# Lecture 17.4: Relational Representations

- Probabilistic Logic Programs
- Weighted Logical Formulae
- Graph Neural Networks
- Existence and Identity Uncertainty

Probabilistic Logic Programs

- the model is described in terms of a logic program with parametrized independent noise variables.
- Plates correspond to logical variables.
- Parametrized random variables are represented as logical atoms,
- A Turing-complete language for relational probabilistic models.
- Extends Datalog / logic programs to include probabilities.
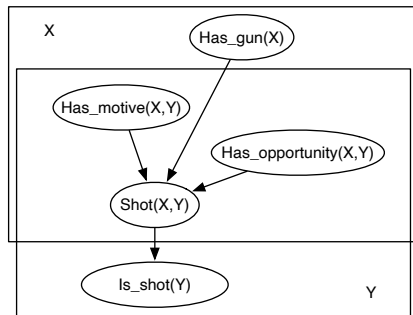
Example: Probabilistic Logic Programs
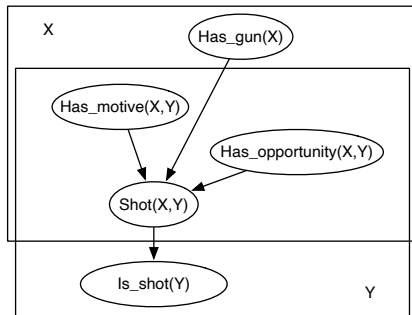
$$is\_shot(Y) \leftarrow shot\_by\_no\_one(Y)$$
$$is\_shot(Y) \leftarrow shot(X, Y) \wedge shot\_succeeds(X, Y)$$

Each ground instance of $shot\_by\_no\_one(Y)$ and
$shot\_succeeds(X, Y)$ are independent noise variables.

Example: Probabilistic Logic Programs

$shot(X, Y) \leftarrow has\_motive(X, Y) \wedge has\_gun(X)$
$\wedge\ has\_opportunity(X, Y) \wedge actually\_shot(X, Y).$

$P(actually\_shot(X, Y))$ is the probability that X would shoot Y if they had a motive, gun, and opportunity.

- Other rules could cover other cases, such as where X doesn't have a motive.

Weighted Logical Formulae

- A weighted logical formula is a pair

  $$(formula, weight)$$

- A world is an assignment of a true value to each ground atom
- In Markov logic networks (MLNs), the measure of a world is proportional to the exponential of the sum of the weights of the formulae true in the world.
- A conditional probability, $P(x \mid obs)$ is the measure of the worlds in which x is true out of the worlds in which obs is true.
- In relational logistic regression, the weighted formulae are used to define conditional probabilities.

Weighted First-Order Logic Formulas
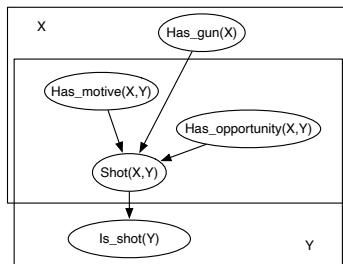
MLNs provide an undirected model, e.g.,

$(is\_shot(Y), w_0)$

$(is\_shot(Y) \vee \neg shot(X, Y), w_1)$

$(shot(X, Y) \vee \neg has\_motive(X, Y) \vee \neg has\_gun(X)$
$\quad \vee \neg has\_opportunity(X, Y) \vee \neg actually\_shot(X, Y), w_2)$

$P(is\_shot(v) \mid shot(p_1, v), \ldots shot(p_n, v))$ is logistic regression if $p_1, \ldots, p_n$ are all the individuals.

Graph Neural Networks

- Graph neural networks are neural networks that act on graph data.

- Each node has an embedding that is inferred from parametrized linear functions and activation functions of the node's neighbors, and their neighbors, to some depth.

- A relational graph convolutional network (R-GCN) is used to learn embeddings for knowledge graphs, where nodes are entities and arcs are labelled with relations.

Relational Graph Convolutional Networks

- Nodes are entities and arcs are labelled with relations.
- The properties for entity e give an embedding $h_e^{(0)}$ (using a standard neural network)
- These embeddings provide inputs to the neural network.
- There are multiple layers, with shared parameters, that follow the graph structure to give an output embedding for each entity.
- Called *convolutional* because the same learnable parameters are used for each entity.

Relational Graph Convolutional Networks

- $h_e^{(0)}$ is the embedding based on the properties of entity e
- The embedding of layer $L+1$ for entity e is the vector $h_e^{(L+1)}$:

$$h_e^{(L+1)} = \phi \left(W_0^{(L)} h_e^{(L)} + \sum_{r \in R} \sum_{\{n : (e,r,n) \in KG\}} \frac{1}{C_{e,r}} W_r^{(L)} h_n^{(L)} \right)$$

where

- ϕ is an activation function (e.g., ReLU)
- $h_e^{(L)}$ is the embedding for entity e from layer L.
- $W_0^{(L)}$ is a matrix defining how the embedding for entity e in layer L affects the same entity in the next layer.
- R is the set of all relations
- KG is the set of triples in the knowledge graph.
- $W_r^{(L)}$ is a matrix for relation r for layer L, which is multiplied by the vector $h_n^{(L)}$ for each neighbor n.
- $C_{e,r}$ is a normalization constant, such as $|\{n : (e,r,n) \in KG\}|$, which gives an average for each relation.

Relational Graph Convolutional Networks

- This model uses separate parameters for each relation
  - ▶ $\longrightarrow$ overfitting for relations with few instances.
  - ▶ $W_r^{(L)}$, can be represented as a linear combination of a few learned basis matrices which are shared among the relations
  - ▶ the weights in the linear combination depend on the relation.
- Question: Is summing or avaraging an appropriate way to aggregate the embeddings of related entities? Would something else be more appropriate?

Existence and Identity Uncertainty

- Previous models are for relational uncertainty; uncertainty about whether a relation is true of some entities. Assumed the set of entities is known.
- Identity uncertainty concerns uncertainty of whether two symbols denote the same entity – the symbols are equal
- Existence uncertainty concerns uncertainty as to whether there exists an entity that fits a description
- Number uncertainty concerns uncertainty as how many entities fit a description

Existence and Identity Example
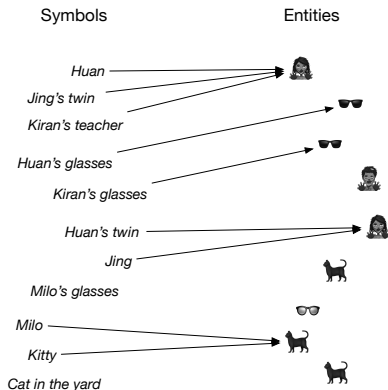
- *Huan* = *Jing's twin*, because they denote the same entity.
- *Huan's glasses* ≠ *Kiran's glasses*, because they denote different pairs of glasses
- *Milo's glasses* do not exist; Milo doesn't have a pair of glasses.

Identity Uncertainty

- Identity Uncertainty is a problem for medical systems: is this patient the same one who visited yesterday asking for drugs?

- Called record linkage, as the problem is to determine which (medical) records are for the same person.

- Citation matching is determining if authors on different papers are the same person, or if two citations denote the same paper. (What is "the same paper"? Is an earlier arxiv.org paper the same paper?)

- Identity uncertainty implies partitioning the symbols.

- The number of partitions of n items is the Bell number, which grows faster than any exponential in n.

- Use Markov-chain Monte Carlo (MCMC): given a partition, entities can be moved to different partitions or to new partitions.

Existence Uncertainty

- Given a description, there might be no entities who fit a description or there may be multiple entities.

- E.g., for "the cat in the yard", there may be no cats in the yard or there might be multiple cats.

- Existence is not a property of entities. Entities that do not exist do not have properties.

- Tricky when there are complex roles involved to determine whether there are entities to fill the roles.

- E.g., finding an apartment for Sam and Sam's child Chris. Preference depend on Sam's bedroom and Chris's bedroom. Apartments don't come labelled with whose bedroom it is.