TJBatchExtractor

Kyle Miller

August 21, 2014

1 Introduction

This document describes the TJBatchExtractor project. This project is a regular expression based information extractor designed to operate on text captured from female escort advertisements originating from US sections of Backpage.com. The motivation is to extract domain specific information that may be informative in identifying individuals or groups responsible for each advertisement. To that end, the information extracted focuses on physical description and contact information.

The project is built upon GATE (General Architecure for Text Engineering) produced by the University of Sheffield. GATE is available at

```
https://gate.ac.uk.
```

All necessary dependencies for the TJBatchExtractor projects are contained within this repository. It should not be necessary to download GATE. GATE may be useful however, if one wishes to modify this project an/or make use of the GUI and performance analysis machinery GATE provides.

In the context of GATE applications, there are a few key phrases for which knowing the definitions may prove helpful. These are given below.

Language Resource A document or collection of documents (corpus)

Processing Resource A self contained program designed to consume and produce/modify annotations on text

Application A collection of processing resources arranged in a pipeline

JAPE A regular expression language that operates on annotations

Gazetteer A look up table that produces annotations

2 Components

The TJBatchExtractor project consists of the following components.

2.1 TJBatchExtractor.java

The java program TJBatchExtractor.java is a wrapper for the GATE application called TJinformationExtractor. It’s function is to wrap TJinformationExtractor in such a way that it can load a text document and process many advertisements in parallel, finally writing the results to a csv file upon completion. TJBatchExtractor.java expects a single text file to be provided as input on which each line represents the text associated with an advertisement. Thus, it is necessary to remove line breaks from ad text. The program writes its results to a user specified file, ”Out.csv” by default.

To compile:

```
$ javac -classpath ’.:/dependencies/*’ TJBatchExtractor.java
```

To run:

```
$ java -classpath ’.:/dependencies/*’ TJBatchExtractor [num_threads]
    [textfile] [outfile]
```

2.1.1 Outfile

The following is a description of the features extracted, their definitions, and notes about notation. Many features may be multi-valued (e.g. names, phone numbers). In such cases, the output file uses ’;’ to delimit values within ’,’ delimited fields.

It should be noted that many of these features have not been normalized. So, for example, blond and blonde may both appear as hair colors, despite having the same interpretation.

Perspective_1st Count of 1st person pronouns

Perspective_3rd Count of 3rd person pronouns

Name Female first names

Age Age

Cost Dollar figure charged for various services. Notation is given as Dollar/Measure/Unit. Dollar represents a cost, Unit represents object of the cost (e.g. hours, minutes, short stay, special, etc.), Measure represents the number of units (e.g. 30 minutes, 1 hour, hhr, etc.)

Height_ft Height in feet, multiple values correlate with multiple values of Height_in

Height_in Remaining inches of height, correlates with Height_ft

Weight Weight in lbs

Cup Cup size

Chest Chest measurement

Waist Waist measurement

Hip Hip measurement

Ethnicity Country referenced ethnicity (e.g. Spanish, Russian, etc.)

SkinColor Color of skin

EyeColor Color of eyes

HairColor Color of hair

Restriction_Type One of [no, over]; the type of restriction, i.e. ”no black men”, or ”only men over 45.”

Restriction_Ethnicity The ethnicity/ skin color restricted

Restriction_Age The threshold age value for the over restrictions

PhoneNumber Phone number

AreaCode_State State associated with phone number’s area code

AreaCode_Cities Cities/ locations associated with phone number’s area code

Email Email address

Url urls specifically referenced or linked to in the body

Media iframes and other foreign sourced content

3 TJinformationExtractor

TJinformationExtractor is a GATE application, the definition of which is stored in

```
/TJInfoExtractor/application.xgapp.
```

The xgapp file may be loaded directly into GATE by selecting ”Restore Application from File...” allowing one to make use of the GATE interface.

This pipeline makes use of the ANNIE English Tokeniser that ships with GATE, two custom processing resources Integer_Tagger and Phone_Number_Tagger, an ANNIE Gazetteer, and JAPE-Plus transducer. The application begins by tokenizing the text. It then identifies all token/sequences of tokens that could be interpreted as an integer. Next, it identifies phone numbers (US and Canada). Finally, it identifies other features and attempts to resolve conflicts. Note that the entire application has been configured to be case insensitive.

3.1 Integer Tagger

The integer tagger is a processing resource that identifies references to integers in noisy (possibly obfuscated) text. The source can be found in

```
/TJInfoExtractor/plugins/Tagger_Integer/src/gate/creole/integers/.
```

It produces the ”Integer” annotation. Each integer annotation indicates the type (”numbers”, ”words”, or ”wordsAndNumbers”), numeric value, and whether any leading zeros are present.

The resource can be set to respect token boundaries (set to false in this application), sentence boundaries (set to false in this application, requires the use of a sentence splitter prior to integer tagger in the pipeline), and dictionary entries (set to true in this application). If set to true, respectTokenBoundaries prevents the resource from annotating a set of characters that cross token boundaries as an integer. If set to true, respectSentenceBoundaries prevents the resource from annotating a set of characters that cross sentence boundaries as an integer. If set to true, respectDictionaryEntries prevents the resource from annotating characters that are contained within a token that appears in the dictionary as an integer. For example, ”one” in will be tagged as an integer in ”some one” but not in ”a cone.”

The resource makes use of dictionaries of integer characters, symbols, and dictionary words. These files are located in

```
/TJInfoExtractor/plugins/Tagger_Integer/resources/languages/.
```

Standard numerical symbols 0-9 are hard coded into the resource.

3.2 Phone Number Tagger

The phone number tagger is a processing resource that identifies references to US and Canadian phone numbers. The source can be found in

```
/TJInfoExtractor/plugins/Tagger_PhoneNumber/src/gate/creole/phonenumbers/.
```

The resource can be configured to find both 10 digit and 7 digit phone numbers (both true by default). The tagger can also be configured to respect sentence boundaries as described above (set to false in this application).

The resource produces the ”PhoneNumber” annotation including the state, region, and value associated to a phone number. This resource finds groups of integers, annotated by the integer tagger, that are separated by no more than two word tokens (as indicated by the tokenizer). Note, an unlimited amount of punctuation may separate integers and they will still be grouped together. Each group of numbers is then analyzed for subgroups of size 7, 10, or 11 that have the following properties. 11 digit groups must begin with 1. The area code in 10 and 11 digit groups must be valid according to an area code dictionary located in

```
/TJInfoExtractor/plugins/Tagger_PhoneNumber/resources/.
```

For all groups the prefix (first three digits following the area code) must not begin with a 1 nor end in 11.

Finally, for larger groups of digits, phone numbers are tagged in a left first greedy fashion modulated by preferences on digit subgrouping. In this context, digits are considered in the same subgroup if there are no non-white space tokens between them. 10 and 11 digit phone numbers are preferred over 7 digit numbers. 11 digit numbers are treated as 10 digit numbers, ignoring the first digit. For 10 digit numbers preference is given to subgrouping sizes according to the following order 3-3-4, 10, 1-1-1-1-1-1-1-1-1-1, *-3-4, 3-7, 3-*, 6-4, *, where * indicates arbitrary digit subgroupings. For 7 digit numbers digit numbers, digit subgrouping sizes must be one of the following (in order of preference) 3-4, 7, 1-1-1-1-1-1-1.

3.3 ANNIE Gazetteer

The gazetteer lists can be found in

```
/TJInfoExtractror/application-resources/TJ_Gazetteer/.
```

This gazetteer resource contains look up tables for prefixes, key words, and suffixes associated to the features of interest (e.g. colors, hair, eyes, names, cup size, etc.). Occurrences of these tokens are annotated for use by the JAPE transducer.

3.4 JAPE-Plus Transducer

JAPE is a regular expression language for GATE that operates over annotations. The JAPE transducer creates or modifies annotations based on matches to rule patterns. The jape files containing the JAPE rules for this application are contained in

```
/TJInfoExtractor/application-resources/jape_transducers/.
```

Each rule file is run in the order specified in TJ_Annotation.jape. These rules try to match common linguistic patterns employed when mentioning each feature. Ultimately they are responsible for creating the feature annotations as well as resolving conflicts between annotations.

4 Performance

Below are performance measures for the features extracted, measured on 1000 randomly selected ads, adjudicated by hand.

Features that have sub-features such as height, cost, and restrictions, are evaluated on match of all sub-features for the purposes of this evaluation. Also, note that here the age feature is evaluated only on age mentions in the body of an add. While it is common to indicate age in the ad title, this extractor was specifically constructed to operate on the body.

Prevalence The count of feature occurrence/ number of ads

Correct The count of feature occurrences correctly extracted

Partial The count of feature occurrences extracted, albeit incorrectly

Missing The count of feature occurrences missed (i.e. false negatives)

False pos The count of falsely identified occurrences

Feature	Prevalence	Correct	Partial	Missing	False Pos	Recall	Precision	F-measure
Age	0.067	49	0	18	1	0.731	0.980	0.838
Cost	0.55	489	0	61	17	0.889	0.966	0.926
Email	0.012	12	0	0	0	1.000	1.000	1.000
Ethnicity	0.29	254	0	36	8	0.876	0.969	0.920
EyeColor	0.106	102	0	4	0	0.962	1.000	0.981
HairColor	0.266	255	4	7	1	0.959	0.981	0.970
Name	0.993	795	7	191	85	0.801	0.896	0.846
PhoneNumber	1.077	1072	2	3	0	0.995	0.998	0.997
Restriction	0.069	56	2	11	1	0.812	0.949	0.875
SkinColor	0.105	102	0	3	3	0.971	0.971	0.971
Url	0.047	41	5	1	2	0.872	0.854	0.863
Height	0.236	227	3	6	2	0.962	0.978	0.970
Measurement	0.206	182	16	8	0	0.883	0.919	0.901
Weight	0.182	166	4	12	0	0.912	0.976	0.943