When and Why to use a Classifier?

Alan Rector

with acknowledgement to
Jeremy Rogers, Pieter Zanstra, & the GALEN Consortium
Nick Drummond, Matthew Horridge, Hai Wang in CO-ODE/HyOntUSE
Information Management Group Dept of Computer Science, U Manchester

Holger Knublauch, Ray Fergerson, … and the Protégé-Owl Team

rector@cs.man.ac.uk
co-ode-admin@cs.man.ac.uk
www.co-ode.org
protege.stanford.org
www.opengalen.org
When to use a classifier

1. At author time: As a compiler
   - Ontologies will be *delivered* as “pre-coordinated” ontologies to be used without a reasoner
   - To make extensions and additions quick, easy, and responsive, distribute developments, empower users to make changes
   - Part of an *ontology life cycle*

2. At delivery time: As a service:
   - Many fixed ontologies are too big and too small
     - Too big to find things; too small to contain what you need
     - Create them on the fly
   - Part of an *ontology service*

3. At application time: as a reasoner
   - Decision support, query optimisation, schema integration, …, …, …
   - Part of a *reasoning service*
When to use a classifier 1: Pre-coordinated delivery classifier as compiler

• The life cycle
  – Gather requirements, sketch, experiment
  – Establish patterns – design a “language”
    • Criteria for success: What a subject domain expert can learn in a few days
  – Bulk authoring
  – Classification
  – Quality assurance
  – Commit classifier results to a pre-coordinated ontology & deliver
    • Polyhierarchies (Protégé, DAG-Edit, OWL-Lite, RDF(S), Topic Maps, …
      – Query and use with your favourite tool
Commit Results to a Pre-Coordinated Ontology

Assert ("Commit") changes inferred by classifier
When to use a classifier 2: Post Coordination
Classifier as an inference engine

• When the ontology too big – “Lazy classification” on demand

Big on the outside: small kernel on the inside
Often combined with other services:
Example - the GALEN Server

Client

Applications

Users

Ontology Services

Service API

Multilingual Module
Multilingual Lexicons

Ontology Module
Ontologies & Classifier

External Resources & “Coding” Module
Resource References

Run time classifier
…but… Why use a Classifier?

- To compose concepts
  - Allow conceptual lego

- To manage polyhierarchies
  - Adding abstractions (“axes”) as needed
  - Normalisation
    - Untangling
      - labelling of “kinds of is-a”

- To avoid combinatorial explosions
  - Keep bicycles from exploding

- To manage context
  - Cross species, Cross disciplines, Cross studies

- To check consistency and help users find errors
Logic-based Ontologies: Conceptual Lego

hand
extremity
body
chronic
acute
abnormal
normal
ischaemic
gene
protein
cell
expression
Lung
inflammation
infection
bacterial
deletion
polymorphism
Logic-based Ontologies: Conceptual Lego

“SNPolymorphism of CFTRGene causing Defect in MembraneTransport of ChlorideIon causing Increase in Viscosity of Mucus in CysticFibrosis…”

“Hand which is anatomically normal”
Species

Genes

Protein

Function

Disease

CFTRGene in humans

Protein coded by (CFTRgene & in humans)

Membrane transport mediated by (Protein coded by (CFTRgene in humans))

Disease caused by (abnormality in (Membrane transport mediated by (Protein coded by (CFTR gene & in humans))))
Logic Based Ontologies: The basics

Validating (constraining cross products)

Primitives  Descriptions  Definitions  Reasoning

Feature

Structure

Heart  MitralValve  Encrustation

* ALWAYS partOf: Heart

* ALWAYS feature: pathological

Encrustation

+ involves: MitralValve
+ (feature: pathological)

Structure

+ feature: pathological
+ involves: Heart

Thing

+ feature: pathological

Thing

pathological  red

Heart  MitralValve  Encrustation

red  + partOf: Heart

Pathological + (feature: pathological)

Encrustation + involves: Heart

Red + (feature: pathological)
Example demonstrations: Take a Few Simple Concepts & Properties
Combine them in Descriptions which can be simple....

Sickle cell disease is a disease caused some sickling haemoglobin
Cystic fibrosis is caused by some non-normal ion transport that is the function of a protein coded for by a CFTR gene.
Add some definitions

“Diseases linked to CFTR Genes”
We have built a simple tree

easy to maintain
Let the classifier organise it
If you want more abstractions, just add new definitions (re-use existing data)

“Diseases linked to abnormal proteins”
And let the classifier work again
And again – even for a quite different category

“Diseases linked genes described in the mouse”
And let classifier check consistency
(My first try wasn’t)
Represent context and views by variant properties

- **Organ**
  - **is_part_of**
  - **OrganPart**
  - **is_clinically_part_of**
  - **Heart**
  - **is_structurally_part_of**
  - **CardiacValve**

**Disease of (Heart or part-of-heart)**

**Disease of Pericardium**
Integration of Contexts in Protégé-OWL

Generic part-of
FMA
Functional
Clinical

FMA
Functional
Clinical

is_part_of ↔ has_part
is_structural_part_of ↔ has_structural_part
is_subdivision_of ↔ has_subdivision
is_portion_of ↔ has_portion
is_distinct_part_of ↔ has_distinct_part
is_functional_part_of ↔ has_functional_part
is_clinical_part_of ↔ has_clinical_part

‘Internal’
Protégé-OWL alternative views

Disorder of “Clinical heart”
“Disorder of heart of any part of the heart”
(including clinical and functional parts)

Disorder of “FMA heart”
“Disorder of heart or any structural part of the heart”
Consequences for classification of diseases

Doctors consider it a heart disease, even though developmentally/structurally it is not.

Disorders of the things anatomists recognise as parts of the heart.
Summary: Why Classify?

• To compose concepts
• To untangle *polyhierarchies* – to *Normalise*
• To avoid *combinatorial explosions*
• To manage *context*
• To check *consistency* and *help users find errors*
Summary: When to Classify?
Applications do not need a classifier to benefit from classification

• Pre-coordination
  – If concepts/terms can be predicted
  – When classifier is not available at run time
  – When we must fit with legacy applications

• Post-coordination
  – When a few concepts are needed from a large potential set
  – When a classifier is available
    • and time cost is acceptable
  – When applications can be built or adapted to take advantage
Remember:
Think about the Life Cycle

• The life cycle for pre-coordinated ontologies
  – Gather requirements, sketch, experiment
  – Establish patterns – design a “language”
    • What a subject domain expert can learn in a few days
  – Bulk authoring
  – Classification
  – Quality assurance
  – Commit classifier results to a pre-coordinated ontology & deliver
    • Taxonomies (Protégé, DAG-Edit, OWL-Lite, RDF(S), Topic Maps, …)
      – Query and use with you favourite tool

www.co-ode.org