Normalising phenotype and disease terms by mapping to ontologies

Matt Wherlock

Background

- CVA report_event records contains phenotype descriptions
  - Focused on results returned by GMCs
- Some are programmatically entered
- Some are free-text entered by people
  - Inconsistent
  - Error-prone
  - Synonym choice

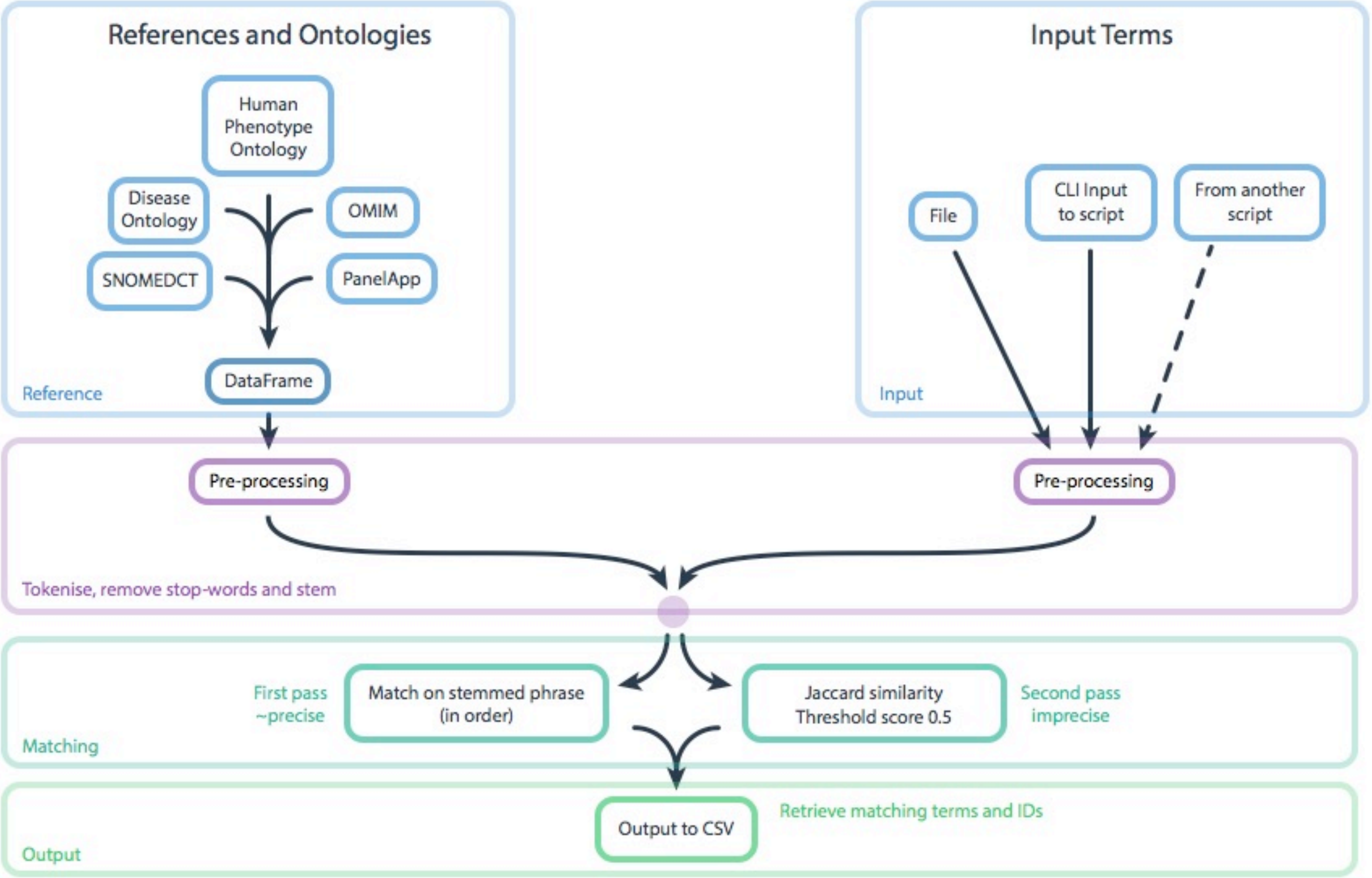
- “paediatric congenital malformation-dysmorphism-tumour sydromes”
- “Tumour” vs “Tumor”
- “rp53”

Aims

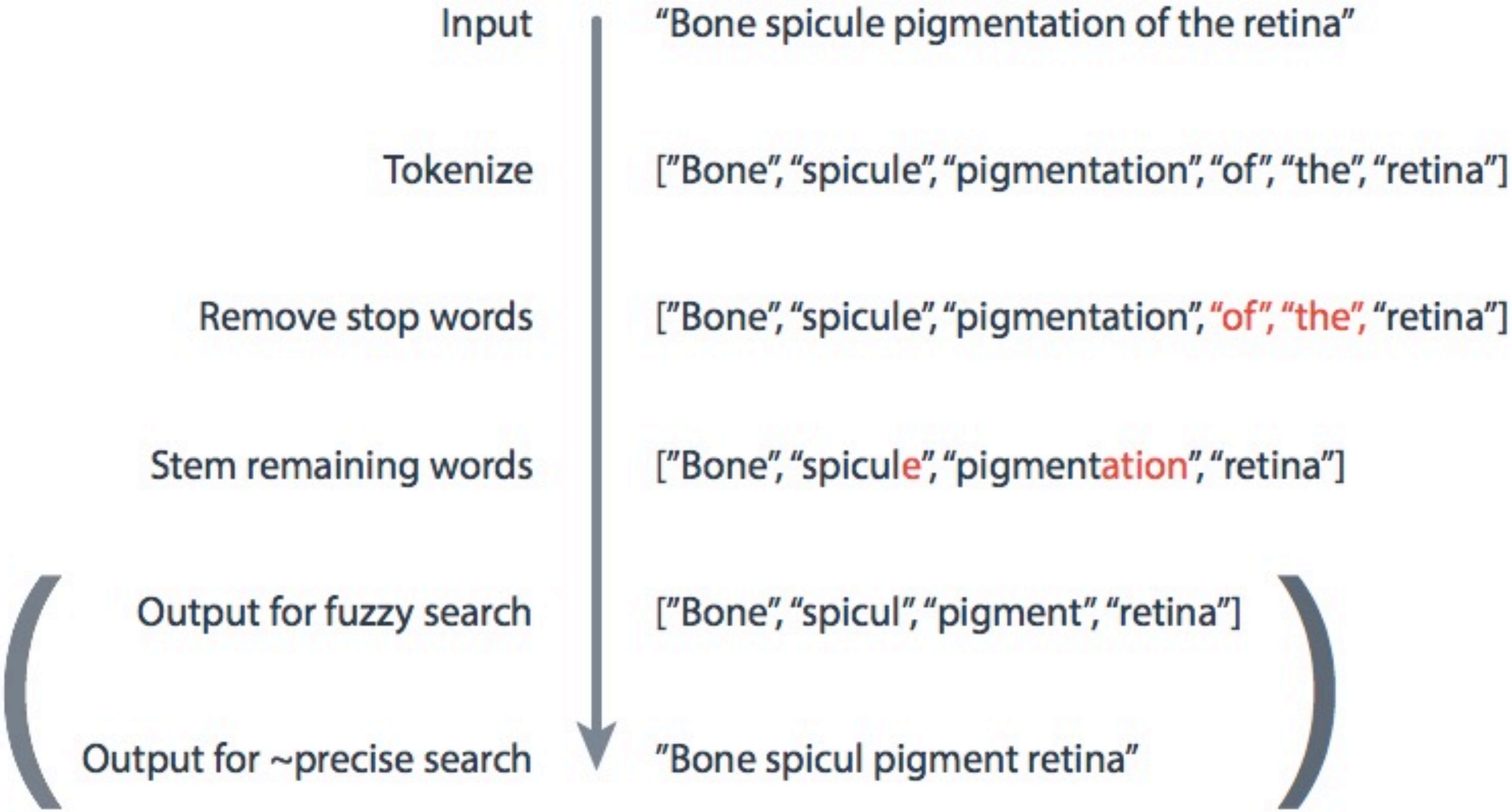
- Normalise phenotype entries
- Map to controlled vocabularies
- Implement imprecise searches to handle free-text issues

Initial strategies

- MongoDB $text indexes
    - Struggled with typos and US spellings
- Phonetic searches
    - NYSIIS, Dmetaphone
    - Good for many cases
    - Couldn’t discriminate some words
        - Head, Hand
        - Pons, Pinna, Penis


References and Ontologies
Human Phenotype Ontology
Disease Ontology
OMIM
SNOMEDCT
PanelApp
DataFrame
Reference
Input Terms
File
CLI Input to script
From another script
Input
Pre-processing
Pre-processing
Tokenise, remove stop-words and stem
First pass ~precise
Match on stemmed phrase (in order)
Jaccard similarity Threshold score 0.5
Second pass imprecise
Matching
Output to CSV
Retrieve matching terms and IDs
Output

Pre-processing



Input	“Bone spicule pigmentation of the retina”
Tokenize	[”Bone”, “spicule”, “pigmentation”, “of”, “the”, “retina”]
Remove stop words	[”Bone”, “spicule”, “pigmentation”, “of”, “the”, “retina”]
Stem remaining words	[”Bone”, “spicule”, “pigmentation”, “retina”]
Output for fuzzy search	[”Bone”, “spicul”, “pigment”, “retina”]
Output for ~precise search	”Bone spicul pigment retina”

Matching pre-processed terms

- ~ Precise
  - Tokenise
  - Remove stop-words
  - Stem remaining words
  - Merge back into a single string
  - Find match between term and reference
    - main or synonyms

- Fuzzy
  - Tokenise
  - Remove stop-words
  - Stem remaining words
  - Calculate Jaccard distance between term and reference
    - main or synonyms
  - Keep terms with a score of lte 0.5

Jaccard distance

$$\text{Jaccard distance} = \frac{\text{Number of items unique to the target term}}{\text{Total number of (unique) terms between target and match term}}$$

- A higher score reflects greater differences
- Testing showed 0.5 to be the best threshold for balancing specificity and sensitivity

The contents of CVA

- Extracted phenotype terms from the test instance of CVA
- 189 unique terms
- 186 mapped to ontology terms (98%)
- Around 10% matched ~ exactly
- Missed terms:
  - “non-specific”
  - ”na”
  - “rp53”
- But there are still some issues with specificity

Potentially awkward cases

- US spellings handled
  - “multiple endocrine tumors”
  - “Multiple endocrine tumours”
- typos fixed
  - “paediatric congenital malformation-dysmorphism-tumour sydromes”
  - “Paediatric congenital malformation-dysmorphism-tumour syndromes”
- Picked up synonyms in some cases

Potentially awkward cases

- Various ontologies matched

HPO	10%
DO	49%
OMIM	36%
SNOMED	3%
PANELAPP	3%

- Gene-specific OMIM entries
  - OMIM contains gene-specific disease entries as well as generic ones
  - Adapted code to minimize off-target

Problems still to resolve

- Some terms are not actually phenotypes or diseases
  - “non-specific”
- Some terms are heavily abbreviated
  - “rp53” is probably “retinitis pigmentosa 53” (OMIM)
- Fuzzy problems with ’without …’
  - “osteogenesis imperfecta” matched to “Dentinogenesis imperfecta without osteogenesis imperfecta”
  - Semantics lost
  - Set operations can be skewed by repeating words

Libraries and packages

- Python
- NLTK (Natural Language Toolkit)
- Pronto (parsing ontologies from .obo files)
- Pandas (dataframes)

Future work

- Performance is still an issue – jaccard scoring is slow over a large dataset
    - Numpy vectorization
    - Cython
- Specificity of some fuzzy searches
- Reduce the number of terms returned (esp OMIM)
- How to integrate returned results into data aggregation
- Package for PyPI or docker container

Thanks!

Thanks to everyone here for putting up with me during my elective placement!

Especially Pablo and Kevin for all their help and support, and the rest of interpretation for advice and tips