Customized automatic corpus annotations using AlvisNLP/ML

Robert Bossy, Mouhamadou Ba, Claire Nédellec

Mathématiques et Informatique Appliquées du Génome à l’Environnement – Bibliome
Institut National de la Recherche Agronomique

16 January 2017 / BLAH3
Who we are

and what we do
Domains of interest: agriculture, food, biology, ecology, sustainable development.

We are a research team in a bioinformatics lab (MaIAGE).

Our specialties: NLP applied to biology, domain-specific Knowledge Acquisition.

Our approach

- Our research and developments always start with and support applied services for end-users.
- We make developments as generic and reusable as possible.
Achievements

Software

- AlvisAE: annotation editor.
- TyDI: terminology and ontology editor.
- AlvisIR: semantic search engine framework.
- AlvisNLP/ML: corpus processing engine.

Projects

- We joined the BioNLP Shared Task in 2011 (BB3 and SeeDev tasks).
- We are part of EU project OpenMinTeD:
  - Objective: offer a text-mining infrastructure for researchers.
  - Lessons could be drawn from this Hackathon.
AlvisNLP/ML

what it is and how it works
NLP workflow engine

Allows users to assemble processing modules into NLP pipelines.

Design goals

- *Reproducibility* of data processing.
- Easy *adaptation* to new documents/data/problems.
- Allow *sharing* pipelines.
- *Scalability*.

Library of modules

- I/O: read and write in a variety of formats.
- Linguistic: tokenization, POS-tagging, parsing, term extraction.
- Generic: lexicon projection, regexp.
The Plan

The plan is the file where the user specifies their pipeline:
- The sequence of modules.
- External resources to be used.
- Parameters for each module.

```xml
<alvisnlp-plan id="REN">
  <module id="reader" class="XMLReader2">
    <sourcePath filter=".xml$">../corpus/Quaero_t3.2_gene_dev+train-v1.1</sourcePath>
    <xsltTransform>../../bibliome/share/xslt/gene-train2alvisnlp.xslt</xsltTransform>
  </module>

  <import file="../resources/segmig.xml" id="segmig"/>

  <module id="tt" class="TreeTagger">
    <treeTaggerExecutable>/bibdev/install/tree-tagger-3.2/bin/tree-tagger</treeTaggerExecutable>
    <parFile>/bibdev/install/tree-tagger-3.2/lib/english.par</parFile>
  </module>

  <module id="train" class="TrainingElementClassifier">
    <algorithm>weka.classifiers.bayes.NaiveBayes</algorithm>
    <classifierFile>classifier.model</classifierFile>
    <relationDefinition>attributes.xml</relationDefinition>
    <examples>documents[@set="train"].sections.layer:candidates</examples>
  </module>
</alvisnlp-plan>
```
Running the plan

**Command-line interface**

The Most Useful Interface.

**REST interface**

- Exposes the documentation.
- Allows to expose selected plans and parameters.
- Users can run plans synchronously or asynchronously.
Our proposal
and the means to achieve it
Link the REST service

Initial idea
- Make the REST service understand the PubAnnotation protocol.
- In order to turn any plan exposed by the REST service into a linked annotation tool.

Benefits
- More exposure of our developments.
- We bring a potential of a collection of NLP annotation tools at once.
Materials

Source code
- Java.
- ALv2.
- https://github.com/Bibliome/alvisnlp

Working deployment
- Ready-to-use plan:
  - bacterial taxon named-entity recognition.
- Running the application on my laptop, Jetty Maven plugin, maybe INRA’s servers.

Workforce
- There are two of us.
- Main AlvisNLP/ML developer.