AtlasRDF

James Malone

October 20, 2016

1 Introduction

The AtlasRDF package exposes the data stored in the Expression Atlas RDF store at the European Bioinformatics Institute. This data concerns the reporting of genes which are differentially expressed under certain experimental and biological conditions in certain species. These conditions - often called experimental factors - include diseases such as cancer, phenotypes such as obesity, cell types such as epithelial cells, organism part such as brain and age. The data is also connected to other RDF stores such as proteins in UniProt and Pathways in Reactome, all of which this packages uses to enrich the queries available.

Here we demonstrate how to use the Atlas to extract a gene list based on a condition of interest to and to explore other data that may offer information about the condition and/or gene list using the AtlasRDF R package.

The package is loaded using

2 Extracting a gene list

First of all we will find the appropriate ontology term to query the database with. AtlasRDF uses EFO <http:/www.ebi.ac.uk/efo> to annotate data with so we will find the appropriate EFO class URI for our search term which is type II diabetes.

```
> termhits <- searchForEFOTerms("type II diabetes")
> print(termhits)
```

If you already use another ontology, we can also find an appropriate EFO term by using the term mappings maintained by NCCBO Bioportal <http://bioportal.bioontology.org/> Here we do a simplel query of passing the URI of the class you use (in this example the URI for type II diabetes from SNOMEDCT) to find any matching terms in EFO, as mapped by BioPortal.

```
> efomappings <- getOntologyMappings("<http://purl.bioontology.org/ontology/SNOMEDCT/44054006>")
> print(efomappings)
```

We select the class URI for type II diabetes called 'type II diabetes mellitus' and extract get a gene list based on those genes differentially expressed in humans for type II diabetes. First we get the URI for the species - human in this case. Then we query for the genes for diabetes Type II.

```
> humanURI <- getTaxonURI("human")
> typeIIgenelist <- getSpeciesSpecificEnsemblGenesForExFactor("<http://www.ebi.ac.uk/efo/EFO_0001
> head(typeIIgenelist)
>
```

3 Extracting pathways from genes

We now have a list of candidate genes of interest. We will now refine the list of approx 250 genes by finding genes which share some common signalling pathways. The following block of code may take a little while to run.

```
> #this will take a little while to run so we comment out for the vignette
> pathways <- getRankedPathwaysForGeneIds(typeIIgenelist[,3])
> head(pathways)
>
```

We can now see a list of pathways that these genes are connected to, with the most common pathway ranked as the first element in the list.

```
> pathways[1]
```

4 Finding related conditions

Given this pathway, we can extract some further information. For instance, let us now find other experimental conditions that are connected to this pathway. First of all let's find other genes attached to this pathway.

```
> #if you have run the previous steps you can extract the pathway uri from the pathways list usin
> #pathways[[1]]@pathwayuri
> #for convenience we will name the pathway
> genes <- getGenesForPathwayURI("<http://identifiers.org/reactome/REACT_12627.3>")
>
```

5 Gene set enrichment with Expression Atlas

We can ask a few other questions about these genes. For instance, what other experimental factors are these genes differentially expressed for? We can use the Atlas enrichment functions to perform this task.

Firstly, we need to download the human background data set. We require two background sets for each species we wish to perform enrichment for, since these are human genes we will download the human backgrounds. These can be found at https://github.com/jamesmalone/AtlasRDF-R/tree/master/src/backgroundsetdata in the human subfolder you will find the two background data sets required.

After downloading we need to load these files into our workspace.

```
> #human_genelist_bg
> load("human/human_gene_list.RData")
> #human_factor_counts
> load("human/human_factor_counts.RData")
```

Finally do the enrichment, passing the gene list

```
> ###do enrichment
> #transcription_pathway_enrichment <- doFishersEnrichment(genes, human_genelist_bg, human_factor
```

If you don't have access to the enrichment background data sets, you can use the example enrichment result set included. To load this data use the command:

```
> data(transcription_pathway_enrichment)
```

You can now visualise these results in a graph.

```
> vizgraph <- vizPvalues(transcription_pathway_enrichment, "0.00005")
```

There are a lot of resuts here, even with a low p-value threshold, so let's filter some types of factors of interest. Let's look at just disease factors these genes are enriched for. The ontology class for disease is efo:EFO_0000408 so let's include only subtypes of this class, i.e. diseases

We can find the class in efo by using the search function since we will not usually know what the identifier of an EFO class is when we perform this filtering.

```
> filteredgenes <- includeOnlySubclasses("efo:EFO_0000408", transcription_pathway_enrichment)
```

Though we still have quite a lot of results, even from this filter, so let's take the top 20 most enriched experimental factors.

```
> sortedset <- orderEnrichmentResults(filteredgenes)
> vizgraph <- vizPvalues(sortedset[1:20], 0.000001)
```

This points us to other factors that may be of interest to this pathway. In addition we can explore the genes enriched for these particular factors. If we look at the top enriched factor we can see the gene list for it.

```
> sortedset[1]
```

```
[[1]]
An object of class "enrichmentresult"
Slot "factoruri":
[1] "<http://www.ebi.ac.uk/efo/EFO_0000571>"

Slot "label":
[1] "lung adenocarcinoma"

Slot "p.value":
[1] 3.30217e-14

Slot "estimate":
odds ratio
 0.2241801

Slot "alternative":
[1] "two.sided"

Slot "null.value":
odds ratio
         1

Slot "method":
[1] "Fisher's Exact Test for Count Data"

Slot "enrichedgenes":
 [1] "<http://identifiers.org/ensembl/ENSG00000142556>"
 [2] "<http://identifiers.org/ensembl/ENSG00000205246>"
 [3] "<http://identifiers.org/ensembl/ENSG00000167232>"
 [4] "<http://identifiers.org/ensembl/ENSG00000170265>"
 [5] "<http://identifiers.org/ensembl/ENSG00000204946>"
```

```
 [6] "<http://identifiers.org/ensembl/ENSG00000244560>"
 [7] "<http://identifiers.org/ensembl/ENSG00000105497>"
 [8] "<http://identifiers.org/ensembl/ENSG00000170949>"
 [9] "<http://identifiers.org/ensembl/ENSG00000147117>"
[10] "<http://identifiers.org/ensembl/ENSG00000196646>"
[11] "<http://identifiers.org/ensembl/ENSG00000067560>"
[12] "<http://identifiers.org/ensembl/ENSG00000115414>"
[13] "<http://identifiers.org/ensembl/ENSG00000140548>"
[14] "<http://identifiers.org/ensembl/ENSG00000229809>"
[15] "<http://identifiers.org/ensembl/ENSG00000198300>"
[16] "<http://identifiers.org/ensembl/ENSG00000122386>"
[17] "<http://identifiers.org/ensembl/ENSG00000130726>"
```

6 Getting raw data from ArrayExpress R

We could follow this up by extracting data from ArrayExpress for these genes using the ArrayExpress R package. To extract the experiment IDs we can use the function getExperimentIdsForGeneURI which will return a list of ArrayExpress experiment IDs which can be used with the ArrayExpress R package to obtain the source data files.

```
> #get experiments for first enriched gene
> experiments <- getExperimentIdsForGeneURI(sortedset[[1]]@enrichedgenes[1])
> #look at first 5 experiment IDs
> experiments[1:5]
```

7 Useful links

The ArrayExpress R package is available from http://www.bioconductor.org/packages/release/bioc/html/ArrayExpre
Gene Expression Atlas is available at http://www.ebi.ac.uk/gxa
The RDFAtlas is available at http://www.ebi.ac.uk/rdf/services/atlas/
EFO, the ontology which is used to describe experimental factors, is available at http://www.ebi.ac.uk/efo