Segmentation of Publication Records of Authors from the Web

Wei Zhang, Clement Yu
Department of Computer Science
University of Illinois at Chicago
{wzhang,yu}@cs.uic.edu

Neil Smalheiser, Vetle Torvik
Department of Psychiatry
University of Illinois at Chicago
smalheiser@psych.uic.edu, vtorvik@uic.edu

Abstract

Publication records are often found in the authors' personal home pages. If such a record is partitioned into a list of semantic fields of authors, title, date, etc., the unstructured texts can be converted into structured data, which can be used in other applications.

In this paper, we present PEPURS, a publication record segmentation system. It adopts a novel "Split and Merge" strategy. A publication record is split into segments; multiple statistical classifiers compute their likelihoods of belonging to different fields; finally adjacent segments are merged if they belong to the same field. PEPURS introduces the punctuation marks and their neighboring texts as a new feature to distinguish different roles of the marks. PEPURS yields high accuracy scores in experiments.

1. Introduction

The publication records in the researchers' personal home pages contain multiple semantic fields, such as authors, title, and journal, etc. The content of these fields can be used in different tasks, such as building publication databases, helping author disambiguation, tracking author's research activities.

In this paper, we solve the publication record segmentation problem. It is to partition a publication record into coherent fields; each field receives a semantic label. But the fields in the records are not made explicit. The ordering of the fields changes according to individual authors. There is no explicit delimiter between two fields. Some fields are optional while other may appear more than once. The heterogeneous structure of the publication records makes this task non-trivial.

We have two helpful observations. First, the punctuation marks serve as field delimiters (FD), for example, a comma between the author and the article title. But the same mark can also be part of the field data, such as a comma between two authors. Distinguishing the FD marks from the non-FD marks can help us solve the segmentation problem. Second, the data from different fields tends to be more different than that from the same field. For example, in "281-287, Orlando, FL". The first comma is a FD. Its left-hand-side (LHS) text is a page *number* while its right-hand-size (RHS) text is a location *word*. The second comma is part of the location field. Both its LHS and RHS text are words. We can determine the nature of a punctuation mark by examining the difference between its LHS and RHS words.

In this paper, we implement a text segmentation system PEPURS (PErsonal PUblication Record Segmenter) to segment the publication records. We make the following contributions:

- Using a novel "Split and Merge" strategy for the segmentation task.
- Using the punctuation marks and their neighboring text as a novel feature to determine the function of the marks.
- Multiple text features are used together as a novel "multiple-feature sequence" to compute the text-field likelihood.
- The stacked generalization method combines multiple classification results.

2. System Overview

Figure 1 shows the architecture of PEPURS.

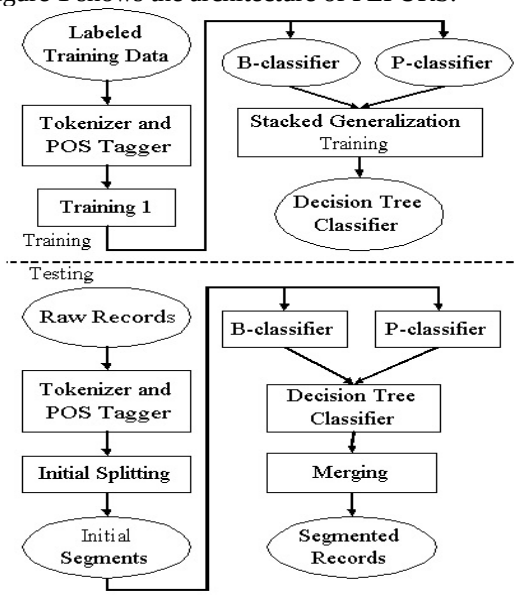

Figure 1. PEPURS System Structure

PEPURS adopts a novel "Split and Merge" framework: (1) Split a raw publication record into text segments according to the punctuation marks that could be used as FDs. Initially we call these marks the undecided marks (UM) as we do not know if they are the real FDs; (2) Compute the likelihood of each segment and each UM belonging to each field by using the distributions of multiple text features; (3) Merge two segments s_1, s_2 and the UM in between to a single segment if the probability of them bearing the same field label is greater than that of s_1 and s_2 having different field labels and the UM having the FD label.

2.1. Getting the Distribution of the Text Features

In the training set, all the fields (FD is also a field) of the records are labeled. A tokenizer and a part-of-speech (POS) tagger also label the words, numbers and punctuation marks in the fields. Thus the distributions of various text features in various fields are collected. The features include the tokens representing the upper-case / lower-case of the words, the POS, the actual words and punctuation marks, and the relative position of the words of different fields in the records. We also collect the paired features. A *pair* refers to two adjacent elements of the same type, such as a pair of words, tokens or POS tags. Keyword lists are collected from the Web for certain fields. Such as DBLP author list, major countries and cities, and organization names.

2.2. Splitting a Record into the Segments

In the "split" step, a publication record is partitioned into initial segments by naively considering that all the instances of the FD marks in this record are real FDs, even some of them are just parts of some fields. We call all these instances the Undecided Marks (UM).

2.3. The B-Classifier

B-classifier computes the probabilities of the initial segments in different fields. It converts a text segment to a "Multiple-Feature Sequence" (MFS). The probabilities of the MFS in various fields are computed in a unigram-bigram combined language model. They approximate the probabilities of the actual segment.

Unigram-Bigram Combined Language Model. The unigram and the bigram are two cases of the n-gram language model [4]. The probability of a string $s=(w_1\ w_2\ w_3\ w_4)$ is a sentence is defined as p(s). In the unigram model, $p(s) = p(w_1) \times p(w_2) \times p(w_3) \times p(w_4)$. In the bigram model, $p(s) = p(w_1) \times p(w_2|w_1) \times p(w_3|w_2) \times p(w_4|w_3)$, which can be written as $p(s) = p(w_1\ w_2) \times p(w_2\ w_3)/p(w_2) \times p(w_3\ w_4)/p(w_3)$ according to the definition of conditional probability $p(w_{i+1}|w_i) = p(w_i\ w_{i+1}) / p(w_i)$. We make an approximation by using the independent events: $p(x\ y) = p(x) \times p(y)$ if x and y never appear as a pair. For example, If $(w_3\ w_4)$ is never found as a pair, $p(s) = p(w_1\ w_2) \times p(w_2\ w_3)/p(w_2) \times p(w_4)$

Multi-Feature Sequence Representation of the Text Segment. If an incoming word is unknown, it will have a probability of 0. Smoothing [7] avoids this by assigning a small value to that word. We modify it to use the token and POS labels of that word to replace the word itself; use the probabilities of the token and POS labels to approximate the probability of the actual word. These probabilities should be better than the small default value. A *Multi-Feature Sequence* (MFS) is designed to replace the original words, in which the unknown words are replaced by their corresponding token and POS labels. The paired elements are distinguished from the single elements in the MFS. Then we compute the probability of a MFS by using the bigram-unigram combined model.

Example 1. W*e have a category C. The training set of C has a word pair of "c d" and the single words of "b" and "e". Given a segment s of "a b c d e f g", its MFS is: "a" as unknown_single, "b" as word_single, "c d" as word_pair, "e" as word_single, "f g" as unknown_pair". The probability of s in C is computed as*

$$
\begin{aligned}
& p(C \mid s) \\
& \propto p(s \mid C) \times p(C) \propto p(MFS|C) \times p(C) \\
& = p(\text{“a” as unknown_single, “b” as word_single,} \\
& \quad \text{“c d” as word_pair, “e” as word_single,} \\
& \quad \text{“f g” as unknown_pair} \mid C) \times p(C) \\
& = \{p(\text{“a” as unknown_single} \mid C) \times \\
& \quad p(\text{“b” as word_ single} \mid C) \times \\
& \quad p(\text{“c d” as word_pair} \mid C) \times \\
& \quad p(\text{“e” as word_ single} \mid C) \times \\
& \quad p(\text{“f g” as unknown_pair} \mid C)\} \times p(C) \\
& = \{[p(token(a) \mid C) + p(POS(a) \mid C)]/2 \times \\
& \quad p(word(b) \mid C) \times \\
& \quad p(word_pair(c\ d) \mid C) \times \\
& \quad p(word(e) \mid C) \times \\
& \quad [p(token_pair(f\ g) \mid C) + p(POS_pair(f\ g) \mid C)]/2\} \\
& \quad \times p(C)
\end{aligned}
$$

2.4. The P-Classifier

The P-classifier uses the position of a segment in the record as the only feature. A record r has m elements $(w_1\ w_2 \ldots w_m)$, the position of w_i, pw_i, is computed as (i/m) in (0, 0.1]. Split the space of (0, 0.1] evenly into ten intervals: (0, 0.1], (0.1, 0.2), …, (0.9, 1]. The pw_i falls into one of them and receives an *interval number*, $intvl_i$, between 0 and 9. w_i is finally represented by $intvl_i$. When

r is partitioned into n segments, assume the i_{th} segment seg_i have the j_{th} word to the k_{th} word.

$$
\begin{aligned}
& p(C \mid seg_i) \\
&= p(C \mid wj \ldots w\,k) = p(wj \ldots w\,k \mid C) \times p(C) \\
&\propto p(wp\,j \ldots wp\,k \mid C) \times p(C) \\
&\propto p(Intvl_j \ldots Intvl_k \mid C) \times p(C)
\end{aligned}
$$

2.5. The Stacked Generalization

Stacked generalization [5, 8] is a general method of combining multiple predictions. We put the B-classifier and P-classifier at level-0 and a decision tree classifier at level-1. The probability outputs of the B- and P-classifier are used to train the decision tree. Given a test segment s, B- and P- classifiers give out the probability values of s in different fields. The decision tree combines these values to a single field label prediction.

2.6. Merging the Initial Segments

An UM m, its LHS segment s_1 and RHS segment s_2 are tested. If the probability of them bearing the same field label is greater than that of s_1 and s_2 having different field labels and the UM having the FD label, they are merged to a single segment. Otherwise they have different field labels. A publication record may have multiple initial segments and UMs, The merging process starts from the first "segment, UM, segment" triple. If they are merged, the new segment replaces the triple. The merging process moves to next available triple. It stops until it reaches the end of the record.

3. Related Work

Hidden Markov Model (HMM) and Viterbi algorithm is a popular combination in the research community. DATAMOLD [3] introduces a hierarchical text feature structure for the record data. Their features are single words and tokens only, not having the POS and the paired features. DATAMOLD has an 87% accuracy score on a Citeseer publication record set. CRAM [1] also adopts the HMM, Viterbi, hierarchical single-element features. It uses a huge database (100,000 records) for training in attempt to replace the human labeled training set. On a test set of the 100 most cited publications in Citeseer, CRAM yields an accuracy score of 91% to 92%.

Human expert knowledge is also used to partition the records. Citeseer [6] adopts human-written heuristics to extract some fields from the whole record. But they only extract author and title fields, because their task is not the record segmentation but a text similarity comparison problem.

The punctuation marks have been ignored by most of the text processing works. However, a rather recent tokenization system BAccHANT [2] showed that the punctuation marks, when used with its left and right neighboring characters, may yield better results than not using them. PEPURS adopts the same idea of using punctuation marks but generalizes the idea of using the left/right neighboring character to the using of left/right neighboring text segment.

4. Conclusions

We evaluated PEPURS with a set of 1674 publication records that were downloaded from the personal home pages of Computer Science researchers. We measured the number of the data that were correctly segmented and labeled. In various 10-fold validation setups, PEPURS had accuracy scores between 92% and 93% when using the whole data set; and scores between 90% and 93% when using half of the set. These scores were higher than the score of 83% from a HMM system. The experimental results show that our "Split and Merge" framework is robust and can be applied to real world data environment.

Acknowledgments

Work supported in part by NIH under grants LM07292 and LM08364. Research funded jointly by the National Library of Medicine and the National Institute of Mental Health.

References

[1] Eugene Agichtein and Venkatesh Ganti, "Mining Reference Tables for Automatic Text Segmentation", In *Proceedings of the ACM SIGKDD*, 2004.

[2] Robert Arens, "A Preliminary Look into the Use of Named Entity Information for Bioscience Text Tokenization", In *Proceedings of HLT/NAACL 2004*: Companion Volume, 2004, pp. 201 - 205.

[3] Vinayak Borkar, Kaustubh Deshmukh and Sunita Sarawagi, "Automatic Segmentation of Text into Structured Records", *ACM SIGMOD*, 2001.

[4] P. F. Brown, V. J. Della Pietra, P. V. deSouza, J. C. Lai and R. L. Mercer, "Class-based n-gram models of natural language", *Computational Linguistics*, 18(4), 1992, pp. 467 – 477.

[5] Melissa K. Carroll and S.-H. Cha, "Application of Stacked Generalization to a Protein Localization Prediction Task", in *Proceedings of 7th JCIS*, Cary, North Carolina, 2003, pp. 13 – 16.

[6] Steve Lawrence, C. Lee Giles and Kurt Bollacker, "Autonomous Citation Matching", In *Proceedings of the Third International Conference on Autonomous Agents*, 1999.

[7] Chris Manning and Hinrich Schütze, "*Foundations of Statistical Natural Language Processing*", MIT Press, 1999.

[8] D. Wolpert, "Stacked generalization", *Neural Networks*, vol. 5, 1992, pp. 241 – 259.