Unsupervised Discovery and Extraction of Semi-structured Regions in Text via Self-Information

Author
Institute
Address
email@organization

Author
Institute
Address
email@organization

Author
Institute
Address
email@organization

ABSTRACT

We describe ongoing work into a general method for identifying and extracting information from semi-structured regions of text that are embedded within a natural language document. These are regions of text, usually in an ad hoc schema, forming structures such as tables, key-value listings, or long and repeated enumerations of properties. They present problems for standard information extraction algorithms that rely on regular grammatical text, as information is encoded in a combination of spatial layout, boilerplate, and repeated strings. Unlike previous work in table extraction, which relies on a relatively noiseless two-dimensional layout, our aim is to accommodate a wide variety of structure types. Our approach is an unsupervised one, based on identifying regions of suprising regularity inside the document. Here, regularity is measured by self-information, and is derived from patterns of semantically meaningful classes of text and visual layout. We present the results of an initial study to assess the ability of these measures to detect semi-structured text in a corpus culled from the web, showing that they outperform baseline methods on an average precision measure. We present initial work that uses significant patterns to generate extraction rules, and conclude with a discussion of future directions of our work.

Keywords

Semi-structured natural language extraction

1. INTRODUCTION

In the course of processing text documents using information extraction (IE) techniques, a key challenge arises when dealing with semi-structured text, where information is still presented textually, but is in a largely ungrammatical form and adheres to an unknown schema [6]. Examples include tables and property-value listings that are embedded in the context of a larger document, such as a report. These regions confound standard IE systems, as they expect grammatical text, and the rich information contained is garbled or lost. For example, the following extract from the FBI's most wanted list [1] has little or no grammatical structure,

```
ERIC JUSTIN TOTH
Height: 6’3"
Weight: 155 pounds
NCIC: W315591233
Hair: Brown
```

Unless special accommodation is made for this style of text, an IE system would not be able to identify the property value pairs and attach them to the subject.

In a survey of a collection of documents culled from the web, we found semi-structured text in very different formats, such as property-value statements (per the above example), tables, and logfile-like enumerations of properties. The schemas were also highly variable, even within the same type of structure. Thus when presented with a new corpus to perform IE over, we must account for a variety of structure types and essentially unknown schemas. Our aim is to develop a method that can deal flexibly with this variability, identify semi-structured regions, and present candidate extraction templates for those regions, ideally with little or no supervision.

For this work, we focus on extraction against pure text, instead of solely in formats such as HTML, XML, or PDF. There is a huge variety of formats and presentation methods, ranging from PPT to dynamic canvases in HTML5, and accommodating them all would be infeasible, thus working directly with the rendered text affords us the most flexibility. We also constrain our analysis to English language documents, although the techniques here could be extended to other langauges.

There has been a large body of work conducted to extract tables from plain text, namely [2] and [4]. The former makes use a combination of visual layout cues and language modeling to identify non-contiguous spans of text, and the latter uses a conditional random field over a combination of visual and textual features to identify types of table structures (header, super-header, content). However, the focus of this body work was to identify and extract from tables, whereas we aim to work with other types of structures. Also, [4] is a supervised algorithm, requiring training examples that may not be on hand for a new corpus. A survey of table extraction research [5] shows most methods rely on visual appearance of the text, but rely on a consistent two dimensional layout [2]. Given our experience, this may not always be the case, particularly for documents that were converted.

Perhaps the work closest in intent to ours are the work in extracting rules from freetext for the WHISK system [6] and from logfiles

[1] http://www.fbi.gov/wanted

[2] Note this body of work also implicitly presumes the use of a fixed width font, which we assume as well for the sake of simplicity.

in the PADS system [1]. In the WHISK system, extraction rules are induced from a set of annotated examples, and successively refined. Although this certainly is a route we would investigate in the future, our immediate goal is to deduce as much as we can about the semi-structured regions first. The PADS system attempts to induce extraction rules over textual logfiles, where no schema is readily available. Their approach centers on first identifying *chunks*, semantically meaningful units texts whose meaning is apparent. For example, the string "127.0.0.1" can reasonably be construed to be an IP address, and *john.smith@unit.gov.uk* is an email. By applying a set of rules expressing known relationships between chunks, an extraction grammar can be generated. This work embodies one of our key desiderata, the ability to derive a schema by inferring relationships between chunks of text, or *schema on read*. However, this work operated over log files, where a regular and uniform structure can be assumed to apply throughout the document, and there is no need to be able to identify these semi-structured regions from a background of regular text.

Based off an analysis of the document corpus, we observed the following:

1. Semi-structured regions tend to exhibit regular and repeated orderings of groups of semantic classes, categories of text.
2. These patterns tend to be longer and repeated throughout semi-structured regions, compared to prose.
3. Compared to prose , the spatial arrangement of text in semi-structured regions is also unusually regular.

Similar to the notion of chunks in PADS, Semantic classes represent spans of text whose general categorical meaning can be inferred directly from the text itself, without requiring contextual information. Examples include surnames, locations, telephone numbers, zip codes and date expressions. The intuition here is text that expresses a specific property tends to have values drawn from a similar semantic category. For example, a single column across a set of table rows tend to be drawn from the same genre, such as numbers, or strings expressing city names.

Following these observations, our hypothesis is semi-structured text will exhibit surprising regularity compared with prose. To model regularity, we characterize a document into multiple orders of ngrams of semantic classes and text runs. Runs are codewords that approximate the visual appearance of non-whitespace lines on a page, and in sequence roughly characterize the visual look of a line. For text T, its self-information $I(T)$ is given in Formula 1.

$$I(T) = -k \sum_{n} \left(\sum_{s_n} s_n \log(s_n) + \sum_{r_n} r_n \log(r_n) \right) \quad (1)$$

where k is a normalization constant, and n is the ngram order. s_n and r_n represent the probability of seeing an ngram of order n against the other ngrams of that type and order, for semantic classes and text runs. For this work, we focus on the line level as it is a natural segmentation level for English text, and save other textual segmentations for future work.

The intuition here is in regular prose, as the order increases the distribution of ngrams in that order will tend to become both more uniform and sparser. Because semi-structured regions obey an implicit schema and also tend to be repeated, their ngrams will occur more frequently than usual, causing their constituent ngrams to violate this uniformity assumption. For example, for a sentence of prose, we would expect $I(T)$ to be lower, as s_n and r_n wil be lower. Contrast this with rows from an inline table, where there are several lines containing runs of numbers. In this case, ngrams for runs of numeric semantic classes and of visual features will be seen more than if only regular prose were observed.

Thus to identify semi-structured lines, we score each line of a candidate document, and we focus our analysis on the highest scoring lines in a document. We now describe the method by which we determine and score line regularity, and then we follow with a preliminary evaluation on how well the method identified semi-structured lines in test documents. We continue with a preliminary effort to define extraction patterns, by using a mix of proximity and edit distance measures using the semantic and text run ngrams, and illustrate a basic method for generating an extraction template from those ngrams. Finally, we conclude with a description of further directions.

2. METHOD

We now describe the steps for scoring lines with unusual regularity. For a document, we do the following for each line: identify semantic classes and text runs, and then collect and sort ngram sequences by length. We then score each line using a normalized self-information score.

2.1 Semantic Class Tagging

For a given text, we assign each token in a text a semantic class. Current semantic classes simply consist of identifying two, three, four, and five digit numbers, digit strings with commas. Tokens where none of the characters are alphanumeric are marked as a distinct class. The rest are simply applied with their part of speech tag, as derived from Stanford CoreNLP [7].

2.2 Text Run Identification

Text runs are a way to capture the rough visual appearance of non-whitespace characters on a line, by identifying what visually should be contiguous chunks of text. A run is described as contiguous span of tokens that is bracketed by newlines, tabs, or at least three whitespace characters. For a set of identified runs, we convert these into codewords, going from left to right. Encoded are the leftside character start offset and the number of characters of the run. To allow for variance in the converted text, the start points and lengths of the runs are binned, with start points at 10 characters, and runs at a length of 20.

2.3 NGram Collection

We identify and collect semantic class ngrams by order, starting at order two and increasing in order until none can be found. In order to winnow down on the number of spurious ngrams, we filter to ngrams that appear in at least one other line. Using line breaks as delimiters, we tabluate all semantic class ngrams of any order that occur at least twice in the document. The same procedure is repeated for the text runs, except we retain all observed ngrams.

2.4 Scoring

In this phase, the self-information of each line of the document is scored, using a combination of the semantic class and text run features, where the probabilities are obtained from set of ngrams found at the given order. We normalize by the number of ngrams in that line, and smooth via Lidstone's Law [3], adding a small weight arbitrarily chosen at $\lambda = 10^{-6}$. To reduce the effect of noise, we only retain ngrams occurring more than two times in the entire document, effectively treating rare ngrams as uniform noise. The procedure for this phase is outlined in Algorithm 1.

3. SETUP

Algorithm 1 Score($document, patterns, runs$)

```
scores ← InitArray(size(document))
for n = 2 → maxN(patterns) do
  for i = 1 → size(document) do
    line ← document[i]
    M ← size(ngrams(line, n))
    matched ← matchedSemClasses(line, patterns[n])
    for m_j ∈ matched do
      p = P(m_j|patterns[n])
      score ← −p log(p)
      scores[i] ← scores[i] + score/M
    end for
  end for
end for
for n = 1 → maxN(runs) do
  for i = 1 → size(document) do
    line ← document[i]
    B ← size(runs(line))
    run ← matchedRuns(line, runs[n])
    for b_j ∈ run do
      p = P(b_j|runs[n])
      score ← −p log(p)
      scores[i] ← scores[i] + score/B
    end for
  end for
end for
```

In order to provide a corpus for conducting an evaluation, we downloaded documents from various online sources, such as labor statistics via fedstats.gov, the FBI's Most Wanted, and various state government and Bureau of Alcohol, Tobacco, Firearms and Explosives (ATF) press releases, for a total of 151 documents. The files consisted of a mix of pure text files, PDFs, and HTML pages. To normalize non-text files into text, we used Apache Tika [3]. An assessment of the documents showed that the labor statistics consisted of paragraph sized text descriptions, followed by tables, encoded in a variety of styles. The FBI Most Wanted consisted entirely of field-value pairs, whereas press releases consisted primarily of freetext, with semi-structured information in the form of contact information in field-value pairs, and long list-like enumerations of properties.

For this evaluation, we focus on the line level, as this is a natural segmentation found in the documents, and save non-line level analyses for future work. For each document in the corpus, we labeled each of its line, indicating whether it was semi-structured or prose. For each of the documents in this set, we apply the above tagging and pattern identification methods to derive a surprise score for each line in that document. Note that at this stage, we have not incorporated any information from other documents, nor from any external corpora. The intent here is to see how well the given method can separate semi-structured lines from regular prose lines.

4. EVALUATION

To evaluate the ability of our scoring method to identify semi-structured lines, we use average precision (AP) the lines sorted by score, over each document, deferring the problem of threshold selection for future work. Results are given in Table 1, listing the mean AP across our document collection. Here, we compare the performance of scoring using the Semantic Class and Text Run features against them individually, using the same scoring setup. For comparison, the score for randomly ordering the document lines was included, where a document AP was computed from the average of 50 trials. Also added was one that orders based on line level character entropy (whitespace included), following the intuition that semi-structured text has less character variety than prose. We note that both the Semantic Class and Run level methods perform better than the baseline methods presented. We note that the spatially motivated Run features gave the most yield in performance, which is not unexpected, as in our experience semi-structured regions tend to exhibit regularity in spatial arrangement. However, it is worth noting that Run level scoring does exhibit a wider standard deviation in score than using Semantic Classes, and incorporating them gives a reduction in variance and increase in mean performance.

[3]http://tika.apache.org

Method	Mean AP	stddev
SemClass+Run	**0.6458**	0.3348
SemClass	0.5534	0.2923
Run	0.5936	0.3578
CharEntropy	0.5072	0.3205
Random	0.3428	0.2043

Table 1: Mean Average Precision for ranking semi-structured lines in documents.

An example of a visual analysis of the scores per line is shown in Figure 1. Here, the score is listed as a bar next to the line, and in the case of the enumeration of locations and dates shown, the self-information score are dramatically higher for the semi-structured enumerations than for the preceding paragraph.

5. PATTERN EXTRACTION

We now describe a preliminary procedure for identifying semi-structured lines and developing extraction templates for them. In the first pass, we identify the contiguous regions of high scoring lines, treating these as semi-structured regions. We then apply simple grammar induction techniques to identify chunks.

For a set of high scoring lines that are in close proximity together, we check the Levenshtein edit distance, in the semantic class sequence space, against surrounding lines. As semi-structured lines follow a schema, their semantic class edit distance should not differ significantly either. From a given seed line, we set an upper and lower bound. For each bound, we increase it until the maximum of the edit distance cross product exceeds a given threshold. The end result is a semi-structured region we consider to operate under a single schema.

A cursory visual inspection of several documents showed relatively clean identification of cut points. This does raise the issue of identifying thresholds for where semi-structured should be, as it is very possible for a document to not have any semi-structured text. This will be addressed in future work.

The next step is to induce an extraction template based off the semantic classes found in the region. Our current approach is to leverage a bank of heuristics about how semantic classes would co-occur, as well as visual cues from the text run information, to infer an extraction pattern. As with [1], the grammars representing the induced schema allow us to identify discrete chunks of information and to align them.

6. CONCLUSIONS AND FUTURE WORK

We have shown preliminary work that can identify semi-structured

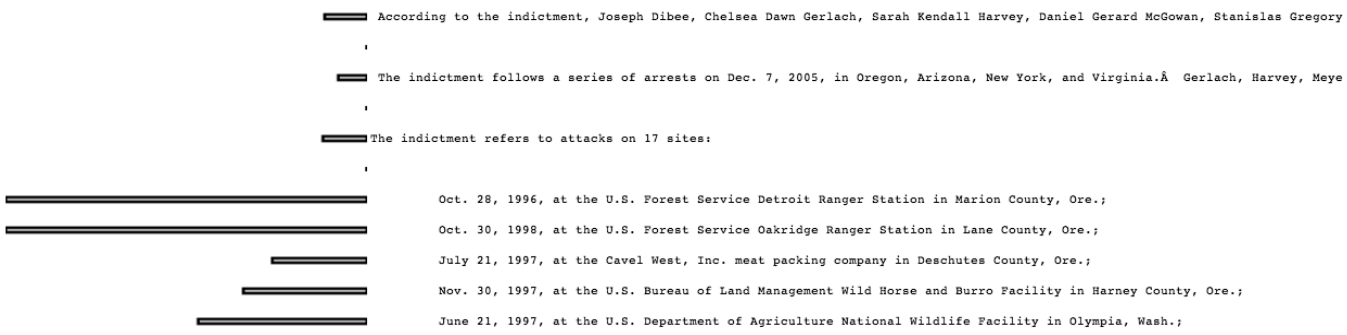

Figure 1: Sample press release, scored self-information displayed by line.

regions from just a document's content itself, in an unsupervised fashion using a self-information measure derived from ground assumptions. The next focus of this effort will be to run this procedure over a corpus of similar documents. We also aim to derive background information in the form of patterns and statistics from corpora, such as Gigaword articles, that are known to be largely prose, as the ngram uniformity in prose can be too gross an assumption. We also plan to develop the extractor system and construct a triple based evaluation for grading the extraction patterns, as used in [4]. We are developing an annotation scheme that permits this type of labeling across the various semi-structured text in our corpus.

This is also very much a "first order" model, and future models will have to address interactions such as cross line or multiple column environments. Another area of investigation are the use of soft semantic class assignments, allowing ambiguity in their membership. This does raise the issue of developing patterns and grammars over vectors of soft assignments. There is also the issue of identifying subject and property assignments given a region and extraction template: we are currently developing an annotation scheme that can work across multiple types of semi-structured text for developing an evaluation, and will explore this further. Finally, techniques that allow discovery of both the extraction templates and identification of semi-structured text are being considered, as the techniques described in the identification and extraction phase share the same principles of identifying regularity.

7. ACKNOWLEDGEMENTS

8. REFERENCES

[1] K. Fisher, D. Walker, K. Q. Zhu, and P. White. From dirt to shovels: Fully automatic tool generation from ad hoc data. In *In POPL*, 2008.

[2] M. Hurst. Layout and language: An efficient algorithm for detecting text blocks based on spatial and linguistic evidence. In *In Document Recognition and Retrieval VIII*, 2001.

[3] C. D. Manning and H. Schütze. *Foundations of statistical natural language processing*. MIT Press, Cambridge Mass., 1999.

[4] D. Pinto, A. Mccallum, X. Wei, and W. B. Croft. Table extraction using conditional random fields, 2003.

[5] A. C. Silva, A. M. Jorge, and L. Torgo. Design of an end-to-end method to extract information from tables. *International Journal Document Analysis Research*, 8:144–171, 2006.

[6] S. Soderland, C. Cardie, and R. Mooney. Learning information extraction rules for semi-structured and free text. In *Machine Learning*, pages 233–272, 1999.

[7] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *IN PROCEEDINGS OF HLT-NAACL*, pages 252–259, 2003.