M3.L3: Semantics and Ontologies

Luke Slater (l.slater.1@bham.ac.uk)

27.11.2019

University of Birmingham - CCB - MSc Bioinformatics - Module 3

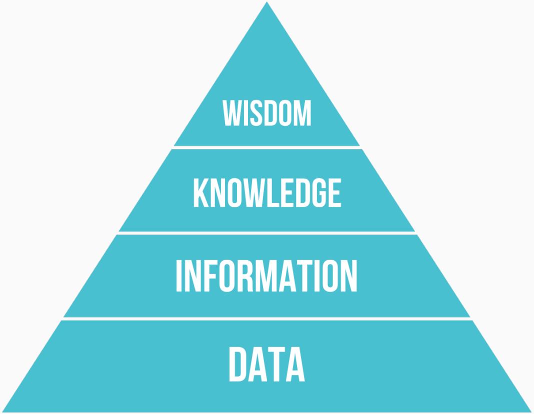
Introduction

Objectives

- Just Semantics
- How do we express and represent knowledge?
- How can computers express and represent knowledge?
  - HTML: a degenerate form of XML
  - XML: a degenerate form of RDF
  - The Resource Description Framework (RDF)
- Ontologies
  - What are they?
  - What do they know? Do they know things? Let's find out!
  - Differences between ontologies & databases
  - Ontologies in the real world
- Afternoon: Practical

Knowledge Representation

Moving from Information to Knowledge and Wisdom



- In learning databases, you will have learned how to take some data and store it as information on a computer.
- You have also learned how to create programs that use that data as knowledge.
- Now we will question what it is exactly that makes that data mean anything.
- We will also consider what it takes for a computer to have knowledge, and how we can move from knowledge to wisdom.

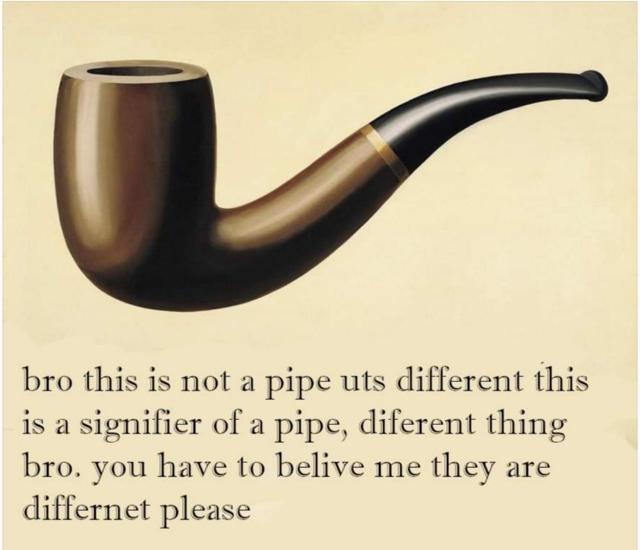
Epistemology

- What does it mean to know something? What is knowledge? Even for humans, it is an unsolved problem.
- For a long time, Justified True Belief was regarded as an adequate solution for the question of what it means to know something. You can be said to know something if your assertion meets the following conditions:
    1. You believe the proposition is true.
    2. Your belief is somehow justified (you have a reason to believe it).
    3. The proposition itself is true.
- In 1963, Edmund Gettier released a mere 3-page paper defeating this notion by introducing the Gettier Problem...
- Nevertheless, we persist with this general idea of knowledge, for the most part.

Wisdom Definition

Likely as many definitions as there are people who know the word, but in this case we mean: the ability to think and act using knowledge.

Symbols: Treachery of Images



Symbols and their Meanings

- This treachery lies not only in what might be called a picture, but also in words. For words are just another kind of image, or symbol.
- And so, what does it actually mean to be that which we refer when we say <u>pipe</u>?
- And how does the symbol <u>pipe</u> come to carry this meaning?
- Do these language symbols somehow directly correspond to reality (as in a picture), or are they very human mental constructs pointing vaguely at reality? (former: see Wittgenstein, latter: see later Wittgenstein).

Consensus Meaning

- These symbols are imbued with meaning by a complex cultural process of consensus, which is in a state of constant flux and ambiguity.
- We further describe both particular objects (you) and classes of objects (Human) with complicated syntactic forms, from which meaning is also derived in a cognitive process fraught with constant flux and ambiguity.
- Things are complicated further when we consider what might be said about an abstract noun, such as peace, green, or disco.

The Syntactic Web

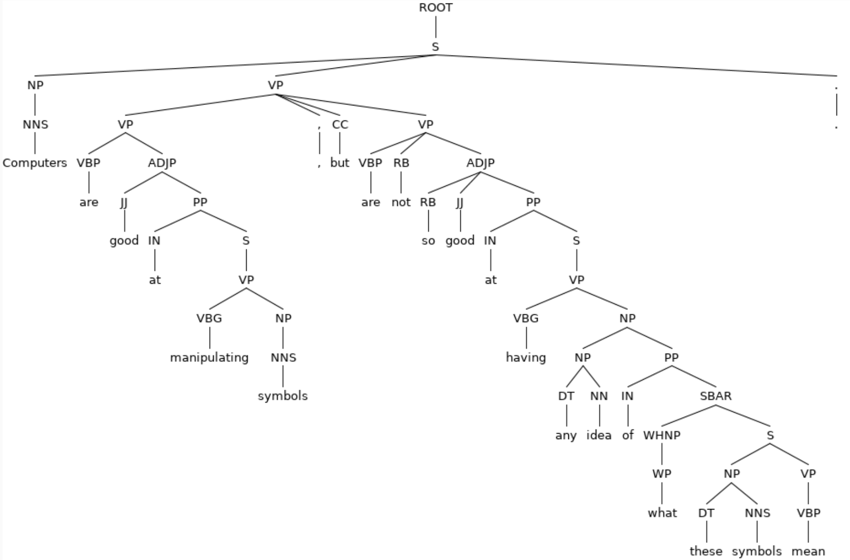
- We all know the syntactic web: the sum of human knowledge presented in natural language, and intended for human interaction.
- As such, meaning is derived and agreed by offline consensus.
- Examples: Wikipedia, BBC news, Facebook, most websites as you know them.
- But how do we make computers understand it? Can computers understand it?

Computers:

- Can count higher than you
- Needs to be a supercomputer to identify a dog

- But for written language, why can't we solve all our problems with natural language processing?
- Indeed, we can 'parse' natural language with computers...

Natural Language Processing



Limitations of Classical Natural Language Processing

- Consider the sentence "I made her fish," which has at least five potential meanings (double for plurality!):
  1. I cooked a fish for her.
  2. I cooked a fish belonging to her.
  3. I created the fish she owns.
  4. I forced her to go fishing.
  5. I made her into the Platonic ideal Fish.
- All problems in NLP may be reduced to the question of ambiguity - and it is <u>everywhere</u>; in human language, symbolic equivalence is not equivalent to semantic equivalence.
- Even without the problem of ambiguity, who's to say what any of these words mean?

Example inspired by "Speech and Natural Language Processing" 2nd ed by Daniel Jurafsky & James H. Martin.

The Limitation: Ambiguity and Meaning

- Computers are good at manipulating symbols, but are not so good at having any idea of what these symbols mean - their semantics.
- Furthermore, they struggle to translate symbols from ambiguous (read: human) languages into definite symbols to manipulate.

The Problem

- How do we make a computer understand the objects that we're describing, and the relationships between them?
- How can we make a computer know something?
- How can we gain wisdom from the knowledge we give a computer?
- Requirements:
  1. To be able to describe and to predicate upon particular entities. For example: this gibbon is very angry, or this tomato is mouldy.
  2. To be able to define the kinds of entities which may exist, and the relationships they may have between them. For example: a hand may have fingers, or a parent must have a child.

Separating the entity and the instance

- This separation of the class and the particular is along the lines of the Plato's Theory of Forms, or more generally Idealism: that particular objects (a chair) are partaking in a pure form which idealises and represents the most accurate reality of that object (The Chair or Chairness).
- This is one solution to the metaphysical problem of Universals.
- While rife with philosophical problems (much like any alternative theory), it happens to be extremely useful to us: computer scientists may notice a strong correlate when considering Object Orientated Programming: Classes and Instances.

The Semantic Web

- Utilises the infrastructure of the world wide web as a data communication methodology.
- Instead of only presenting data: link, interpret, and analyse data.
- In the consumer sphere this forms the basis for the 'Internet of Things.'
- Define things semantically in terms of how they relate to other things - their context.

Describing instances

Almost accidentally, the means by which we could develop a semantic framework for knowledge representation was already in heavy use throughout the world wide web.

HTML

Web pages are created in HyperText Markup Language (HTML)

 Hypertension Heart disease Right ventricular hypertrophy

Patient x has these symptoms:

- Hypertension
- Heart disease
- Right ventricular hypertrophy

Natural Language Inference

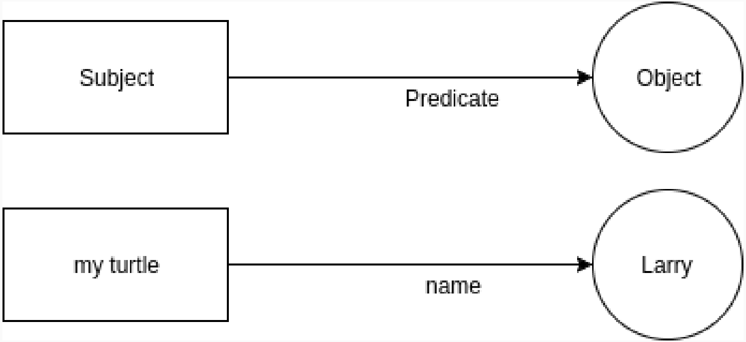
- As humans reading the list, he assertions we can infer from this text are numerous:
  1. A person exists.
  2. A person is a patient.
  3. There is at least one person.
  4. There is at least one patient.
  5. The patient has something wrong with their heart.
  6. At least one person has hypertension, heart disease, right ventricular hypertrophy.
  7. One person can have hypertension, heart disease, right ventricular hypertrophy simultaneously.
  8. ...

HTML: A Degenerate Form of XML

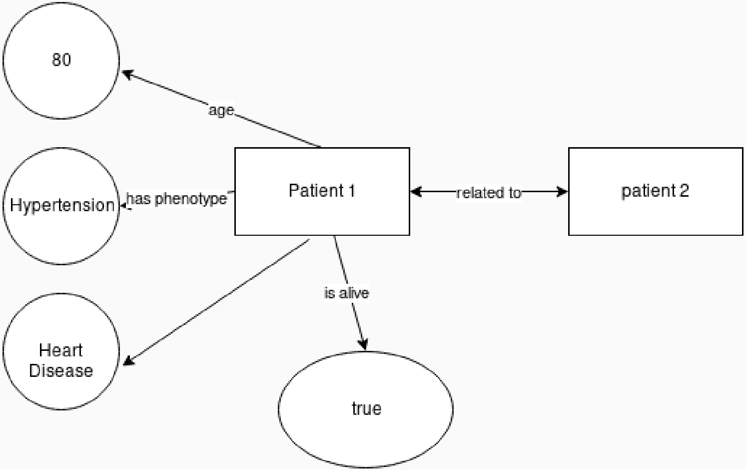
- HTML was later generalised, to create the eXtensible Markup Language (XML), without a focus on describing how a document should be rendered visually.
- This is a more general markup language, which can organise and ”mark up” or extend any kind of textual data with any additional information.

RDF: An Exhilirate Form of XML

- A method of representing structured data for the semantic web.
- Used heavily within science for experimental results, datasets and various open data, particularly in the biological and biomedical domains.



RDF: Semantic Networks



Wikipedia vs DBPedia (syntactic vs semantic web)

Aberystwyth

From Wikipedia, the free encyclopedia

Aberystwyth (Mouth of the Ystwyth, /ˌæbəˈrɪstwɪθ/, Welsh: [abɛrˈəstʊɨθ]) is a historic market town, administrative centre, and holiday resort within Ceredigion, West Wales, often colloquially known as **Aber**. It is located near the confluence of the rivers Ystwyth and Rheidol.

Historically part of Cardiganshire, since the late 19th century, Aberystwyth has also been a major Welsh educational centre, with the establishment of a university college there in 1872. At the 2001 census, the town's population was 15,935;[3] it was reduced to 13,040 at the 2011 Census. During nine months of the year, there is an influx of students—to a total number of 10,400 as of September 2012.[4] Including the suburbs of Llanbadarn Fawr, the population is 16,420.

Contents [hide]

dbo:areaCode	▪ 01970
dbo:ceremonialCounty	▪ dbr:Dyfed
dbo:country	▪ dbr:United_Kingdom ▪ dbr:Wales
dbo:distanceToCardiff	▪ 144840.960000 (xsd:double)
dbo:gridReference	▪ SN585815
dbo:populationTotal	▪ 13040 (xsd:integer)
dbo:postalCode	▪ SY23
dbo:principalArea	▪ dbr:Ceredigion
dbo:thumbnail	▪ wiki-commons:Special:FilePath/Aberystwyth_shore.jpg?width=300

Ontologies

What is an ontology, anyway?

Ontology: Describing Kinds of Objects

- Ontologies categorise, define and relate the kinds of things in a domain.
- Easily thought of as the 'schema' for the data represented by RDF (and in any other format).
- For example, we've talked about patients and diseases - but how do we define patients and diseases themselves?
- Exact definitions differ, but most ontologies share four main features:
    1. Classes and relations.
    2. Domain vocabulary.
    3. Metadata and descriptions.
    4. Axioms and formal definitions.

Ontology: Classes (and relations)

- A class is a category of things which can exist within a particular world, defined by the necessary and sufficient conditions for a thing to exist within that category (intensional definition).
- Each distinct class has its own IRI (Internationalised Resource Identifier), but may share labels, descriptions, or any other information with other classes.
- This does not necessarily make them semantically equivalent!
- e.g. HP:0002457 and MP:0000436 both have the label 'abnormal head movements'

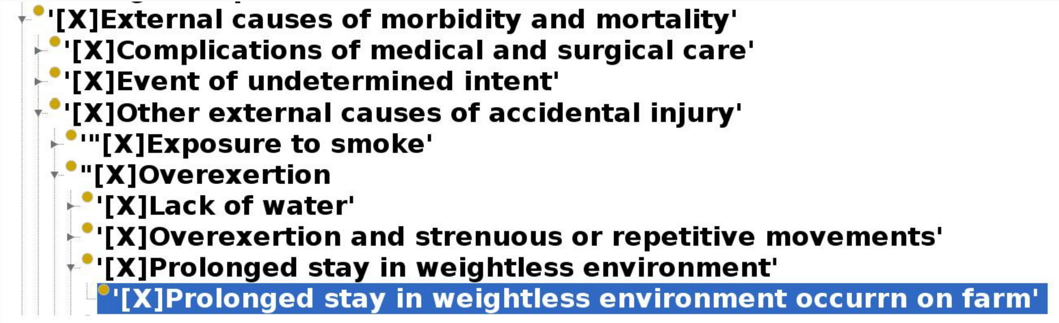
Ontology: (Classes and) relations

- Also concerned with the relationships between things. A fundamental relationship between classes is the is-a relationship.
- All ontologies start with the root class Thing, and all other classes in the ontology stand in a transitive subclass relationship to it.
- In this way, they form, at a most basic level, a taxonomy of categorisation for a particular universe of interest.
- Other common relationships are: part-of, inheres-in, caused-by, regulates.
- Many relationships carry deeply argued issues and side-effects: fingers!

Ontology: Domain Vocabulary

- Structured vocabulary for a domain
- Usually provided in the form of short 'labels' for each class (can have multiple)
- Provision for translated terms for classes
- Synonyms; the same category of things in a different context
- Together with the structure of the ontology, we gain a large and well-organised set of relevant terms for our domain
- Reduces ambiguity and misunderstanding in the conceptualisation for a domain

Ontology: Domain Vocabulary



Ontology: Metadata and Descriptions

- Descriptions, usually provided using the genus-differentia method.

  temporal pattern The speed at which disease manifestations appear and develop.
  insidious onset Gradual, very slow onset of disease manifestations

- Database cross-links, providing an assurance of semantic equivalence to data about an entity from an external source.
- Synonyms, broad or narrow

Ontology: Axioms and Formal Definitions

- The formal aspect of ontologies, axiomatic expression of meaning.
- Example: hypertension defined as has part some ((blood and (part-of some arterial system)) and (has-quality some ((increased pressure and (has-modifier some chronic)) and (has-modifier some abnormal))))
- Enable semantic analysis and (ostensibly) cross-species integration of knowledge.
- In many common ontologies phenotype axioms use the EQ (Entity-Quality) pattern
- We can also use these statements to query ontologies!

Ontology Reasoning

- Reasoning is the use of a classifier to evaluate all logical consequences of the explicit statements in an ontology.
- This allows us to infer new knowledge, and evaluate the consistency of our logical model (and can tell us why in the case of inconsistency).
- There are many different reasoners, which use slightly different methods and support different subsets of description logic (they are often said to have different levels of 'expressivity').

How do ontologies differ from databases?

Database	OWL Ontology
Closed World Assumption	Open World Assumption
Unique Name Assumption	No UNA
Schema constraints data structure	Axioms behave like inference rules

Web Ontology Language (OWL)

- Reference to Winnie the Pooh.
- Based on description logics, a fragment of first order logic.
- A set of language specifications based on description logics.

  Full Every construct in OWL is available. Fully undecidable to reason. Cardinality restrictions, domain, range, and value restraints.

  EL Subset of OWL which is guaranteed to be classifiable in polynomial time. Only way to reach inconsistency is through disjointness.

  RL A subset which mirrors the features of relational databases.

- Less expressive reasoners can simply ignore the more expressive content within ontologies.

OWL: Ontology Axioms

- `X SubClassOf: Y`: $X \xrightarrow{\text{is-a}} Y$
- `X SubClassOf: part-of some Y`: $X \xrightarrow{\text{part-of}} Y$
- `X SubClassOf: regulates some Y`: $X \xrightarrow{\text{regulates}} Y$
- `X DisjointWith: Y`: $X \xleftrightarrow{\text{disjoint}} Y$
- `X EquivalentTo: Y`: $X \overset{\equiv}{\Leftrightarrow} Y$, $\{X, Y\}$

OWL Syntax

- originally an extension of RDF and RDF Schema
- Several different (mostly) syntaxes

Consider the axiom $Parent \equiv Human \sqcap \exists hasChild.\top$

Functional Syntax

$$Parent \equiv Human \sqcap \exists hasChild.\top$$

```
EquivalentClasses(:Parent
  ObjectSomeValuesFrom(:hasChild owl:Thing))
```

RDF/XML Syntax

$$Parent \equiv Human \sqcap \exists hasChild.\top$$

```
<owl:Class rdf:about="http://example.com/demo-ontology.owl#Parent">
    <owl:equivalentClass>
        <owl:Restriction>
            <owl:onProperty rdf:resource="http://example.com/demo-ontology.owl#hasChild"/>
            <owl:someValuesFrom rdf:resource="&owl;Thing"/>
        </owl:Restriction>
    </owl:equivalentClass>
</owl:Class>
```

RDF Turtle Syntax

$$Parent \equiv Human \sqcap \exists hasChild.\top$$

```
:Parent rdf:type owl:Class ;

        owl:equivalentClass [ rdf:type owl:Restriction ;
                              owl:onProperty :hasChild ;
                              owl:someValuesFrom owl:Thing
                            ] .
```

OWL/XML Syntax

$$Parent \equiv Human \sqcap \exists hasChild.\top$$

```
<EquivalentClasses>
    <Class IRI="#Parent"/>
    <ObjectSomeValuesFrom>
        <ObjectProperty IRI="#hasChild"/>
        <Class abbreviatedIRI="owl:Thing"/>
    </ObjectSomeValuesFrom>
</EquivalentClasses>
```

Manchester OWL Syntax

$$Parent \equiv Human \sqcap \exists hasChild.\top$$

```
Class: Parent
    EquivalentTo:
        hasChild some owl:Thing
```

Manchester OWL Syntax

Manchester OWL Syntax gives us a human-readable construction and query language for description logic concepts in ontologies.

DL Syntax	Manchester Syntax	Example
$C \sqcap D$	C and D	Human and Male
$C \sqcup D$	C or D	Male or Female
$\neg C$	not C	not Male
$\exists R.C$	R some C	hasChild some Human
$\forall R.C$	R only C	hasChild only Human
$(\geq nR.C)$	R min n C	hasChild min 1 Human
$(\leq nR.C)$	R max n C	hasChild max 1 Human
$(= nR.C)$	R exactly n C	hasChild exactly 1 Human
$\{a\} \sqcup \{b\} \sqcup \ldots$	{a b ...}	{John Robert Mary}

Basic Inference Example

1. All meat comes from animals. (meat comes-from animal)
2. All beef is meat. (beef is-a meat)
3. Therefore, all beef comes from animals.

A simple subsumptive relationship, if the first two statements are true (note: our ontology isn't concerned with the truth of the statements), then the conclusion is necessarily true. So, our reasoner will add this logical statement to the ontology (beef comes-from animal)

Consistency Checking

- A reasoner evaluates every axiom in the ontology, and infers new knowledge.
- In doing this, we can identify whether any axioms are logically inconsistent with each other.

Basic Inconsistency Example

- Reptile is-a NOT Mammal
- Person is-a Mammal
- Turtle is-a Reptile
- Turtle is-a Person

Because a turtle is necessarily a reptile, and a reptile is necessarily not a mammal, a turtle cannot be both a reptile and a person.

Ontologies in the Real World

Overview

- Biomedical ontologies
- Application ontologies
- Ontology Uses and Examples
- Ontology Problems

Biomedical Ontologies

- Ontologies are used heavily in the biomedical field, as a source of systematised knowledge and as a tool for research.
- They describe things like diseases, drugs, side effects, phenotypes, or genes.

Biomedical Ontologies: Human Phenotype Ontology

- Describes more than 26,085 phenotypic abnormalities.
- Links these phenotypic abnormalities to their anatomy, and related systemic categories.
- Used to annotate biological and experimental databases such as OMIM.

Biomedical Ontologies: Human Phenotype Ontology

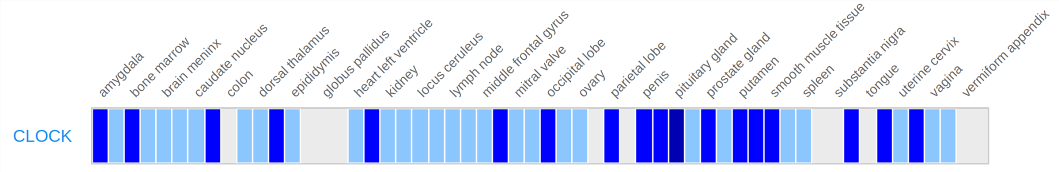
Annotation	Value
label	Hypertelorism
definition	Interpupillary distance more than 2 SD above the mean (alternatively, the appearance of an increased interpupillary distance or widely spaced eyes).
class	http://purl.obolibrary.org/obo/HP_0000316
ontology	HP
Equivalent	has part some (increased length and (inheres in some anatomical line between pupils) and (has modifier some abnormal))
SubClassOf	Abnormality of globe location
synonyms	Increased distance between eyes, Wide-set eyes, Excessive orbital separation, Ocular hypertelorism, Widely spaced eyes, Increased distance between eye sockets, Increased interpupillary distance, Widened interpupillary distance
database_cross_reference	UMLS:C0020534, SNOMEDCT_US:194021007, MSH:D006972, SNOMEDCT_US:22006008
has_alternative_id	HP:0007871, HP:0002001, HP:0004657, HP:0000578
has_obo_namespace	human_phenotype
id	HP:0000316

Application Ontologies

- Rather than categorise very general concepts, some ontologies are created around very specific concept areas, by particular methods or for particular methods.
- Because of their focus on a very specific set of concepts, it allows them to go into a lot of detail, or tailor the conceptualisation around it.
- These are known as application ontologies. Sometimes they are built into applications or have GUIs for use by clinicians or researchers.

Application Ontologies: EFO

- EFO: Experimental Factor Ontology
- Developed by EBI to annotate experimental results for their data, and to describe the experiments themselves.
- Includes classes and relations from many other ontologies, and define some of their own specific classes.
- Used for the annotation of data in Ensembl, ChEMBL, and the gene expression atlas.



Ontology Uses: Annotation

- The previous was an example of annotation.
- Annotation is attaching an ontology term to an entity in a database.
- By annotating an entity with an ontology term, we are indicating that it is an instance of that class.
- The annotation link allows us to link up with all the other information connected with the ontology.
  - Both in the ontology itself: superclasses, anatomy, side-effects
  - And in other databases using the ontology annotations: a common schema

Ontology Uses: Uveitis

- Uveitis is an inflammatory disease of the retina.
- The Human Phenotype Ontology already has a definition of uveitis (below)
- However, what if we want to more deeply characterise phenotypes with this disease? And what can we learn from such a characterisation?

- Uveitis
  - Anterior uveitis
    - Iridocyclitis
  - Intermediate uveitis
  - Iritis
  - Nongranulomatous uveitis
  - Panuveitis
  - Posterior uveitis

has part some (increased amount and (inheres in some (inflammatory response and (occurs in some uvea))) and (has modifier some abnormal))

Abnormal uvea morphology, Inflammatory abnormality of the eye

Ontology Uses: Uveitis

- An ontology describing the clinical phenotype of Uveitis was built, using a clinical document created by Uveitis clinicians
- The ontology was then expanded with synonys, using natural language processing on a patient discussion forum for the condition
- Contains 564 classes describing the phenotype of the disease

Ontology Uses: Uveitis

- By building the ontology from one patient forum, we were able to generalise it to another forum
- This also gained us a characterisation of these patients, using which we were able to learn about patient experience of the disease
- For example, we found with a sentiment analysis that patients were less positive if they mentioned anterior uveitis, steroid therapy, or biologic therapy.

Annotation: OMIM

- Online Mendelian Inheritance in Man
- A database of genetic phenotypes, annotated with HPO, and linked to diseases.
- We will be using this in the practical later...

OMIM Example

ACROKERATODERMA, HEREDITARY PAPULOTRANSLUCENT

Hair
- Fine-textured scalp hair

Inheritance
- Autosomal dominant

Immunology
- Atopic diathesis

Lab
- Skin lesions show orthohypergranulosis, acanthosis, and a relatively normal dermis

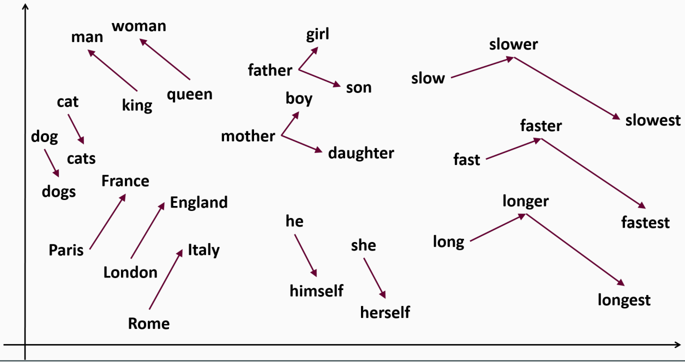
Skin
- Persistent, asymptomatic, yellowish-white, translucent papules and plaques of hands and feet

Ontology Uses: Semantic Similarity

- Semantic similarity is a measure of how similar two classes in an ontology are, based on the structure of the ontology.
- There are many different methods, but most use a form of Information Content.
- Information Content is a measure of how informative a term is
- You can also compare groups of terms, and link these to entities such as genes, proteins or diseases.
- This is used for protein-protein similarity prediction, disease diagnosis, functional similarity comparison, and more.

Ontology Uses: Vectorisation

- In Natural Language Processing, word2vec attempts to capture the semantics of a word by recording the words that surround it.
- For example, we might get the idea that the symbol ‘biscuit’ is some kind of food because it’s commonly used in sentences around verbs like ‘eat’, which is used with other foods.



Ontology Uses: Vectorisation

- Certain new approaches, such as OPA2Vec, attempt to use ‘sentences’ constructed from ontology axioms to build these similarity vectors
- Vectors also have measures of similarity, so we can use them in a similar way to semantic similarity
- Vectors can also be used as predictors in machine learning models! This means background information about concepts can be included alongside instance information in predictive models!

has part some (increased amount and (inheres in some (inflammatory response and(occurs in some uvea))) and (has modifier some abnormal))

Graphic on previous page taken from https://medium.com/analytics-vidhya/implementing-word2vec-in-tensorflow-44f93cf2665f

Ontology Uses: Natural Language Processing

- We can use ontologies to create vocabulary lists for natural language processing.
- When we do this, we also get semantically linked concepts, in the same way that we discussed when we spoke of annotation.

Some Problems in Ontology

- There are more than 550 biomedical ontologies, containing more than 7,736,665 classes and 80,832,785 axioms.
- How can we find what we're looking for?
- How can we access and integrate data?
- How can we integrate data between ontologies

Ontology Repositories

- Ontology repositories allow us to make use of ontology features.
- Finding, searching, downloading ontologies.
- Exploring and viewing classes, metadata, links to other databases.
- Advanced query functionality and programmatic API (AberOWL has the best one).
- Programmatic and interface-based data retrieval from many sources.
- Examples: Ontology Lookup Service, AberOWL, Bioportal, OntoQuery

Ontology Alignment

- Ontology alignment finds matches between classes between ontologies.
- For example, there are many different clinical coding systems e.g. ICD-10, SNOMED, READCODES.
- Another example: Finding class matches between the Human Phenotype Ontology, and the Mammalian Phenotype Ontology (mouse). In this case, it can help us integrate knowledge from mice and apply it to humans.
- There are two main approaches:
    1. Lexical: AgreementMakerLite
    2. Semantic: PhenomeNET
    3. Manual: MONARCH

Some Other Problems in Ontology

- Lots of people have very different ideas about doing things. There are some ontologies that cover very similar domains, such as the clinical coding ontologies, that are unnecessarily reproducing things and creating an alignment problem!
- The people who know how to create and modify ontologies are not the domain experts. This can lead to incorrect knowledge being stored in ontologies.
- A lot of knowledge is probabilistic, something that isn't supported by current widely used ontology technology.

Questions

Practical

Jupyter Notebook on the VM (first time)

You will need two(!) separate terminal windows:

```
ssh <username>@172.31.11.61
mkdir module3 # only the first time round
cd module3
git clone https://github.com/athro/msc_bio_module3_practicals.git # only the first time round
module load IPython/6.4.0-foss-2018b-Python-3.6.6
module load scikit-learn/0.20.0-foss-2018b-Python-3.6.6
pip3.6 install bioservices --user } # (you might want to use environments) # only the first time round
jupyter notebook --no-browser
```

The output will give you an available port automatically, i.e. the link similar to the following:

```
http://localhost:9014/?token=fb0f0bc5dd12d410bdb7cd1375254e2781d2b544eeea6b5c
```

Take note of this port (in this case 9014) and in another terminal window (on your local machine) execute the following command:

```
ssh -N -L <notedport>:172.31.11.61:<notedport> <username>@172.31.11.61
```

Hopefully you should be able to use the link `http://127.0.0.1:<notedport>` on your local browser.

Jupyter Notebook on the VM (afterwards!!)

Start with setting up jupyter:

```
ssh <username>@172.31.11.61
cd module3
module load IPython/6.4.0-foss-2018b-Python-3.6.6
module load scikit-learn/0.20.0-foss-2018b-Python-3.6.6
#git fetch --all # WARNING - this will delete ALL changes you have made
#git reset --hard origin/master # WARNING - this will delete ALL changes you have made
 jupyter notebook --no-browser
```

And than start the tunneling in the second terminal window (as about - gain take a not of the port).

Jupyter Notebook on own Laptop

- If possible install Jupyter Notebook. `https://jupyter.readthedocs.io/en/latest/install.html` for Python 3.7
- Within your terminal install bioservice

```
pip install --user bioservices # please check if you have to use pip3 or pip3.6
```

- No pip? Install: `https://pip.pypa.io/en/latest/installing/`
- Create and goto the folder `module3` and clone the practical repository (see slide before)
- Start the notebook

```
mkdir module3
cd module3
git clone https://github.com/athro/msc_bio_module3_practicals.git
jupyter notebook --no-browser --port=9000
```

Clean pull - ignoring all local changes

To get a clean pull from a git repository, i.e. one that overides any changes on your local filestore:

```
git fetch --all
git reset --hard origin/master
```