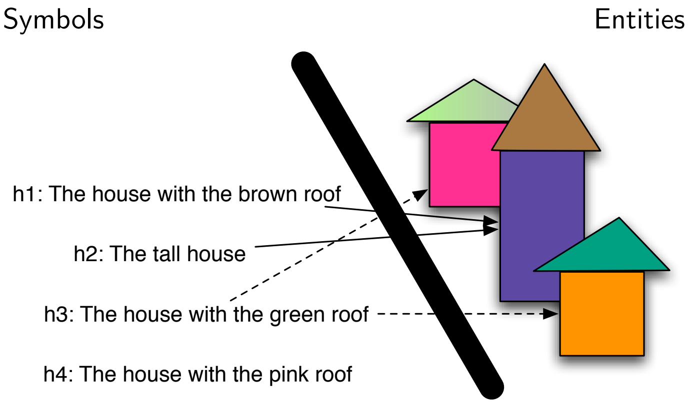
- Is this reference to the same paper as this other reference?
- Was there a burglar here last night?
- Was the burglar the same person as one of the people in the line up?

- Existence isn't a property of an entity!
  When existence is false, there is no entity.
- Two entities cannot be equal (have same identity); otherwise there is only one object.

Existence and Identity are tricky

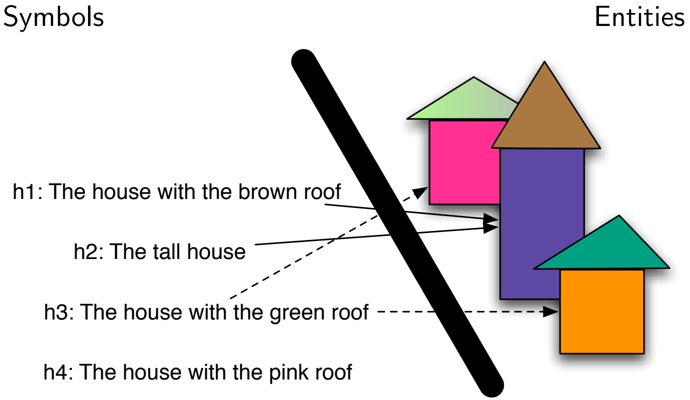
- Is this reference to the same paper as this other reference?
- Was there a burglar here last night?
- Was the burglar the same person as one of the people in the line up?
- Existence isn't a property of an entity!
  When existence is false, there is no entity.
- Two entities cannot be equal (have same identity); otherwise there is only one object.
- Many methods (e.g., graph neural networks, Markov logic networks, probabilistic logic programs . . .) assume that this is already solved: we know what entities exist and what descriptions are equal.

Correspondence Problem



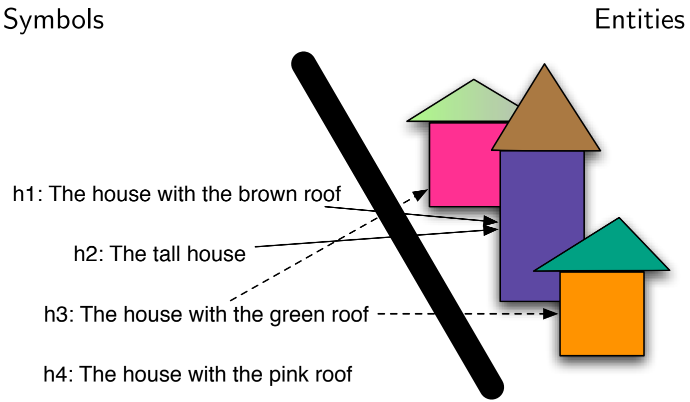
Equality corresponds to partition of the symbols.

Correspondence Problem



Equality corresponds to partition of the symbols.
There are more that exponential (in number of symbols) partitions (Bell number).

Correspondence Problem



Equality corresponds to partition of the symbols.
There are more that exponential (in number of symbols) partitions (Bell number).
Common to use MCMC.

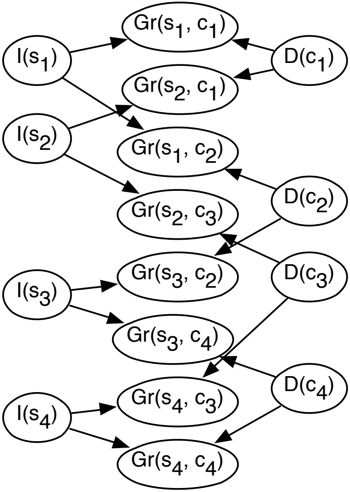
Outline

Example: Predicting Relations

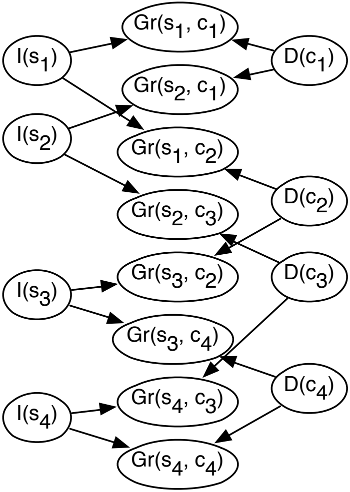
Student	*Course*	*Grade*
s_1	c_1	A
s_2	c_1	C
s_1	c_2	B
s_2	c_3	B
s_3	c_2	B
s_4	c_3	B
s_3	c_4	?
s_4	c_4	?

- Students s_3 and s_4 have the same averages, on courses with the same averages.
- Which student would you expect to better on course c_4?

From Relations to Bayesian Belief Networks



From Relations to Bayesian Belief Networks

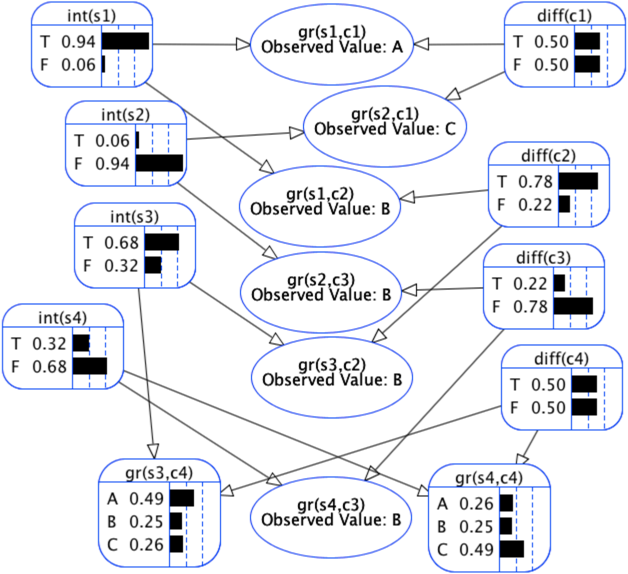


$I(S)$	$D(C)$	$Gr(S,C)$		
		A	B	C
true	*true*	0.5	0.4	0.1
true	*false*	0.9	0.09	0.01
false	*true*	0.01	0.09	0.9
false	*false*	0.1	0.4	0.5

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

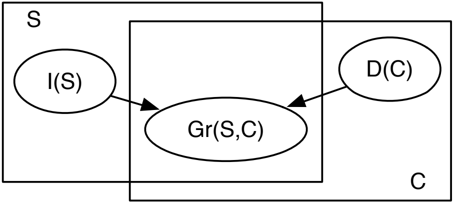
"parameter sharing"

Example: Predicting Relations



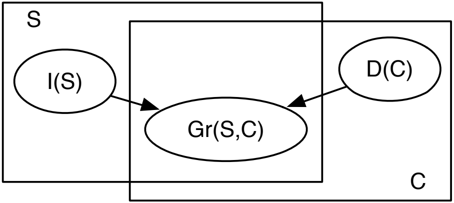
`http://artint.info/code/aispace/grades.xml`

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

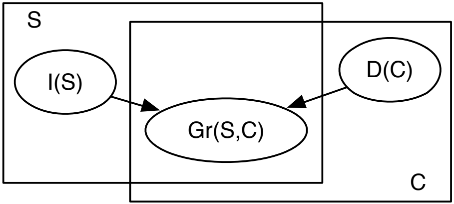


- S, C logical variable representing students, courses
- the set of entities of a type is called a population
- $I(S)$, $Gr(S, C)$, $D(C)$ are parametrized random variables

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

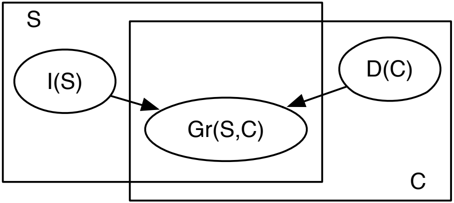


- S, C logical variable representing students, courses
- the set of entities of a type is called a population
- $I(S)$, $Gr(S, C)$, $D(C)$ are parametrized random variables

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

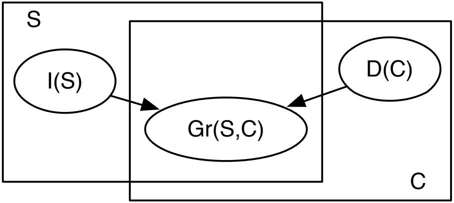


- S, C logical variable representing students, courses
- the set of entities of a type is called a population
- $I(S)$, $Gr(S, C)$, $D(C)$ are parametrized random variables

Grounding:

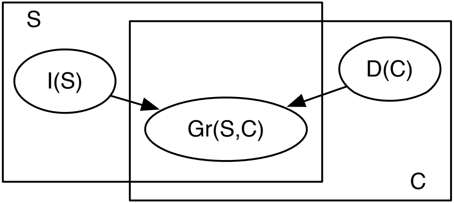
- for every student s, there is a random variable $I(s)$
- for every course c, there is a random variable $D(c)$
- for every s, c pair there is a random variable $Gr(s, c)$
- all instances share the same structure and parameters

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.

Outline

Relational Learning in (Lifted) Graphs

A common framework:

- Nodes are entities (all existing, already disambiguated)
- (Hyper)- edges between entities that are related.
- Function of the dependency depends on the types of the entities. (E.g., whether one person likes another depends on the properties of entities, not their identity.)

Relational Learning in (Lifted) Graphs

A common framework:

- Nodes are entities (all existing, already disambiguated)
- (Hyper)- edges between entities that are related.
- Function of the dependency depends on the types of the entities. (E.g., whether one person likes another depends on the properties of entities, not their identity.)
- Markov logic networks (MLNs), probabilistic logic programs (PLPs) defined by factors with learnable parameters. All properties have a probabilistic interpretation.

Relational Learning in (Lifted) Graphs

A common framework:

- Nodes are entities (all existing, already disambiguated)
- (Hyper)- edges between entities that are related.
- Function of the dependency depends on the types of the entities. (E.g., whether one person likes another depends on the properties of entities, not their identity.)
- Markov logic networks (MLNs), probabilistic logic programs (PLPs) defined by factors with learnable parameters. All properties have a probabilistic interpretation.
- Graph neural networks (GNNs) defined by how properties (embedding) of a node depend in its neighbours using a differentiable function.

Relational Learning in (Lifted) Graphs

A common framework:

- Nodes are entities (all existing, already disambiguated)
- (Hyper)- edges between entities that are related.
- Function of the dependency depends on the types of the entities. (E.g., whether one person likes another depends on the properties of entities, not their identity.)
- Markov logic networks (MLNs), probabilistic logic programs (PLPs) defined by factors with learnable parameters. All properties have a probabilistic interpretation.
- Graph neural networks (GNNs) defined by how properties (embedding) of a node depend in its neighbours using a differentiable function.

— main difference: in MLNs and PLPs, latent variables have a probabilistic interpretation. GNNs choose the function to maximize predictive performance.

Aggregation

- Aggregation is how the neighbours affect the prediction on a node.
- Typically build a model of how neighbours of different types affect the node.
- When there can be an unboundedly many neighbours (of same type), we need to combine them: aggregation

Aggregation

- Aggregation is how the neighbours affect the prediction on a node.
- Typically build a model of how neighbours of different types affect the node.
- When there can be an unboundedly many neighbours (of same type), we need to combine them: aggregation
- Example: predict $gender(P)$ or $age(P)$, the gender or age of person P, from $rating(P, M)$ the rating of person P on movie M.

Aggregation

- Aggregation is how the neighbours affect the prediction on a node.
- Typically build a model of how neighbours of different types affect the node.
- When there can be an unboundedly many neighbours (of same type), we need to combine them: aggregation
- Example: predict $gender(P)$ or $age(P)$, the gender or age of person P, from $rating(P, M)$ the rating of person P on movie M.
    - E.g., Predict age of David from movies he rated.

Aggregation

- Aggregation is how the neighbours affect the prediction on a node.
- Typically build a model of how neighbours of different types affect the node.
- When there can be an unboundedly many neighbours (of same type), we need to combine them: aggregation
- Example: predict $gender(P)$ or $age(P)$, the gender or age of person P, from $rating(P, M)$ the rating of person P on movie M.
  - E.g., Predict age of David from movies he rated.
  - Methods that project to lower dimensional representations don't work, because there isn't one for gender or age.

Aggregation

- Aggregation is how the neighbours affect the prediction on a node.
- Typically build a model of how neighbours of different types affect the node.
- When there can be an unboundedly many neighbours (of same type), we need to combine them: aggregation
- Example: predict $gender(P)$ or $age(P)$, the gender or age of person P, from $rating(P, M)$ the rating of person P on movie M.
  - E.g., Predict age of David from movies he rated.
  - Methods that project to lower dimensional representations don't work, because there isn't one for gender or age.
  - One of the embeddings of each person can just memorize the age — no generalization.

Aggregation

- Aggregation is how the neighbours affect the prediction on a node.
- Typically build a model of how neighbours of different types affect the node.
- When there can be an unboundedly many neighbours (of same type), we need to combine them: aggregation
- Example: predict $gender(P)$ or $age(P)$, the gender or age of person P, from $rating(P, M)$ the rating of person P on movie M.
  - E.g., Predict age of David from movies he rated.
  - Methods that project to lower dimensional representations don't work, because there isn't one for gender or age.
  - One of the embeddings of each person can just memorize the age — no generalization.
- Requires aggregation: some models provide built-in aggregation, and some you can use whatever aggregation you want.

Representations of Lifted Graphical models

Common representations:

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

Representations of Lifted Graphical models

Common representations:

Aggregator	Undirected	Directed
Weighted Formulas	Markov Logic Networks (MLNs)	Relational Logistic Regression (RLR)
Existential quantification = noisy-or		Probabilistic logic programs

Representations of Lifted Graphical models

Common representations:

Aggregator	Undirected	Directed
Weighted Formulas	Markov Logic Networks (MLNs)	Relational Logistic Regression (RLR)
Existential quantification = noisy-or		Probabilistic logic programs
explicit: typically sum, average, or max	Graph Neural Networks	

Aggregation

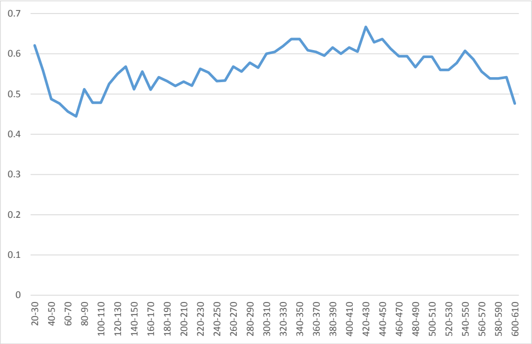
- For average and max aggregation , more data for a particular individual doesn't give better predictions.
  E.g., one movie rated vs 1000 rated for a person.

Aggregation

- For average and max aggregation , more data for a particular individual doesn't give better predictions.
  E.g., one movie rated vs 1000 rated for a person.
- for sum aggregation in GNNs, and aggregation in Markov Logic networks and probabilistic logic programs, either
  - the prediction does not depend on neighbours, or
  - the prediction goes to 0 or 1 as the number of neighbours goes to infinity, or
  - the numbers cancel out in an unstable way

  for some models, this occurs even if there are no observations

Real Data



number of movies rated

Observed $P(25 < Age(u) < 45 \mid n)$, where n is number of movies watched from the Movielens 100k dataset.

Danger of fitting to data without understanding the model

- Beware of models that fit polynomials of degree greater than 1.

Danger of fitting to data without understanding the model

- Beware of models that fit polynomials of degree greater than 1.
- 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 $n \to \infty$?

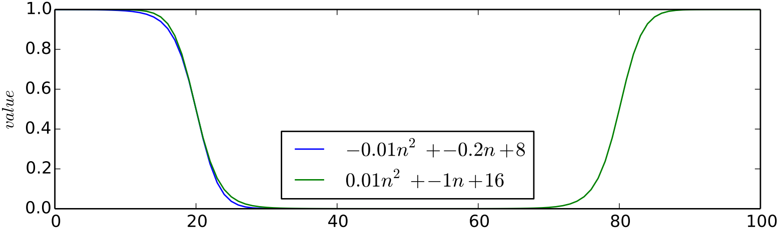
Danger of fitting to data without understanding the model

- Beware of models that fit polynomials of degree greater than 1.
- 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

Missing data cannot be ignored

Example: there is a drug that only causes sick people to get sicker, and drop out of study.

Missing data cannot be ignored

Example: there is a drug that only causes sick people to get sicker, and drop out of study.

- If people who drop out are ignored, it looks like the drug works: A larger proportion of those who took the drug are well.

Missing data cannot be ignored

Example: there is a drug that only causes sick people to get sicker, and drop out of study.

- If people who drop out are ignored, it looks like the drug works: A larger proportion of those who took the drug are well.
- We need to go beyond the non-missing data to determine why data is missing.

Missing data cannot be ignored

Example: there is a drug that only causes sick people to get sicker, and drop out of study.

- If people who drop out are ignored, it looks like the drug works: A larger proportion of those who took the drug are well.
- We need to go beyond the non-missing data to determine why data is missing.
- EM (and other methods) work, but produce nonsense!
- Almost all data in relational models is missing, but missingness is usually ignored

Can't Big Data and Deep Networks Just Solve This

- Solution to poor prediction in ML models:

Can't Big Data and Deep Networks Just Solve This

- Solution to poor prediction in ML models: collect more data.

Can't Big Data and Deep Networks Just Solve This

- Solution to poor prediction in ML models: collect more data.
- While knowledge graphs and relational databases might be huge, there is often very little data about individual entities.

Can't Big Data and Deep Networks Just Solve This

- Solution to poor prediction in ML models: collect more data.
- While knowledge graphs and relational databases might be huge, there is often very little data about individual entities.
- SNOMED CT is a medical ontology of clinical terms with about 100,000 concepts of what could be called diseases.

Can't Big Data and Deep Networks Just Solve This

- Solution to poor prediction in ML models: collect more data.
- While knowledge graphs and relational databases might be huge, there is often very little data about individual entities.
- SNOMED CT is a medical ontology of clinical terms with about 100,000 concepts of what could be called diseases.
  For almost every pair of symptoms, pair of diseases, *no one* in the world has both!

Can't Big Data and Deep Networks Just Solve This

- Solution to poor prediction in ML models: collect more data.
- While knowledge graphs and relational databases might be huge, there is often very little data about individual entities.
- SNOMED CT is a medical ontology of clinical terms with about 100,000 concepts of what could be called diseases.
  For almost every pair of symptoms, pair of diseases, *no one* in the world has both!
- Finding minerals to build batteries will help fight climate change.

Can't Big Data and Deep Networks Just Solve This

- Solution to poor prediction in ML models: collect more data.
- While knowledge graphs and relational databases might be huge, there is often very little data about individual entities.
- SNOMED CT is a medical ontology of clinical terms with about 100,000 concepts of what could be called diseases.
  For almost every pair of symptoms, pair of diseases, *no one* in the world has both!
- Finding minerals to build batteries will help fight climate change.
  For all mined commodities combined, there are fewer than 10,000 known high-grade deposits. These are complex. Undiscovered deposits are (probably) unlike the known ones.

Can't Big Data and Deep Networks Just Solve This

- Solution to poor prediction in ML models: collect more data.
- While knowledge graphs and relational databases might be huge, there is often very little data about individual entities.
- SNOMED CT is a medical ontology of clinical terms with about 100,000 concepts of what could be called diseases.
  For almost every pair of symptoms, pair of diseases, *no one* in the world has both!
- Finding minerals to build batteries will help fight climate change.
  For all mined commodities combined, there are fewer than 10,000 known high-grade deposits. These are complex. Undiscovered deposits are (probably) unlike the known ones.
- Reified entities have very few facts about them.
- There is a long tail of entities about which we know very little

Outline

1. What are relational probabilistic models and relational learning?
   - Relational Models
   - Knowledge Graphs
2. Learning Knowledge Graphs
3. Existence and Identity Uncertainty
4. Lifted Graphical Models
5. Graph-based models
6. Missing data
7. Big Data
8. Bayesian $\Rightarrow$ Exchangeability $\Rightarrow$ Lifted Inference
9. Conclusion

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 Van den Broeck, Kersting, Natarajan and Poole (Eds) *An Introduction to Lifted Inference*, MIT Press, 2021.

Outline

Conclusion / Challenges

- We need to try to avoid overfitting to standard datasets.

Conclusion / Challenges

- We need to try to avoid overfitting to standard datasets.
- We need better evaluation metrics
  (e.g., one that an omniscient agent would do well on)

Conclusion / Challenges

- We need to try to avoid overfitting to standard datasets.
- We need better evaluation metrics
  (e.g., one that an omniscient agent would do well on)
- We need better ways to incorporate prior knowledge.
  E.g., Wikidata has (almost) complete information about celebrities, but the people in Wikidata is not a random sample.

Conclusion / Challenges

- We need to try to avoid overfitting to standard datasets.
- We need better evaluation metrics
  (e.g., one that an omniscient agent would do well on)
- We need better ways to incorporate prior knowledge.
  E.g., Wikidata has (almost) complete information about celebrities, but the people in Wikidata is not a random sample.
- Predicting properties of entities and relationships among entities are (currently) different problems. Methods that work for one often don't work for other, but authors are often not explicit about what they do.

Conclusion / Challenges

- We need to try to avoid overfitting to standard datasets.
- We need better evaluation metrics
  (e.g., one that an omniscient agent would do well on)
- We need better ways to incorporate prior knowledge.
  E.g., Wikidata has (almost) complete information about celebrities, but the people in Wikidata is not a random sample.
- Predicting properties of entities and relationships among entities are (currently) different problems. Methods that work for one often don't work for other, but authors are often not explicit about what they do.
- We need better models of aggregation.

Conclusion / Challenges

- We need to try to avoid overfitting to standard datasets.
- We need better evaluation metrics
  (e.g., one that an omniscient agent would do well on)
- We need better ways to incorporate prior knowledge.
  E.g., Wikidata has (almost) complete information about celebrities, but the people in Wikidata is not a random sample.
- Predicting properties of entities and relationships among entities are (currently) different problems. Methods that work for one often don't work for other, but authors are often not explicit about what they do.
- We need better models of aggregation.
- We need to take existence and identity (equality) of entities more seriously in general learning models.

Conclusion / Challenges

- We need to try to avoid overfitting to standard datasets.
- We need better evaluation metrics
  (e.g., one that an omniscient agent would do well on)
- We need better ways to incorporate prior knowledge.
  E.g., Wikidata has (almost) complete information about celebrities, but the people in Wikidata is not a random sample.
- Predicting properties of entities and relationships among entities are (currently) different problems. Methods that work for one often don't work for other, but authors are often not explicit about what they do.
- We need better models of aggregation.
- We need to take existence and identity (equality) of entities more seriously in general learning models.
- We need to model missing data (causal problem)

Conclusion / Challenges

- We need to try to avoid overfitting to standard datasets.
- We need better evaluation metrics
  (e.g., one that an omniscient agent would do well on)
- We need better ways to incorporate prior knowledge.
  E.g., Wikidata has (almost) complete information about celebrities, but the people in Wikidata is not a random sample.
- Predicting properties of entities and relationships among entities are (currently) different problems. Methods that work for one often don't work for other, but authors are often not explicit about what they do.
- We need better models of aggregation.
- We need to take existence and identity (equality) of entities more seriously in general learning models.
- We need to model missing data (causal problem)
- Lots of (environmental) data coming, but need to deal with time, ontologies, causality, big data (but small data about almost everything) ...

Conclusion / Challenges

- We need to try to avoid overfitting to standard datasets.
- We need better evaluation metrics
  (e.g., one that an omniscient agent would do well on)
- We need better ways to incorporate prior knowledge.
  E.g., Wikidata has (almost) complete information about celebrities, but the people in Wikidata is not a random sample.
- Predicting properties of entities and relationships among entities are (currently) different problems. Methods that work for one often don't work for other, but authors are often not explicit about what they do.
- We need better models of aggregation.
- We need to take existence and identity (equality) of entities more seriously in general learning models.
- We need to model missing data (causal problem)
- Lots of (environmental) data coming, but need to deal with time, ontologies, causality, big data (but small data about almost everything) . . .
- 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)