Scholarly Document Information Extraction using Extensible Features for Efficient Higher Order Semi-CRFs

Nguyen Viet Cuong, Muthu Kumar Chandrasekaran, Min-Yen Kan, Wee Sun Lee*
Department of Computer Science, National University of Singapore
{nvcuong,muthu.chandra,kanmy,leews}@comp.nus.edu.sg

ABSTRACT

We address the tasks of recovering bibliographic and document structure metadata from scholarly documents. We leverage higher order semi-Markov conditional random fields to model long-distance label sequences, improving upon the performance of the linear-chain conditional random field model. We introduce the notion of extensible features, which allows the expensive inference process to be simplified through memoization, resulting in lower computational complexity. Our method significantly betters the state-of-the-art on three related scholarly document extraction tasks.

Categories and Subject Descriptors

H.3.7 [**Information Storage and Retrieval**]: Digital Libraries

General Terms

Algorithms, Experimentation

Keywords

Metadata Extraction, Logical Structure Discovery, Conditional Random Fields

1. INTRODUCTION

The publication metadata of a scholarly work and those of its referenced publications form the foundation of citation indices, which enable a variety of digital library services. Accurate extraction, parsing and matching of bibliographic reference strings is needed to properly attribute a work and its components: author, institution and publication venue. The full text also enables extraction of citation context and document structure used in literature review generation, citation function and keyphrase extraction. However, a work's metadata and those of its cited works – along with its logical document structure – remain largely machine-inaccessible as the ubiquitous PDF document format often does not expose this information. Automated extraction is needed to address this class of scholarly document information extraction needs.

Extraction from PDF typically employs optical character recognition to first recover the text and its formatting and layout characteristics. In digital libraries, this input is passed to other systems, such as a conditional random field (CRF), to extract or infer the document metadata. CRF-based systems for such tasks are common in the literature, and fielded in both academic and industrial circles. Mendeley[1], Citeseerχ [1] and ParsCit [2] are notable systems that address reference string parsing using standard linear-chain CRFs. Similar works have also applied them to the logical document extraction task [3, 4, 5]. While it works well, a linear model is limited: it cannot capture non-adjacent dependencies common in such tasks. Stacked linear-chain CRF models have been proposed for the task of reference string parsing [6], in response to this shortcoming. This solution has a cheaper inference cost, but still lacks the expressiveness needed to model long-range dependencies.

To address this shortcoming, we focus on two improvements to CRFs that have been proposed to increase their modeling sophistication and performance. Higher order CRFs capture long-range label patterns, while semi-Markov CRFs (semi-CRFs) model successive labels of the same type as cohesive segments. Both methods individually improve model fidelity, and hence prediction accuracy, but are often computationally expensive, increasing running time (both training and testing) and memory requirements. Our work applies both advances to document information extraction tasks. With such higher order semi-Markov CRFs (HO-SCRFs), we capture long-range dependencies between segments of labels which often occur in scholarly data: *e.g.*, patterns such as *author+ date+ title+*[2] in reference string parsing and *abstract+ introduction+ method+* in document structure labeling.

Our key contribution is to make HO-SCRFs more tractable for practical use. By introducing the notion of *extensible features*, we can distinguish features that can benefit from reusing prior computations (memoization) in the calculation of the potential functions Ψ. Using our technique, we demonstrate overall inference improvements HO-SCRFs make over

* This research is supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office.

JCDL'15, June 21–25, 2015, Knoxville, Tennessee, USA.

ACM 978-1-4503-3594-2/15/06 ...$15.00.
DOI: http://dx.doi.org/10.1145/2756406.2756946.

[1] `http://www.mendeley.com/`
[2] The symbol '+' denotes a segment with one or more consecutive labels of the same class.

Definitions: Let $\mathbf{x} = (x_1, \ldots, x_{|\mathbf{x}|})$ be an input sequence and $\mathcal{Y}$ be a set of labels.
· A *segment* of $\mathbf{x}$ is a triplet (u, v, y) where $1 \leq u \leq v \leq |\mathbf{x}|$ and $y \in \mathcal{Y}$ is the common label of the subsequence $\mathbf{x}_{u:v} = (x_u, \ldots, x_v)$.
· A *segmentation* for $\mathbf{x}$ is a sequence of consecutive segments $\mathbf{s} = (s_1, \ldots, s_p)$, where $s_t = (u_t, v_t, y_t)$ with $u_1 = 1$, $v_p = |\mathbf{x}|$, and $u_{t+1} = v_t + 1$ for all t (i.e., the segments are juxtaposed and cover the whole input).

Figure 1: Segment and segmentation definitions drawn from the semi-CRF literature.

the standard, state-of-the-art linear-chain CRFs on three public scholarly document information extraction tasks.

2. METHOD

We describe the higher order semi-CRFs (HO-SCRFs) with extensible features that are used in our paper. A linear-chain CRF (L-CRF) models the probability of a label sequence for an input sequence. Semi-CRFs [7] extend the L-CRF to model the probability of a sequence of variable-length segments, each of which consists of consecutive tokens with the same label, for an input sequence. This model additionally allows features over segments (as opposed to just tokens), such as aggregate properties of segments like length of fields. For example, in reference string parsing, transitions between two fields such as *author* → *title* can be explicitly specified. HO-SCRFs allow features to be specified over more than two consecutive segments. For example, a 2^{nd}-order semi-CRF can model the transition between three consecutive fields such as *author* → *date* → *title*.

Formally, a semi-CRF [7] models the conditional distribution over all segmentations $\mathbf{s}$ of an input sequence $\mathbf{x}$ by:

$$P(\mathbf{s}|\mathbf{x}) = \frac{1}{Z_{\mathbf{x}}} \exp\left(\sum_{i=1}^{m}\sum_{t=1}^{|\mathbf{s}|} \lambda_i f_i(\mathbf{x}, \mathbf{s}, t)\right),$$

where $\mathcal{F} = \{f_i(\mathbf{x}, \mathbf{s}, t) : 1 \leq i \leq m\}$ is a set of semi-Markov features, each of which has a corresponding weight λ_i, and $Z_{\mathbf{x}}$ is the partition function to normalize $P(\mathbf{s}|\mathbf{x})$ to a proper probability. An HO-SCRF [8] allows the semi-Markov features to have the following form:

$$f_i(\mathbf{x}, \mathbf{s}, t) = \begin{cases} g_i(\mathbf{x}, u_t, v_t) & \text{if } y_{t-|\mathbf{z}^i|+1} \ldots y_t = \mathbf{z}^i \\ 0 & \text{otherwise} \end{cases},$$

where $\mathbf{z}^i$ is the segment label pattern associated with feature f_i, and $\mathbf{s}$ is a segmentation of $\mathbf{x}$. *I.e.*, at segment t, if the label pattern of the segmentation $\mathbf{s}$ matches the segment label pattern $\mathbf{z}^i$, then f_i has the value g_i that depends only on the observation sequence. The feature f_i is said to be a k^{th}-order semi-Markov feature, if the length of the label pattern $\mathbf{z}^i$ is $k + 1$.

We use an example drawn from our digital library datasets to explain the modeling power of such features. In the task of reference string parsing, we can specify a 2^{nd}-order semi-Markov feature which returns the number of times a specific word *Scholarly* appears in a *title* field, when the previous two fields are *author* and *date*, respectively. The segment label pattern $\mathbf{z}^i$ associated with this feature is *author+ date+ title+*. Such *bag-of-words* features can be further generalized as *bag* features that count the number of times a certain property appears in a field rather than simply counting specific word occurrences.

2.1 Extensible Semi-Markov Features

To enable faster inference, we make the key observation that certain higher order features can be incrementally computed – such features can reuse computations made previously for inferring the label of a previous token. We thus partition the feature set into two disjoint sets: the *extensible* features and the *non-extensible* ones. Extensible features are such features; those whose values can be aggregated over tokens within a segment, e.g. the bag features above.

Formally, a feature $f_i \in \mathcal{F}$ is called *extensible* if for any $1 \leq u < v \leq |\mathbf{x}|$, we have:

$$g_i(\mathbf{x}, u, v) = g_i(\mathbf{x}, u, v-1) + h_i(\mathbf{x}, v) = h_i(\mathbf{x}, u) + g_i(\mathbf{x}, u+1, v),$$

where $h_i(\mathbf{x}, u) := g_i(\mathbf{x}, u, u)$ for all u. In other words, a feature is extensible if its value on a segment can be computed from the value at either of the segment's boundary plus the value on the remaining segment. In many cases, h_i is an easy-to-compute function.

Unigram L-CRF features (those depending only on the label of the current token) can often be encoded as bag features in semi-CRFs. Thus, we can include many L-CRF features into HO-SCRFs as extensible features. Examples of extensible features include word counts within segments and lengths of segments.

2.2 Inference Algorithms

Inference algorithms for HO-SCRFs with extensible features are essentially similar to the original algorithms for HO-SCRFs in [8], except for modifications to re-use the computations for extensible features which we detail next.

Recall that inferences for HO-SCRFs require the computation of both forward and backward variables. These variables will be used to compute the partition function $Z_{\mathbf{x}}$, the expected feature sum, and the marginal probabilities. During testing, the most likely segmentation for a given input sequence is computed using the Viterbi algorithm.

A key step in the inferences is to compute the factor $\Psi_{\mathbf{x}}(u, v, \mathbf{p}) = \exp(\sum_{i:\mathbf{z}^i \leq^s \mathbf{p}} \lambda_i g_i(\mathbf{x}, u, v))$, where $\mathbf{p}$ is a sequence of segment labels and $\leq^s$ is the suffix relation. This factor gives the total contribution within the subsequence $\mathbf{x}_{u:v}$ of all the activated features that match the segment label sequence $\mathbf{p}$. We can factorize $\Psi_{\mathbf{x}}(u, v, \mathbf{p})$ into $\Psi^e_{\mathbf{x}}(u, v, \mathbf{p}) \times \Psi^n_{\mathbf{x}}(u, v, \mathbf{p})$ such that:

$$\Psi^e_{\mathbf{x}}(u, v, \mathbf{p}) = \exp(\textstyle\sum_{i:f_i \in \mathcal{F}_e \wedge \mathbf{z}^i \leq^s \mathbf{p}} \lambda_i g_i(\mathbf{x}, u, v)), \text{ and}$$
$$\Psi^n_{\mathbf{x}}(u, v, \mathbf{p}) = \exp(\textstyle\sum_{i:f_i \in \mathcal{F}_n \wedge \mathbf{z}^i \leq^s \mathbf{p}} \lambda_i g_i(\mathbf{x}, u, v)),$$

where $\Psi^e_{\mathbf{x}}$ only aggregates over extensible features while $\Psi^n_{\mathbf{x}}$ aggregates over non-extensible features.

If f_i is an extensible feature, we can decompose g_i into $g_i(\mathbf{x}, u, v) = h_i(\mathbf{x}, u) + g_i(\mathbf{x}, u+1, v)$. With such decomposition and the above factorization, we can write:

$$\Psi_{\mathbf{x}}(u, v, \mathbf{p}) = \exp(\sum_{\substack{i:f_i \in \mathcal{F}_e \\ \mathbf{z}^i \leq^s \mathbf{p}}} \lambda_i h_i(\mathbf{x}, u)) \Psi^e_{\mathbf{x}}(u+1, v, \mathbf{p}) \Psi^n_{\mathbf{x}}(u, v, \mathbf{p}).$$

Here, $\Psi^e_{\mathbf{x}}(u+1, v, \mathbf{p})$ is the computation that is memoized previously. We then only need to calculate the incremental value of $\Psi^n_{\mathbf{x}}(u, v, \mathbf{p})$ and the exponential factor to get the new value of $\Psi_{\mathbf{x}}(u, v, \mathbf{p})$. Hence, using this formula and a simple rearrangement for the recurrence to compute the forward variable (in Section 2.3.1 of [8]), we can achieve a speedup in the forward inference as the value of $\Psi^e_{\mathbf{x}}(u+1, v, \mathbf{p})$ is

memoized when we compute $\Psi_{\mathbf{x}}(u, v, \mathbf{p})$ (i.e., dynamic programming can be used to compute the values of $\Psi_{\mathbf{x}}$).

This memoization based speed-up is applicable to many parts of the inference pipeline. Aside from the forward variable computation, the speedup also applies to the backward variables (by decomposing g_i similarly into $g_i(\mathbf{x}, u, v) = h_i(\mathbf{x}, v) + g_i(\mathbf{x}, u, v-1)$), computation of marginal probabilities, and for the Viterbi algorithm during testing. To be clear, this computational shortcut applies to both HO-SCRFs and normal semi-CRFs.

2.3 Computational Complexity

The complexity analysis for the original inferences in [8] assumed that the features $g_i(\mathbf{x}, u, v)$ can be computed in $O(1)$, an unrealistic assumption since the computation of such segment features depends on the length of the segment. In this paper, we assume the computation of $h_i(\mathbf{x}, v)$ is $O(1)$ instead, thus making the computation of g_i linear in the segment length, which we feel is more realistic.

For simplicity, we also assume that all the values of $\Psi_{\mathbf{x}}$ (equivalently, $\Psi^e_{\mathbf{x}}$ and $\Psi^n_{\mathbf{x}}$) are pre-computed before we compute the forward and backward variables. In practice, $\Psi_{\mathbf{x}}$ is computed on-the-fly using memoization. Note that computing $\Psi_{\mathbf{x}}$'s is the computational bottleneck in CRF inference, and that our strategy directly decreases its computational complexity.

Indeed, if we do not leverage extensibility, the worst-case time complexity to pre-compute $\Psi_{\mathbf{x}}$'s is $O(T^3|\mathcal{F}||\mathcal{P}||\mathcal{Y}|^2) = O(T^3(|\mathcal{F}_e| + |\mathcal{F}_n|)|\mathcal{P}||\mathcal{Y}|^2)$, where T is the maximal input sequence length and $\mathcal{P}$ is the forward-state set [8]. In contrast, when we use extensibility, this worst-case complexity is $O((|\mathcal{F}_n|T^3 + |\mathcal{F}_e|T^2)|\mathcal{P}||\mathcal{Y}|^2)$. Since most ordinary features are extensible, the bulk of the inference changes from cubic to quadratic complexity in T, a large savings. Once we have all the values of $\Psi_{\mathbf{x}}$, the time complexity for other inference steps remain identical.

3. EVALUATION

To validate the performance of HO-SCRFs, we re-use three scholarly extraction tasks which have previously been formally defined and which have freely-available datasets.[3]

· **Reference String Parsing** is the task of tokenizing and labeling individuals fields of reference strings (i.e., the bibiography) of a scholarly work. Given a reference string as input, the models should appropriately label its tokens. The set of labels are listed in Column 1 of Table 2. We re-use the exact features and values described in [2] for direct comparison.

· **Generic Section Labeling** seeks to recover the logical structure of the main sections of a scholarly work. Models are tasked to label the section sequence of an input work, picking labels from Column 1 of Table 3. We re-use the exact features and their values as described in [4].

[3]We note that the reference string parsing dataset is a compendium of Cora, FLUX-CiM, ICONIP and humanities datasets. Section labeling is evaluated on the generic section labeling dataset used in [4]. Author and Affiliation extraction is evaluated on the compendium of works published by the ACL and from a cross-domain dataset reported in [5]. All datasets are publicly available at `https://github.com/knmnyn/ParsCit/tree/master/crfpp/traindata`.

Table 1: Statistics of the datasets used.

Task [Citation]	Train	Validation	Test
Reference string parsing [2]	883	–	501
Generic section labeling [4]	102	44	65
Author extraction [5]	13	10	144
Affiliation extraction [5]	6	10	145

Table 2: F_1 (%) for HO-SCRFs on the reference string parsing task.

Label (Size)	L-CRF	1SCRF	2SCRF	3SCRF
author (2085)	99.00	98.97	99.02	98.78
booktitle (3116)	93.60	93.67	94.15	93.71
date (671)	93.61	92.98	93.26	93.11
editor (207)	75.33	71.54	75.60	75.60
institution (27)	79.17	79.17	79.17	96.43
journal (451)	89.31	89.60	90.12	88.22
location (408)	89.20	89.18	89.91	90.68
note (69)	57.14	57.14	57.53	60.00
pages (580)	95.91	95.59	94.56	95.49
publisher (125)	83.33	83.68	83.33	84.39
tech (5)	46.15	46.15	46.15	62.50
title (4086)	94.53	94.74	95.35	95.22
volume (154)	91.28	92.20	87.74	90.00
Micro-average	**94.01**	**94.01**	**94.35****	**94.26****

· **Author and Affiliation Extraction** is to extract author (affiliation) occurrences from the author (affiliation) lines in the header section of a paper [5]. These lines also contain other markers (symbols) and other separators to delimit multiple authors (affiliations). Models are to assign labels from the inventory of *author* (*affiliation*), *symbol*, and *separator*.

For ease of reference, we restate the demographics of the public datasets we used in Table 1. Except for reference string parsing, we use a validation set to select the best regularization parameter σ among the values 0.1, 1, 10, 100. The models with the best parameter settings are then trained on the union of the train and validation sets, and then tested on the test set. For reference string parsing, we only use the default $\sigma = 1$ without applying validation, due to the long training time.

Results. In what follows, L-CRF, 1SCRF, 2SCRF and 3SCRF denote the linear-chain CRF (our baseline), 1^{st}-, 2^{nd}- and 3^{rd}-order semi-CRF models, respectively. In the prior literature, only L-CRFs were used to address these tasks. As input for 1SCRF, we use all the features of the L-CRF together with all 1^{st}-order semi-Markov transition features. For both the 2SCRF and 3SCRF models, we re-use all features for the $k-1^{th}$ model and in addition incorporate the k^{th}-order semi-Markov transition features.

For the reference string parsing task, from Table 2, 2SCRF and 3SCRF perform significantly better than L-CRF ($p < 0.01$) in aggregate. Overall, 2SCRF achieves the best score (94.35%), and it performs equally or better than L-CRF on 10 out of 13 labels, including some dominant labels such as *title*, *booktitle*, and *author*. Many errors for this task come from the humanities datasets, which contain non-English references. Other errors come from the ambiguity of the labels, some of which even confused human annotators. For instance, the token *16(3):52-55* was labeled as *volume* by 2SCRF but its true label is *pages* (the token contains both the volume and page information).

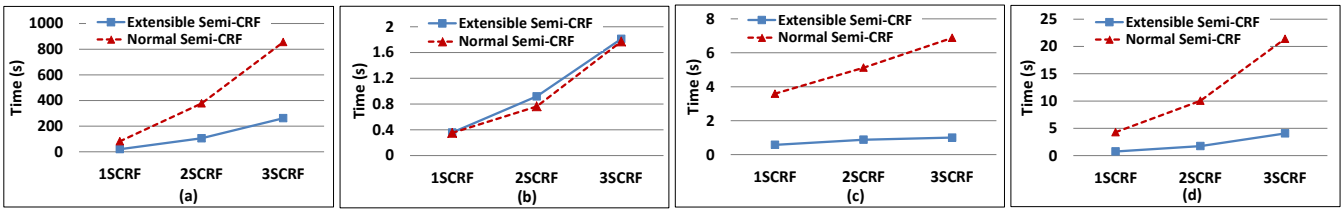

Figure 2: Elapsed running time of the standard and our extensible higher order semi-CRFs for the (a) reference string parsing, (b) generic section labeling, (c) affiliation extraction, and (d) author extraction tasks on their respective test sets (or 25% of the test set for reference string parsing).

Table 3: F_1 (%) for HO-SCRFs on the generic section labeling task.

Label (Size)	L-CRF	1SCRF	2SCRF	3SCRF
abstract (65)	100	100	100	100
acknowledgement (29)	100	94.92	94.92	93.10
background (8)	80.00	85.71	94.12	100
categories (50)	99.01	99.01	99.01	99.01
conclusions (55)	92.98	92.31	94.74	92.17
discussions (15)	72.00	50.00	80.00	50.00
evaluation (42)	89.74	81.08	92.31	83.95
general terms (46)	100	100	100	100
introduction (64)	97.67	97.67	97.67	97.67
methodology (183)	96.24	93.96	97.56	95.70
references (65)	100	100	100	100
related works (30)	100	94.74	100	100
Micro-average	**96.63**	**94.79**	**97.39***	**95.71**

Table 4: F_1 (%) for HO-SCRFs on the author and affiliation extraction tasks.

Problem	L-CRF	1SCRF	2SCRF	3SCRF
Author	93.64	93.53	94.06*	93.21
Affiliation	98.33	98.50	98.50	98.50

For generic section labeling, Table 3 shows that 2SCRF performs significantly better than L-CRF ($p < 0.05$) in aggregate. 2SCRF achieves the best score overall (97.39%), performing equally or better than L-CRF on 11 of 12 categories, inclusive of dominant categories such as *methodology* and *conclusions*. Errors for this task include confusion between the *conclusions* section and the *evaluation* or *discussion* sections in the data. For example, 2SCRF may predict that there is a conclusions section before the references section in a paper, while the correct label is *discussion*.

On the author extraction task, Table 4 shows that 2SCRF performs significantly better than L-CRF in aggregate ($p < 0.05$) and achieves the best F_1 score (94.06%). For the affiliation extraction task, all of the semi-CRF configurations perform better than L-CRF with $p < 0.057$. Many errors for these tasks, especially the author extraction task, come from the *separator* class.

Running Time. For all tasks, we measured the testing time on the test data sets[4] using a 24-core machine (800 MHz per core; Figure 2). Similar trends also occurred in training times. For the reference string parsing and author–affiliation extraction tasks, Figure 2a,c,d shows that leveraging the extensible property of the semi-Markov features improves the running time significantly. Generic section labeling task is the exception (Figure 2b). One possible explanation is that the test set for this task is small (65 instances), and the running time is dominated by the parallel communication between the threads (which is non-deterministic).

[4]We used 25% of the test set for reference string parsing.

4. CONCLUSION

We have demonstrated the feasibility of using higher order semi-CRFs (here, HO-SCRFs) to improve performance on scholarly document extraction tasks. By noting that many semi-CRF features are extensible (amenable to incremental calculation and hence memoization), we can also efficiently train and test such models.

We note that learning solutions are not the only solution to scholarly document information extraction tasks. For reference string parsing, sourcing scholarly metadata through external sources (*i.e.*, the Web) is a promising avenue [9, 10]. In future work, we plan to integrate both learning and Web lookup (when appropriate) to solve such tasks.

5. REFERENCES

[1] C. Lee Giles, Kurt D. Bollacker, and Steve Lawrence. Citeseer: An automatic citation indexing system. In *DL*, pages 89–98, 1998.

[2] Isaac G. Councill, C. Lee Giles, and Min-Yen Kan. ParsCit: An open-source CRF reference string parsing package. In *LREC*, 2008.

[3] Alan Souza, Viviane Moreira, and Carlos Heuser. ARCTIC: metadata extraction from scientific papers in pdf using two-layer CRF. In *DocEng*, 2014.

[4] Minh-Thang Luong, Thuy Dung Nguyen, and Min-Yen Kan. Logical structure recovery in scholarly articles with rich document features. *IJDLS*, 1(4):1–23, 2010.

[5] Huy Hoang Nhat Do, Muthu Kumar Chandrasekaran, Philip S. Cho, and Min Yen Kan. Extracting and matching authors and affiliations in scholarly documents. In *JCDL*, pages 219–228, 2013.

[6] Peter Kluegl, Martin Toepfer, Florian Lemmerich, Andreas Hotho, and Frank Puppe. Stacked conditional random fields exploiting structural consistencies. In *ICPRAM*, 2012.

[7] Sunita Sarawagi and William W. Cohen. Semi-Markov conditional random fields for information extraction. In *NIPS*, pages 1185–1192, 2004.

[8] Nguyen Viet Cuong, Nan Ye, Wee Sun Lee, and Hai Leong Chieu. Conditional random field with high-order dependencies for sequence labeling and segmentation. *JLMR*, 15:981–1009, 2014.

[9] Liangcai Gao, Xixi Qi, Zhi Tang, Xiaofan Lin, and Ying Liu. Web-based citation parsing, correction and augmentation. In *JCDL*, pages 295–304, 2012.

[10] Dat T. Huynh and Wen Hua. Self-supervised learning approach for extracting citation information on the web. In *Web Technologies and Applications*. 2012.