CITS3005 Knowledge Representation

Lecture 5: Uncertainty

Tim French

The University of Western Australia

2023

Overview

We will then consider the types and sources of knowledge, and robust reasoning methodologies:

- Modalities of information
- Types of Uncertainty
- Unstructured Information
- Knowledge Extraction
- Reasoning about Uncertainty

Types of knowledge

There are a number of ways we could classify knowledge:

Certain

- Vocabulary
- Taxonomy
- Facts
- Rules

Uncertain

- Causality/Correlation
- Fuzzy/Vague Rules
- Probabilities
- Frequencies
- Possibilities

Domain

- Temporal
- Spatial
- Deontic
- Agency
- Knowledge
- Numeric
- Strategy

Sources of Knowledge

We can also classify knowledge by its source, or how it is stored an accessed:

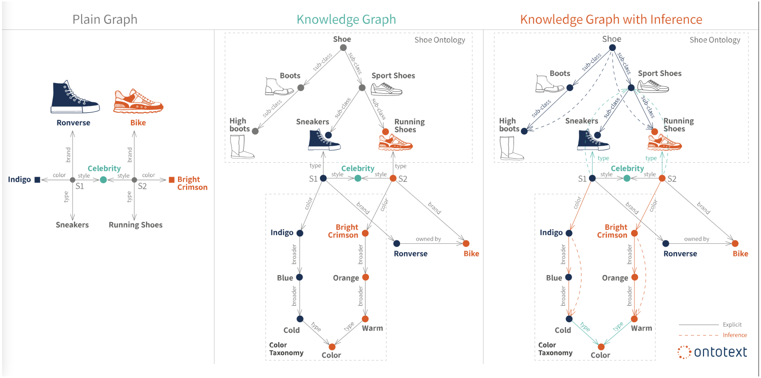
- Intrinsic/A priori: is there such a thing as *pure common knowledge*.
- Formal symbolic representations: like a logic.
- Source files: structured, machine readable information, with a formal grammar and semantics.
- Database records: or spreadsheets, CSVs, JSON and XML?
- Data streams and sensor readings: signals that are perceived.
- Unstructured text: like witness accounts, or comments in a file.
- images or audio or video.

Each of these, implicitly or explicitly, has some notion of information content and semantics.
Note, while all of these have (implicit) semantics, most do not describe the semantics. The semantics are often conveyed (not represented) in a knowledge graph.

Knowledge Graphs

- **Google Knowledge Graph**. Google made this term popular with the announcement of its knowledge graph in 2012. However, there are very few technical details about its organisation, coverage and size.
- **DBPedia**. This project leverages the structure inherent in the infoboxes of Wikipedia to create an enormous dataset of 38.3 million things and an ontology that has encyclopedic coverage of entities such as people, places, films, books, organisations, species, diseases, etc.
- **Geonames**. Under a creative commons, users of Geonames dataset have access to 25 million geographical entities and features.
- **Wordnet**. One of the most well-known lexical databases for the English language, providing definitions and synonyms.
- **YAGO**. YAGO contains both entities (such as movies, people, cities, countries, etc.) and relations between these entities (who played in which movie, which city is located in which country, etc.). YAGO contains more than 50 million entities and 2 billion facts.

Graphs vs Knowledge Graphs



Knowledge Extraction

We should consider the process of transforming the given source into *knowledge*.
Given a signal (perhaps a video, or a book) and some query (how to replace a sim card, or what is a *fluent*) we need a process to transform the raw data into knowledge.
Considering such a process at a very high level this requires that we have:

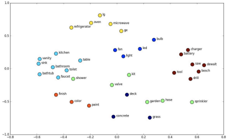
- an ability to parse the source into *features*.
- some implicit *semantics* regarding what these features mean.
- some target *ontology* containing a representation of the query.
- a *mapping* from our features to the target ontology.
- some computational *process* for inferring an answer to the query from the populated ontology.

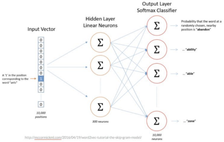
If we constrain ourselves to a world where knowledge sources are knowledge bases, queries are written in first order logic, and we have a sufficiently powerful reasoner, then all these elements are given.
... however the real world is rarely so clean.

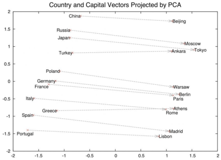
Word Embeddings

An example of real world knowledge extraction comes from unstructured text.

- *Word embeddings* are representations of words as vectors (points in a high dimensional Euclidean space).
- These embeddings optimised to preserve a word's context (surrounding words in a document), by deep neural networks.
- When trained on a sufficiently large document corpus, the topology of the space corresponds to semantic features.
- These embeddings can be powerful tools for machine translation, computing ontology mappings etc. but require a large corpus of relevant documents to be effective.

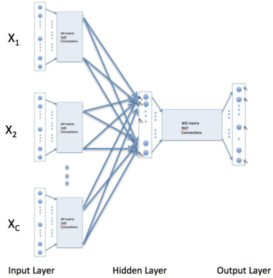






Word2Vec

Word2Vec is one of the early and very successful word embedding models (trained on a large part of the internet and open sourced in 2013). It combines multiple embedding methods together, and there are now several well-known and useful embedding methods available.



Distributed Representations of Words and Phrases and their Compositionality

Tomas Mikolov
Google Inc.
Mountain View
mikolov@google.com

Ilya Sutskever
Google Inc.
Mountain View
ilyasu@google.com

Kai Chen
Google Inc.
Mountain View
kai@google.com

Greg Corrado
Google Inc.
Mountain View
gcorrado@google.com

Jeffrey Dean
Google Inc.
Mountain View
jeff@google.com

Original Word2Vec paper (2013) by the Google team. Word2Vec is an efficient training algorithm for effective word embeddings, which advanced the field considerably.

You can interact with word2vec via the application *semantle.com*

Semantle

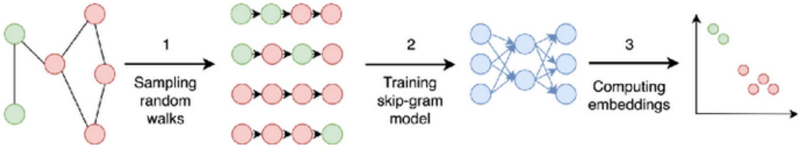
Today is puzzle number **183**. The nearest word has a similarity of **71.86**, the tenth-nearest has a similarity of 62.41 and the one thousandth nearest word has a similarity of 38.38.

Guess | Guess

#	Guess	Similarity	Getting close?
29	playful	42.51	558/1000
16	stylish	71.34	998/1000
17	graceful	65.50	994/1000
18	classy	61.91	989/1000
19	gorgeous	60.07	988/1000
20	chic	59.89	987/1000
21	beautiful	58.71	986/1000
22	lovely	57.35	983/1000
23	charming	53.56	960/1000
24	fanciness	48.96	897/1000
25	classily	46.79	828/1000
26	beguilingly	43.75	664/1000
27	satisfyingly	43.32	633/1000
28	flowery	42.72	578/1000
15	mussy	41.13	425/1000
14	creamy	41.12	422/1000

Graph Embeddings

Just as word embeddings use the context of surrounding words, to infer a semantic representation of a word, a *graph embedding* uses the context of adjacent nodes to infer a semantic representation of a *concept*.
This allows a machine to infer meaning from semi-structured data.



Phases of DeepWalk approach

However we are no longer working with carefully chosen and vetted facts: The outputs of these models are weights produced by an optimisation algorithm, with unclear providence.

Confidence and Uncertainty

The type of "knowledge" produced by machine learning models must be used cautiously. We must consider:

- What data was the model trained on?
- What kind of biases exist in the data?
- How accurate the model is on the available test data?
- How confident are we in the output of the model?
- How definitive is the output of the model?
- Are there multiple plausible interpretations?

Many concepts we reason about have a degree of uncertainty: traffic, weather, shopping destinations, friends and colleagues intents,... and not all of it comes from machine learning scenarios.
Often we are required to make a guess, having seen no reasonable training data before.

Reasoning About Uncertainty

There are a variety of types of uncertainty:

- **Vagueness**: Many concepts are inherently vague. Someone may be *tall* or *clever*, or *absent minded* with us needing to define or measure these concepts.
- **Possibility**: Sometimes we can do no more than decide whether something is *possible* or not: "I can't have left my keys at work since I drove home" etc.
- **Beliefs**: What do we assume to be true, and what do we do when our assumptions turn out top be false.
- **Epistemic Probability**: Would you bet that a concept is true? How much would you bet, and what odds would you accept? Epistemic or Bayesian probability assumes a rational self-interest, and an optimal use of evidence.
- **Frequentist Probability**: How often would we see this concept, if this current scenario (as we understand it) were played out many times over.

Fuzzy Logic

Fuzzy logic is the logic of vagueness, and defines a logic that may be applied to concepts that aren't wholly true or false (e.g. the man is old or the man is bald).

These linguistic variables may be assigned a *fuzzy truth value* between 0 and 1 (inclusive) indicating how true the concept is. (Tim is old (0.7), Tim is bald (0.3)).

Syntax

Fuzzy logics use the same syntax as propositional logic:

$$\alpha ::= \; X \mid \neg\alpha \mid \alpha \wedge \alpha$$

and can be include predicates and quantifiers as well.

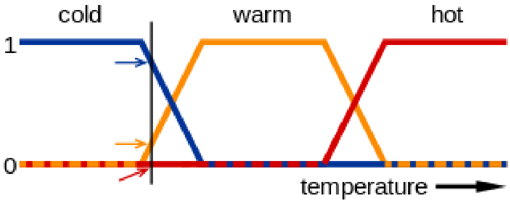
An *interpretation* of fuzzy logic assigns a value $\mathcal{I}(X) \in [0,1]$ to each atomic concept X, and then the semantics follow from:

$$\begin{array}{rcl}
\mathcal{I}(\neg\alpha) & = & 1 - \mathcal{I}(\alpha) \\
\mathcal{I}(\alpha \wedge \beta) & = & \min\{\mathcal{I}(\alpha), \mathcal{I}(\beta)\}, \text{or} \\
\mathcal{I}(\alpha \wedge \beta) & = & \mathcal{I}(\alpha) \times \mathcal{I}(\beta), \text{or} \\
\mathcal{I}(\alpha \wedge \beta) & = & \min\{\mathcal{I}(\alpha) + \mathcal{I}(\beta) - 1, 0\}
\end{array}$$

Applying Fuzzy Logic

Fuzzy logic is applied in several ways:

- Fuzzy logic is employed in automation and control (cruise control, thermostats, inverted pendulums). This involves *fuzzy policy* and a *defuzzification process* to associate an action to a fuzzy set.
- Fuzzy logic is a natural way to interpret the output of neural networks and machine learning models.
- Fuzzy logic is very useful as an interface for *explainable AI* and expert systems.



A challenge with fuzzy logic is assigning values to fuzzy variables, in a robust and defendable way.

Possibility, Plausibility and Expectation

When confronted with a lack of clear knowledge, an agent has to consider what is *possible*, how *plausible* it is, and what the agent *expects* to be true.

- Possibility is similar to consistency, and any concept without an explicit refutation is considered possible.
- Plausibility is a measure of belief that is applied to what is possible. It allows the possible to be compared: "it is possible that it is snowing, but ash from a fire is the more plausible explanation".
- Expectation is the quantification of possibility: a one in a million chance, or a 50-50 call.

Possibility and Necessity

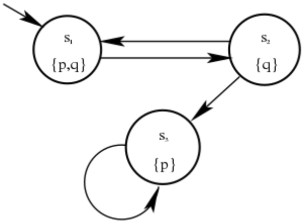
Without any measure of certainty we are simply left with what is possible or not *given our current evidence*.

Epistemic logic introduces a modality $K_i\alpha$ to mean "agent *i knows* α", (α is necessary) and $\neg K_i \neg\alpha$ is interpreted as "agent *i* accepts α is possible".

Reasoning in such a scenario can be done by enumerating a *possible* interpretations and systemically checking each one.

There are systems for deductive reasoning via axioms:

$$K_i(\alpha \to \beta) \to K_i\alpha \to K_i\beta$$



Belief

The theory of belief functions, or Dempster–Shafer theory (DST), is a general framework for reasoning with uncertainty. Belief functions base degrees of belief (or confidence, or trust) for one question on the subjective probabilities for a related question. The basic elements are:

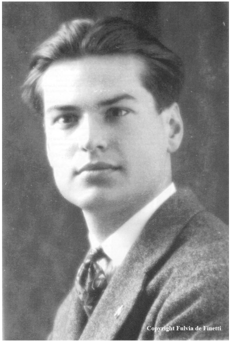
- A frame of reference, consisting of all the concepts under consideration.
- Subjective probabilities (*masses*) are applied to all subsets of concepts.
- *Belief* in set of concepts is the sum of masses of subsets of concepts.
- *Plausibility* is the dual of belief (1 subtract the belief in the negation).

Belief and Plausibility give a lower and upper bound for the likelihood of a concept.

Hypothesis	Mass	Belief	Plausibility
Null (neither alive nor dead)	0	0	0
Alive	0.2	0.2	0.5
Dead	0.5	0.5	0.8
Either (alive or dead)	0.3	1.0	1.0

Epistemic Probabilities

Epistemic Probabilities is a numerical representation of the *likelihood* a concept holds.
It assumes a rational agent will use experience to predict how likely it is a concept will be true.
To quantify this expectation, we can ask what odds would an agent be willing to accept that the concept is true.
For example, accepting odds 2:1 or greater that fair coin lands heads is rational, so we assign a probability of 2:1.
This is the *Dutch Book* argument and makes sense when a hypothesis cannot be repeatedly tested.



Expectation

The epistemic probabilities may be updated in response to new evidence by applying Bayes rule over conditional probabilities.

$$\Pr(A|B) = \frac{\Pr(B|A) \times \Pr(A)}{\Pr(B)}$$

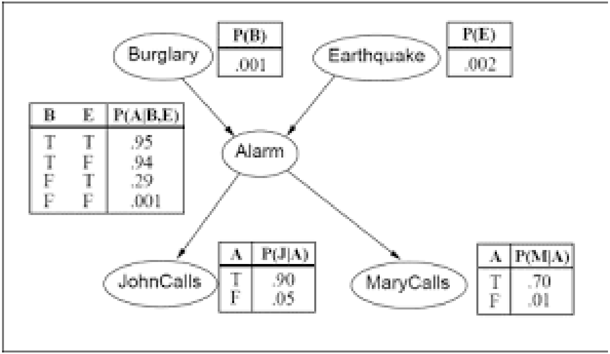
A new observation (B) can be used to update our *prior* probability for A, to get a *posterior* probability for A. The relies on having estimates for the joint probabilities of the events A and B.

Evidence can accumulate and the probability estimate becomes increasingly accurate.

Frequentist probability considers likelihood to be a count of how of how many times an event is observed in a series of random trials. It is easier to calculate, but harder to justify.

Bayesian Networks

Bayesian networks are an efficient method of representing and approximating the joint probability distribution of a set of events.



Frameworks

There are numerous probabilistic reasoning frameworks, including BLOG (Bayesian Logic) and Pyro (PyTorch plus logic programming)

Bayesian Logic (BLOG) is a probabilistic modelling language. It is designed for representing relations and uncertainties among real world objects.

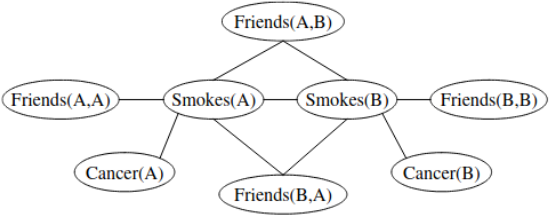
Pyro is a universal probabilistic programming language (PPL) written in Python and supported by PyTorch on the backend.

```
random Boolean Burglary ~ BooleanDistrib(0.001);

random Boolean Earthquake ~ BooleanDistrib(0.002);

random Boolean Alarm ~
  if Burglary then
    if Earthquake then BooleanDistrib(0.95)
    else  BooleanDistrib(0.94)
  else
    if Earthquake then BooleanDistrib(0.29)
    else BooleanDistrib(0.001);

random Boolean JohnCalls ~
  if Alarm then BooleanDistrib(0.9)
  else BooleanDistrib(0.05);

random Boolean MaryCalls ~
  if Alarm then BooleanDistrib(0.7)
  else BooleanDistrib(0.01);

/* Evidence for the burglary model saying that both
 * John and Mary called.  Given this evidence, the posterior probability
 * of Burglary is 0.284 (see p. 505 of "AI: A Modern Approach", 2nd ed.).
 */

obs JohnCalls = true;
obs MaryCalls = true;

/* Query for the burglary model asking whether Burglary
 * is true.
 */

query Burglary;
```

Markov Logic Networks

A Markov logic network (MLN) is a probabilistic logic which applies the ideas of a Markov network to first-order logic, enabling uncertain inference. Markov logic networks generalise first-order logic, in the sense that, in a certain limit, all unsatisfiable statements have a probability of zero, and all tautologies have probability one.

English	First-Order Logic	Clausal Form	Weight
Friends of friends are friends.	$\forall x \forall y \forall z\ \mathtt{Fr}(x,y) \wedge \mathtt{Fr}(y,z) \Rightarrow \mathtt{Fr}(x,z)$	$\neg\mathtt{Fr}(x,y) \vee \neg\mathtt{Fr}(y,z) \vee \mathtt{Fr}(x,z)$	0.7
Friendless people smoke.	$\forall x\ (\neg(\exists y\ \mathtt{Fr}(x,y)) \Rightarrow \mathtt{Sm}(x))$	$\mathtt{Fr}(x, g(x)) \vee \mathtt{Sm}(x)$	2.3
Smoking causes cancer.	$\forall x\ \mathtt{Sm}(x) \Rightarrow \mathtt{Ca}(x)$	$\neg\mathtt{Sm}(x) \vee \mathtt{Ca}(x)$	1.5
If two people are friends, either	$\forall x \forall y\ \mathtt{Fr}(x,y) \Rightarrow (\mathtt{Sm}(x) \Leftrightarrow \mathtt{Sm}(y))$	$\neg\mathtt{Fr}(x,y) \vee \mathtt{Sm}(x) \vee \neg\mathtt{Sm}(y)$,	1.1
both smoke or neither does.		$\neg\mathtt{Fr}(x,y) \vee \neg\mathtt{Sm}(x) \vee \mathtt{Sm}(y)$	1.1



ProbLog

Probabilistic logic programs are logic programs in which some of the facts are annotated with probabilities. ProbLog is a tool that allows you to intuitively build programs that do not only encode complex interactions between a large sets of heterogeneous components but also the inherent uncertainties that are present in real-life situations.

```
0.3::stress(X) :- person(X).
0.2::influences(X,Y) :- person(X), person(Y).

smokes(X) :- stress(X).
smokes(X) :- friend(X,Y), influences(Y,X), smokes(Y).

0.4::asthma(X) :- smokes(X).

person(angelika).
person(joris).
person(jonas).
person(dimitar).

friend(joris,jonas).
friend(joris,angelika).
friend(joris,dimitar).
friend(angelika,jonas).
```

We will look into ProbLog in more detail next lecture.