EXTRACT AND MATCH IN CITATION EXTRACTION:
A COMPARATIVE STUDY ON CONSTRAINT USAGE

BY
GUANGYU ZHOU

THESIS
Submitted in partial fulfillment of the requirements
for the degree of Bachelor of Science in Computer Science
in the Engineering College of the
University of Illinois at Urbana-Champaign, 2016

Urbana, Illinois

Advisor
Professor Kevin C. Chang
Abstract

Information extraction is the task of automatically extracting structured information from unstructured data. Among many techniques to solve the problem, Conditional Random Fields (CRF) is the state-of-the-art. However, CRF requires local dependencies, i.e., Markov property, to enable efficient inference via Viterbi algorithm. To relax the requirement, long-range dependencies are usually expressed as constraints to be enforced during the inference. As the long-range dependencies violate the Markov property, previous work has been focusing on replacing Viterbi algorithm by approximate solutions that can incorporate non-local structure while preserving tractable inference. In this work, we enforce constraints by a different approach, i.e., using a modified version of Viterbi, which can incorporate most types of constraints users need. The trade-off of such exact enforcement is the additional cost compared to the original Viterbi algorithm. We carry an extensive study of the complexity to allow users to know the cost before doing the extraction. In particular, we use the context of the web citation extraction problem to demonstrate our constraint enforcement technique, result, and analysis.
# Table of Contents

Chapter 1  Introduction .......................................................... 1

Chapter 2  Related work ......................................................... 3

Chapter 3  General framework ................................................ 4
  3.1  Opportunity: From “Matching” to “Extract” knowledge ............... 4
  3.2  Challenge: Finding the impactful constraints .......................... 4

Chapter 4  Citation Extraction .................................................. 6
  4.1  Problem statement .......................................................... 6
  4.2  Models ................................................................. 6
    4.2.1  Introduction to CRF model ...................................... 7
    4.2.2  Features vs. constraints ......................................... 7
    4.2.3  Inference by Constraints enforcement .......................... 7

Chapter 5  Techniques ............................................................ 8
  5.1  The definitions of local vs. non-local constraints .................... 8
  5.2  The schema of categorizing and generating constraints .............. 9
  5.3  The introduction of the segment-based Viterbi model ............... 10
  5.4  Adapting the segment-based Viterbi to more general constraints ... 10

Chapter 6  Results ............................................................... 12
  6.1  The constraint template vs. time complexity table .................. 12
    6.1.1  Single field constraints ....................................... 12
    6.1.2  Multiple fields constraints .................................... 13
  6.2  Argument about the essence of the constraint cost .................. 14

Chapter 7  Conclusions and Future work ................................... 15

References ................................................................. 16
Chapter 1

Introduction

Recent year has witnessed unprecedented proliferation of data, with the largest portion encoded in natural language and therefore unstructured. One sub-field of information extraction is Named Entity Recognition (NER). Different from traditional classification, NER encoded a hidden state sequential structure where the label sequence of word is latent and both word and tags have dependency upon others. Conditional Random Fields (CRF) has been applied mostly to model such NER problem [7, 8], [9, 10], [16, 19]. CRF allows both discriminative training and the bi-directional flow of probabilistic information across the sequence, which makes it a state-of-the-art tool.

However, research indicates that even state-of-the-art NER systems (CRF) are brittle, meaning that NER systems developed for one domain do not typically perform well on other domains. Even in one specific domain of NER, the performance cannot be guaranteed among different domains with inconsistency of the format or rules of writing. Lots of relevant work has been done to extend the CRF model by injecting the domain knowledge as the form of constraint enforcement. Most non-local constraints are hard to be directly captured by the CRF model due to the Markov property. To facilitate users to add general (local and non-local) constraints over the output space in a natural and systematic fashion, they provide different approximate inference procedures such as Gibbs sampling or beam search. Their approaches, though can be extremely fast by reducing the search space, may result in local optima and sacrifice the effectiveness.

We specifically focus on the Citation Extraction problem: Given unstructured publication data from personal webpages, we would like to use NER tools to structure them into predefined fields (e.g. Author, Title, Venue, Year, etc.) so that the structured data can be better indexed. The opportunities of our problem lay in the consistent patterns (i.e. Match) of the publications sources. To leverage such “Matching” knowledge, we use constraint enforcement. However, the challenge of such approach is how to select useful constraints from instantiated constraint candidates with affordable cost.

To address the challenge, in this work, we approach the constraints enforcement from a different perspective: We adapt the extended version of (segment-based) Viterbi that can incorporate most types of constraints users need. More specifically, we conduct a comparative analysis between local and non-local
constraints. Our objective is to achieve similar performance in citation extraction problem without incurring the cost of enforcing expensive non-local constraints. We formalize our problem into two sub-directions: 1) use cheaper local constraints to express expensive non-local constraints; 2) enforce expensive non-local constraints based on a cheaper model (Viterbi). While the first direction is lack of systematic argument, the experiment results give us guidance on the second direction.

In the following sections, we will present the related work. Then, we will describe the extract and match problem as a general framework, after which the technique of non-local constraint enforcement by extended Viterbi will be presented. Finally, a comprehensive study of the complexity under different types of constraints will be provided to user as guidelines before doing extraction and select constraints.
Chapter 2

Related work

Global constraint has been proved to be an inherent property in sequence labeling task including citation extraction. Several approaches have been proposed to enlarge the information set exploited by incorporating global constraints during training and inference phase in the past years. Two main particular research directions include: (1) relaxing the Markov assumption [1, 18] to model long distance relationships and (2) introducing additional domain knowledge as constraints during the inference phase [4, 5, 14, 17].

The efficiency of Linear Chain Conditional Random Fields (CRF) is strictly related to the underlying Markov assumption: given the observation of a token, the corresponding hidden state (label) depends only on the labels of its adjacent tokens. The first direction, relaxing the Markov assumption, implies an increasing computational complexity in both training and inference phase.

Recent work has been focusing on the second direction. The common theme in this direction is how to efficiently incorporate global constraint as a way of representing domain knowledge. While the efficient Viterbi algorithm in CRF inference can incorporate some types of local constraints (e.g., constrained Viterbi described in [14]) by manipulating the matrix, this matrix modification mechanism cannot be applied when the constraints define the relation of two distant tokens [17]. To reduce the cost of incorporating such global information, most constraint enforcement models, therefore, take the heuristic approach and thus enhance the efficiency of capturing non-local dependencies.

One of the most relevant work is Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling [11]. To account for the long distance structure that is prevalent in language use, they leverage the Monte Carlo method which is used to perform approximate inference in factored probabilistic models. By using simulated annealing in place of Viterbi decoding in sequence models including CRF, it is possible to incorporate non-local structure while preserving tractable inference.

The other relevant series of work [5, 6, 17] have studied models that incorporate learned models with declarative constraints. They formulate the inference process as a constrained optimization problem (ILP). They provide a solution called “beam search” to find the approximate solution with low cost.
Chapter 3

General framework

3.1 Opportunity: From “Matching” to “Extract” knowledge

There have been many systems developed for automatically integrating unstructured text into a multi-relational database using state-of-the-art statistical models for structure extraction and matching. Those systems, aiming at solving the extract and matching problem in general, lack the awareness of the specificity of publication data extraction from personal webpage.

The opportunities of our problem come from the context of the publications sources. The citations under the same professors page should have some consistency and follow some consistent patterns. For instances: 1) They have some common fields; 2) They follow similar order; 3) They consist of high frequent tokens such as same author names appears in every citation.

To leverage the opportunity, we have tried several methods along the way. The first approach we come up with is to use multiple extractors with independent error to generate multiple labeled results. Then, we select a set of labeled tokens with high accuracy as seeds to bootstrap the model. Among consistent result, we need to avoid consistent errors (all False comparing to ground truth) by using extractors with initially independent errors. Otherwise, instead of the knowledge, the error will be propagated. However, there is not a systematic way to generate many independent errors extractors, and thus we proceed to the second method by using constraint enforcement. Refer to Fig 3.1 In the second method, we leverage statistical information across citations to select the most consistency knowledge and inject them as constraints. The benefit of this approach, however, is not significant since the constraints with high weights are already frequent with high support in the sources and may not serve as useful guidance at all.

3.2 Challenge: Finding the impactful constraints

The challenge is how to define or select those constraints that really have impact on the test datasets. Our current approach measures the consistency as prior knowledge by using constraints in the extraction models.
Specifically, we will give user some guidelines to design constraint template and then instantiate them. The issue of current approach is that to select the most useful constraints from the candidates by calculating the score of each constraint, we have to be able to afford the cost of enforcing all those constraints. As described in the introduction as well as related work, for those non-local constraints that violate the Markov property, even Gibbs sampling requires high cost (big number of iterations) to achieve decent quality. Gibbs sampling can be fast with a trade-off for the performance of results. It thus makes the enumerating and searching for the best candidate among all constraints we instantiated not feasible with limited computational sources. Therefore, the main focus of the thesis is to answer two questions: 1) what is the key difference between local vs. non-local constraint that make non-local constraint expensive? 2) Can we enforce them efficiently as local constraints?
Chapter 4

Citation Extraction

4.1 Problem statement

We want to identify fields from citations from personal web page. Given a citation, the task is to extract the fields that appear in the given reference. Here is an example: The labels are annotated by underline and are to the left of each bracket.

(a) [TITLE Incremental and Accuracy-Aware Personalized PageRank through Scheduled Approximation. F. Zhu,] [AUTHOR Y. Fang, K. C.-C. Chang, and J. Ying. ] [JOURNAL PVLDB] [VOL 6(6),] [DATE 2013.] [BOOKTITLE In VLDB 2013.]

(b) [TITLE Incremental and Accuracy-Aware Personalized PageRank through Scheduled Approximation. ] [AUTHOR F. Zhu, Y. Fang, K. C.-C. Chang, and J. Ying. ] [JOURNAL PVLDB] [VOL 6(6),] [DATE 2013.] [BOOKTITLE In VLDB 2013.]

Figure 4.1: A example for citation extraction and constraint enforcement

Now, with the trained CRF model, the constraint enforcement can help to improve the result. As presented, Fig 4.1(a) shows the result outputted by CRF while Fig 4.1(b) shows the corrected result. If we enforce the constraint “title field must end with dot”, then the error can be fixed and the same model will result in the correct one.

4.2 Models

We compared two models: HMMs and CRF for the sequence-labeling task of citation extraction. We choose to use CRF because HMMs is a generative model, which has difficulty of modeling overlapping and non-independent features while CRF are discriminative graphical models that can model these overlapping, non-independent features.
4.2.1 Introduction to CRF model

A (linear chain) Conditional Random Field [15] defines a conditional probability of label sequence $Y$ given observation sequence $X$. The scalar $\lambda_i$ is the weight for feature function $f_i()$, which are parameters that should be learned in training phase. The scalar $Z$ is a normalization factor, $n$ refers to word position in the sequence and $f_i()$ are weighted features.

$$P(Y|X) = \frac{1}{Z} \left( \sum_{n=1}^{N} \sum_{i=1}^{F} \lambda_i f_i(y_{n-1}, y_n, X, n) \right)$$

In terms of inference process, the CRF will find the best label sequence by using Viterbi algorithm that maximize this conditional probability: $Y^* = \text{argmax}_Y P(Y|X)$.

4.2.2 Features vs. constraints

Many other recent publications [2, 3, 5, 12, 13, 17] have demonstrated the importance of separating features and constraints. There are two most important reasons of such separation:

1. The constraints term is different from the features term because it can be used to enforce hard constraints by setting the weight of constraint to be infinity.

2. The benefit of adding constraints to existing models is partly due to the fact that constraints can be a lot more expressive than the features used in the existing models.

4.2.3 Inference by Constraints enforcement

We borrow the idea of Constrained Conditional Random Fields from [14]. Since the CRF model leverages the Markov property assumption to avoid an exponentially large search space, its inference Viterbi keep track of the path with highest probability at time $t$ which accounts for the $t$ previous observations and ends in state $s_t$. As such, the induction step of the Viterbi is defined as

$$\delta_{t+1}(S_i) = \max_{s'}[\delta_t(s') \exp(\sum_k \lambda_k f_k(s', s_i, x, t))]$$

To enforce the constraints, the Viterbi in the above Equation is altered such that $s^*$ is constrained to pass through some subpath $C = <s_t, s_{t+1}, ...>$. These constraints $C$ redefine the new induction as below:

$$\delta_{t+1}(S_i) = \begin{cases} 
\max_{s'}[\delta_t(s') \exp(\sum_k \lambda_k f_k(s', s_i, x, t))] & \text{if } s_i = s_{t+1} \\
0 & \text{otherwise}
\end{cases}$$
Chapter 5

Techniques

In this section, we discuss our main contribution: The segment-base Viterbi model to enforce the non-local constraints. We start with a theoretical definition for local and non-local constraints based on the CRF model. Then we introduce the schema of categorizing as well as the way of generating different types of non-local constraints. We then define our segment-base model by showing the computational formulas. With that model, we will show examples to adapt the model to enforce different types of non-local constraints. Then we present a series of table to illustrate the time complexity for different types of non-local constraints. Finally, we argue the essence of different types of constraints and end up with a new categorization based on their cost.

5.1 The definitions of local vs. non-local constraints

According to the feature function definition, we define the constraints as below:

**Definition 1** Local constraint: \( c_i(y_{n-1}, y_n, x_{1:N}, n) \), takes a pair of adjacent states \( y_{n-1}, y_n \), the whole input sequence \( x_{1:N} \), and where we are in the sequence at \( n \).

**Definition 2** Global constraint: \( c_i(Y_k, x_{1:N}) \), takes whole input sequence and a set of states where \( Y_k \) is a subset of \( y_1, \ldots, y_n \) with no limitations that they have to be adjacent.

Here are examples of how we express real constraints into the above definition:

1. Local constraint:
   a) Chang is author: \( c_i(y_{n-1}, y_n, x_{1:N}, n) = 1 \) if \( x_n = \text{Chang} \) and \( y_n = \text{author} \)
   b) Starting of Volume must be adjacent next to the end of journal \( c_i(y_{n-1}, y_n, x_{1:N}, n) = 1 \) if \( y_n = \text{volume} \) and \( y_{n-1} = \text{journal} \)

2. Global Constraint:
   a) Title is before author \( c_i(Y_k, x_{1:N}) = 1 \) if \( Y_k = \{Y_{k1}, Y_{k2}\} \), \( Y_{k1} = y_{[n,n+1,...,n+k1]} = \text{title} \), \( Y_{k2} = y_{[m,m+1,...,m+k2]} = \text{author} \) and \( n + k1 \leq m \)
b) Author, (title, booktitle) must appear once and only once. \( c_i(Y_k, x_{1:N}) = 1 \) if \( Y_k = y_{[n,n+1,...,m]} \) = author and \( y_i \neq \) author if \( i < n \) or \( i > m \)

5.2 The schema of categorizing and generating constraints

The above definitions hint us the potential opportunity to enforce non-local constraints efficiently expressed by local constraint. To conduct such comparative analysis, we need a comprehensive list of useful non-local constraints.

The following schema is developed to help us categorize and show that list of non-local constraints. The insight of the scheme design is from how to generate a field in citation.

For generating a single field, we may have the following dimensions:

1. **Quantitative** First, we need to specify whether the field exist or how many times does it appear.

2. **Position** Then, we want to specify where the field is located.

3. **Writing rules (Format, content)** Within the field, what are the inner rules.

Then we show that the categorization schema is principle since it is from the designing view of users in our setting. In our specific setting: web page citation extraction, we have the following properties for them.

**Property 1:** The quantity of a field should belong to a limited range \([0, 2]\).

The quantity of a field specifies whether the field exist or how many times does it appear. In a real citation, mostly there will be once or at most twice for a field. Or some fields may not exist at all. Thus, specifying the number of times or the existence of a field is the first type of constraints that the user want to have.

**Property 2:** The possible position of a field should belong to a limited range.

The position of a field refers to where this field is located among all the segments. Since there are usually less than 15 possible labels, the possible positions should be limited in a range. In our setting, it can be assumed that all citations from single page should have the consistent positioning of a field.

**Property 3:** All the writing rules for a single field can be checked with \(O(1)\) cost.

The writing rules dimension has more diversity: it can be Format (Capitalize, word shape, alphabetic order, initials, italicize, delimiter, etc.) and Content (Contain specific word or a list of words).

In fact we can treat all such constraints as Boolean constraints, which means that given a segment of tokens, the predicted label and such Boolean constraint: \((x, i, l, k, C_b)\), it can output the Satisfiability at once with \(O(1)\) cost.
5.3 The introduction of the segment-based Viterbi model

Here we define our extended segment-based Viterbi Inference model.

**Definition 3** Segment-based Viterbi Inference model: It takes additional constraints and enforce them as hard constraints during the inference. This model will use dynamic programming techniue to store and compute the extraction score in the matrix.

Here is the recursive formula:

\[ F(i, l, k) = \begin{cases} 
0 & \text{if } i < 0, \forall l, \forall k \\
-\infty & \text{if segment}(i, l, k) \text{ does not satisfy } C \\
\max_{l_p = 1 \rightarrow L, k_p = 1 \rightarrow K, k_p \neq k} [F(i - l_p, l_p, k_p) + f(i, k_p, k)] + \\
\sum_{j = i+1 \rightarrow i+l-1} f(j, k, k) & \text{otherwise}
\end{cases} \]

The meaning of the variables are explained here:

- \( F(i, l, k) \): extraction score if segment beginning at token i and having length l has label k
- \( f(i, k_p, k) \): score received if token i is labeled with k given token i-1 is labeled with \( k_p \)
- \( n \): length of the sequence
- \( L \): max length of a field
- \( K \): max number of labels
- \( C \): a set of predefined hard segment-based constraints, e.g., all titles must end with dot, all authors must contain Kevin, etc., such that all constraints in C can be evaluated on a segment, i.e., a continuous list of tokens that have the same label k, and the the previous and after tokens have different labels with k. Termination condition: \( F(n, 1, \text{EOL}) \): extraction score if segment beginning at token n, having length 1 has label End-Of-Line.

The above algorithm yield the complexity of \( O(n \times L \times K) \) where for regular Viterbi, the complexity is only \( O(n \times K) \).

5.4 Adapting the segment-based Viterbi to more general constraints

While the above formula can enforce some non-local constraints, our goal is to enforce all types of non-local ones. It is not feasible to enforce quantitative constraints like “Author must appear once” in the model...
above. This section, our extended model is presented.

**Extension: For quantitative constraints**

Our insights come from the fact that we want to have the constraint: Field X must appear \( k \) times. In order to leverage the Markov property s.t. Viterbi can be used to compute efficiently, we want to store one additional variable to keep track of the count of times the field X appears before.

Thus, \( F(i, l, k, c_u) \) means that extraction score if segment beginning at token \( i \) and having length \( l \) has label \( k \) and the label \( k_u \) appears \( c_u \) times before. We want to update \( c_u \) whenever there is a state transition from \( k_p \neq k_u \) to \( k = k_u \). Therefore, we can break it down into two different cases. We still want to enforce it as a constraint by making sure \( c_u \) is bounded by some value \( \theta \). And thus, we can set the score to be \(-\infty\) for \( c_u \) out of bound.

The new recursive formula is shown here with an additional parameter and the computation complexity becomes: \( O(n \times L \times K \times |c_u|) \)

\[
F(i, l, k, c_u) = \begin{cases} 
0 & \text{if } i < 0, \forall l, \forall k \\
-\infty & \text{if } c_j > \theta \\
\max_{l_p=1 \rightarrow L, k_p=1 \rightarrow K, k_p \neq k} \left[ F(i - l_p, l_p, k_p, c_u - 1) + f(i, k_p, k) \right] + \\
\sum_{j=i+1 \rightarrow i+l-1} f(j, k, k) & \text{if } k = k_u \\
\max_{l_p=1 \rightarrow L, k_p=1 \rightarrow K, k_p \neq k} \left[ F(i - l_p, l_p, k_p, c_u) + f(i, k_p, k) \right] + \\
\sum_{j=i+1 \rightarrow i+l-1} f(j, k, k) & \text{otherwise}
\end{cases}
\]
Chapter 6

Results

The above Segment-based Viterbi technique provides with us the key enforce non-local constraints that cannot be enforced using dynamic programing directly before. To understand and compare the different time complexity for various types of constraints, in this chapter, we present a series of tables to illustrate the time complexity for different types of non-local constraints beforehand of actually running the extraction algorithm. Then, we argue the essences of different types of constraints that lead to their differences of cost and end up with a new categorization.

6.1 The constraint template vs. time complexity table

In this section, we summarize the time complexity for different types of constraints organized by the 3 dimensions.

6.1.1 Single field constraints

<table>
<thead>
<tr>
<th>Constraint</th>
<th>Complexity</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field X must appear (exist)</td>
<td>$O(n \times L \times K \times 2)$</td>
<td>Additional variable to store the existence.</td>
</tr>
<tr>
<td>Field X must not appear (exist)</td>
<td>$O(n \times L \times (K - 1))$</td>
<td>Remove from label candidate: K -&gt; K-1</td>
</tr>
<tr>
<td>Field X must appear once</td>
<td>$O(n \times L \times K \times 2)$</td>
<td>Additional variable to store the count (0, 1).</td>
</tr>
<tr>
<td>Field X must appear twice</td>
<td>$O(n \times L \times K \times 3)$</td>
<td>Additional variable to store the count (0, 1, 2).</td>
</tr>
</tbody>
</table>

Table 6.1: Dimension 1: Quantitative

<table>
<thead>
<tr>
<th>Constraint</th>
<th>Complexity</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field X is (not) at the beginning/last</td>
<td>$O(n \times L \times K)$</td>
<td>No need to store the position since the constraint can be checked at the 1st token or last token.</td>
</tr>
</tbody>
</table>

Table 6.2: Dimension 2: Position
### 6.1.2 Multiple fields constraints

<table>
<thead>
<tr>
<th>Constraint</th>
<th>Complexity</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field X and Y must both exist</td>
<td>(O(n \times L \times K \times 2) = O(n \times L \times K \times 2))</td>
<td>(O(n \times L \times K \times 2^2))</td>
</tr>
<tr>
<td>Field X and Y must both not exist</td>
<td>(O(n \times L \times (K - 1)) = O(n \times L \times (K - 1)))</td>
<td>(O(n \times L \times (K - 1)) = O(n \times L \times (K - 1)))</td>
</tr>
<tr>
<td>Field X and Y must both appear once</td>
<td>(O(n \times L \times K \times 2) = O(n \times L \times K \times 2))</td>
<td>(O(n \times L \times K \times 2) = O(n \times L \times K \times 2))</td>
</tr>
<tr>
<td>Field X and Y must both appear twice</td>
<td>(O(n \times L \times K \times 3) = O(n \times L \times K \times 3))</td>
<td>(O(n \times L \times K \times 3) = O(n \times L \times K \times 3))</td>
</tr>
</tbody>
</table>

Table 6.4: Dimension 1: Quantitative

<table>
<thead>
<tr>
<th>Constraint</th>
<th>Complexity</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field X must (not) come before Y (adjacent)</td>
<td>(O(n \times L \times K))</td>
<td>We can use (f(i, k_p, k)) to check the constraint.</td>
</tr>
<tr>
<td>Field X must (not) come before Y (not adjacent)</td>
<td>(O(n \times L \times K \times 2))</td>
<td>Need to store the states whether X happens before in order to check if Y follows afterwards.</td>
</tr>
<tr>
<td>N Fields should (not) follow their order (adjacent)</td>
<td>(O(n \times L \times K))</td>
<td>Similar to the 2 fields case. This is a ((N-1)) combination of ((X,Y)) type constraints. Each of them can be enforced by using the (f(i, k_p, k)) and it will not incur additional cost.</td>
</tr>
<tr>
<td>Field should (not) follow their order (not adjacent)</td>
<td>(O(n \times L \times K \times N))</td>
<td>Need to store the states for all N fields.</td>
</tr>
</tbody>
</table>

Table 6.5: Dimension 2: Position
Field X and Field Y should (not) contain a set S of words

<table>
<thead>
<tr>
<th>Constraint</th>
<th>Complexity</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field X and Field Y should (not) contain a set S of words</td>
<td>$O(n \times L \times K)$</td>
<td>Check the two sets of words in the fields for existence correspondingly. No need to store additional information.</td>
</tr>
<tr>
<td>Field $X_1...X_N$ should (not) contain a set S of words</td>
<td>$O(n \times L \times K)$</td>
<td>Similar as above</td>
</tr>
</tbody>
</table>

Table 6.6: Dimension 3: Writing rules

6.2 Argument about the essence of the constraint cost

In our setting, we have the Markov property assumption. The current state (label) only depends on adjacent previous state. Based on previous observations of the time complexity, we concluded the following 3 types of constraints: 1). Token-based constraints; 2). Segment-based constraints; 3). Sentence-based constraints

1. For token-based constraints (local constraint), we conclude that they are cheap: Here the function is: $C_1(x_i, y_i, i)$. It still follows the Markov property on top of token. Thanks to the property, when we label token $x_i$, we only need to memorize the label of token $x_{i-1}$, which is cheap, rather than the labels of all other tokens, which is expensive.

2. For segment-based constraints, we conclude that they are also cheap if we relax the Markov property from token to segment by segment.

Here the function is: $C_2(x_{(i...i+l)}, y_{(i...i+l)}, i, l)$, $l$ is the length of segment. Thanks to the property, when we label field $k$, we only need to memorize information about field $k-1$. Here, field $(k-1)$ is specified by not only label but also the starting point (or the length) of that field. Thus, it is still affordable, even though it is more expensive than the case of token-based Markov property, because besides label, we need to memorize the length/starting point of field $(k-1)$.

3. For sentence-based constraints, we have to examine the whole sequence with labels to make the decision.

We define them as $C_3(X, Y)$, where we $X$ and $Y$ are the whole sequence and label assignment.

It does not follow the Markov property. However, for many of them, we can track the progress. For example, if we have to enforce title appears twice, we can track the number of title appearing in the citation so far until the number is equal to two. It is more expensive than the two above types of constraints because we have to memorize the progress besides the length and the label of the previous field. We may call the type of constraints as track-able sentence-based constraints.

For others, we define them as non-track-able sentence-based constraints. E.g.: Field X and field Y must contain a common word (don’t know the specific word, just know they must contain a common word) Field X and field Y must end same delimiter (don’t know the specific delimiter)
Chapter 7

Conclusions and Future work

We conduct a comparative study on constraint usage in Citation extraction problem. The constraint enforcement in CRF has been proved to be an effective approach to encode domain knowledge. The previous work has been focusing on finding the approximate solution to incorporate non-local structure while preserving tractable inference. We approach the non-local constraint enforcement from a different perspective: Instead of starting from the model, we start from the constraints themselves. With a comprehensive list of various types of non-local constraints generated in three dimensions, we adapt the segment-based Viterbi model to different constraints and make it possible to enforce in a dynamic programming fashion. We present the result in the way of showing the time complexity for incorporating non-local constraints by using the extended Viterbi model.

The current model and the analysis on the constraint enforcement serve as an important foundation for future work on the “Extraction” and “Match” model. With such model, it will become feasible to do enumerating and searching for the best candidate among all constraints we instantiated, which may be the key for “Match” model. There are a few open questions that need to be answered and potential work that should be done in the future:

1. How to prove that the list of constraints is complete in a more systematic way?

2. What do those complexities imply comparing to other constraint enforcement method?

3. Have the implementation of the model and conduct experiments to show more quantitative evidence.
References


