A Probabilistic Address Parser using Conditional Random Fields and Stochastic Regular Grammar

Minlue Wang, Valeriia Haberland, Amos Yeo,
Andrew Martin, John Howroyd and J. Mark Bishop
Tungsten Centre for Intelligent Data Analytics (TCIDA),
Goldsmiths, University of London, United Kingdom
Email: m.wang, v.haberland, ma201ya, a.martin, j.howroyd, m.bishop@gold.ac.uk

Abstract—**Automatic semantic annotation of data from databases or the web is an important pre-process for data cleansing and record linkage. It can be used to resolve the problem of imperfect field alignment in a database or identify comparable fields for matching records from multiple sources. The annotation process is not trivial because data values may be noisy, such as abbreviations, variations or misspellings. In particular, overlapping features usually exist in a lexicon-based approach. In this work, we present a probabilistic address parser based on linear-chain conditional random fields (CRFs), which allow more expressive token-level features compared to hidden Markov models (HMMs). In additions, we also proposed two *general* enhancement techniques to improve the performance. One is taking original semi-structure of the data into account. Another is post-processing of the output sequences of the parser by combining its conditional probability and a score function, which is based on a learned stochastic regular grammar (SRG) that captures segment-level dependencies. Experiments were conducted by comparing the CRF parser to a HMM parser and a semi-Markov CRF parser in two *real-world* datasets. The CRF parser out-performed the HMM parser and the semi-Markov CRF in both datasets in terms of classification accuracy. Leveraging the structure of the data and combining the linear-chain CRF with the SRG further improved the parser to achieve an accuracy of 97% on a postal dataset and 96% on a company dataset.**

I. Introduction

Record linkage is the task of finding records that refer to the same entity across one or more data sources. The challenges of record linkage comes from three main aspects. At the **schema-level**, heterogeneous data from multiple sources might exhibit different schemas for the same domain, such as differing addresses schemas. Once some comparable fields across multiple schemas have been identified, their field alignments have been found to be imperfect at the **field-level**. The misalignment can also exist for the data coming from the same source. At the **data-level**, noise in the data needs to be taken into account; e.g., data formats ("Unit 6-7" vs. "Unit 6/7"), typographical errors ("London" vs. "Londoon"), abbreviations ("Industrial Park" vs. "Ind PK"), missing values ("Queens Building" vs. "Queens"), and so on.

There has been much more work done regarding the issues at the schema-level and the data-level than at the field-level in record linkage[1], [2]. An illustrative example of the imperfect field alignment is shown in Table I. Record 1 and record 2 refer to the same restaurant in a database and need to be identified as the same entity to uphold data integrity. Most of the variations in data, i.e., misspelling ("Morto" vs. "Morton") and abbreviations ("Los Angeles" vs. "La"), occurs at the data-level. However, the second record's **State** field contains both postcode and state information, which would not be suitable when it is compared to other records directly. Jaro–Winkler similarity [3] between strings in each field of the record is also shown in Table I. Values of string comparisons in both **State** and **PostCode** fields are zero, so a non-matching decision is more likely to be made. An effective address parser could successfully identify "90048" in the second record as a postcode so that the parsed data could be re-arranged accordingly. As we can see from Table I, more meaningful values of string comparison can be achieved by resolving these imperfect field alignment issues.

In the real-world, addresses are often inconsistent, incomplete and errant in nature. Preprocessing of the data with addresses needs to take place before record linkage. In this work, we are interested in address parsing, which attempts to assign a *semantic* label to every token in an address so that a pair of the records can be better aligned against each other independent of whether the records come from the same database or from sources with different schemas.

Traditional rule-based address parsers have been shown to be limited in terms of classification accuracy and require too much domain knowledge in order to design the system [4]. Several probabilistic address parsers based on hidden Markov models (HMMs) [5], [4], [6] were developed to improve rule-based systems. However, *generative* models, such as Bayesian Networks or HMMs, have more difficulties when dealing with rich and complex features compared to *discriminative* models. Our proposed address parser is based on a conditional random field (CRF) model, a *discriminative* sequential classifier that has been applied to many other annotation tasks ranging from image segmentation [7] to entity extraction [8], [9].

Overlapping features are automatically extracted from raw addresses given a set of specialised lexicons, such as road types or county names. Boundaries between fields are also taken into account so that the data in different fields are less likely to be tagged as being in the same semantic category. However, informative token-wise features could be missing because of the incompleteness of our reference data or noisy input data, which affects the final classification performance.

DOI 10.1109/ICDMW.2016.44.

TABLE I
AN EXAMPLE OF DATA WITH IMPERFECT FIELD ALIGNMENT.

Record	**Restaurant**	**Street**	**City**	**State**	**PostCode**
1	Morton's	435 S La Cienega Blvd	Los Angeles	CA	90048
2	Morto's	435 S Los Angeles Cienega Blvd	La	90048,CA	
JaroWinkler	0.98	0.51	0.58	0	0
***Parsed* Records**					
1	Morton's	435 S La Cienega Blvd	Los Angeles	CA	90048
2	Morto's	435 S Los Angeles Cienega Blvd	La	CA	90048
JaroWinkler	0.98	0.51	0.58	1	1

In this work, a score derived from a learned stochastic regular grammar (SRG) is combined with the conditional probabilities of the CRF for reordering of the output sequences. The learned SRG is intended to capture the **segment-level** dependencies which cannot be easily supported by the token-wise feature model. Experiments on two *real-world* datasets demonstrated a better annotation result of our address parser than a HMM parser and a semi-Markov CRF parser [10] that one might expect to work better than the linear-chained CRF because of its ability to learn the segment-level dependencies. Differences between address parsing and standard name-entity recognitions (NERs) [11] are also discussed.

The rest of the paper is organized as follows. Section II discusses work that is related to address parsing. Section III presents background on conditional random fields for sequential classification. A lexicon-based address parser using linear-chained CRFs and stochastic regular grammar is shown in Section IV. In Section V, we present our empirical results by comparing our address parser to a HMM address parser and a semi-Markov CRF address parser on two real-world datasets. Finally, we conclude the paper and discuss future work in Section VI.

II. RELATED WORK

The first HMM-based address parser was proposed by Borkar et al. [5]. The hidden Markov model [12] consists of a set of observations and a set of states. Each state can transit to any other state associated with a transition probability and there is also an observation matrix that governs how likely an observation is generated by the state. Both transition matrix and observation matrix can be learned from a set of training examples using the *maximum likelihood* approach. Given a new sequence of observations, the best state sequence can be computed by the Viterbi algorithm [12]. The states in [5] are semantic labels for the address tokens; the observations in [5] are extracted based on the characteristics of tokens: all individual numbers are converted to a single special token; all delimiters are converted to another special symbol; all alphabetic words are left the same. One drawback of this approach is trying to learn the relationship directly between the semantic label and the individual raw token. If the training data does not have enough coverage on all possible tokens, a smoothing technique usually has to be employed in order to take care of any unseen token. For instance, a small observation probability can be assigned to the unseen token from the states. No special lexicons were used to extract observations further from the raw tokens. Li et al. [6] trained a similar HMM address parser based on nearly 100 million unique addresses from a high quality data source. In particular, they added some variations to generate better synthetic data. However, such large high-quality training data might not be available at the first place for a specific country and there is also no flexibility to design an alternative schema rather than the one used at the 'golden' source.

Churches et al. [4] designed an alternative HMM-based parser that makes more use of reference tables for extracting observations. Each address input string is firstly tokenised into a set of words and then each word is assigned with an observation from a set of look-up tables. The reference tables contain information about postal codes, city names or county names from postal authorities or governments. The observation assignments follow a greedy matching algorithm, which prefers assigning labels over a sequence of words than the individual word. Automatically generated observations are not good enough for parsing an arbitrary address because the greedy assignment algorithm is deterministic so that each word is always given a particular label. For example, "London" is always observed as "City" even when appearing in other contexts, such as "London Road" where it is more likely to be a street name. A HMM is able to recover from this incorrect observation by considering the underlying state sequence.

However, HMMs do not allow more complex observations, such as multiple observations for one token at the same time. A simple extension of previous HMM address parsers to handle multiple observations was done in [13], where all possible observation sequences are considered as input of the *Viterbi* algorithm and the best states sequence is the one with highest probability. However, the number of observation sequences is therefore exponential to the number of tokens in an address.

An early attempt of allowing complex observations without concern of their dependant relationships was using maximum entropy Markov models (MEMMs) [14]. The transition function and the observation function of traditional HMMs are replaced by a single transition function so that current state depends not only on previous state but also on current observations. However, there is a major weakness of MEMMs, which is called the *label bias problem*. Because each transition function is normalised per-state rather than over the entire sequence, significant bias can be passed from one state to the next. Conditional random fields can be considered as unnormalised version of MEMMs [15], so that the label bias problem can be avoided by considering normalisation over all sequences.

Semi-Markov CRFs have been proposed to capture the label dependency [10] at segment-level rather than at token-level as in linear-chain CRFs , which have been shown improvements in many name-entity recognitions (NERs)[11], [16]. However, there are two main differences between standard NERs and address parsing. First of all, entity values in many NERs are distinct, such as DNA names and RNA names in bioinformatics. In the addresses, entity values can be overlapping, for instance, "market street" can be either a road name or a sub-locality. Secondly, the entities in NERs usually have distinct segment-level features, such as "entity length" or "similarity to other known entities". The segment-level similarities are usually computed based on the dictionaries that store some **full-length** entities. In the address parsing, although some of our lexicons do store full-length entities, such as cities or counties, we only store some part of other entities, such as road identifiers ("road", "street") for road entity. There is no sufficient segment-level features for all entities. Overall, the address parsing can be considered as a multi-entity extraction problem with overlapping entity values and the external dictionaries do not have full-length entity values. It is anticipated that semi-Markov CRFs do not produce comparable results for the address parsing.

III. Conditional Random Fields

In this section, we are going to provide some background on conditional random fields: a *discriminative* approach for solving problems of *sequential classification*.

Following the work in [17], let $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T)$ be an input feature vector sequence , and $\mathbf{Y} = (y_1, y_2, \dots, y_T)$ be a random output variable over the sequential data $\mathbf{X}$ of length T. Then $\mathbf{x}_t = \{x_{t1}, x_{t2}, \dots, x_{tK}\}$ is a feature vector with (fixed) size K and y_t is assumed to range over a finite label alphabet S. The input sequence $\mathbf{X}$ are assumed to be observed, and called the *features*, and the outputs Y are named (underlying) *labels*. The goal of sequential classification is to find the best assignment to $\mathbf{Y}$ given the features $\mathbf{X}$. For example, in natural language processing, the task of *part-of-speech tagging* is to assign a particular part-of-speech tag (such as noun, verb or preposition) to each word in a sentence. Thus y_t, in this case, is the part-of-speech tag assigned to t^{th} word in the sentence, and $\mathbf{x}_t$ is a feature vector that captures useful information about the corresponding words. For instance, $\mathbf{x}_t$ could include a binary variable $\mathbf{x}_{tk}$ to indicate whether the word is capitalised.

Generative approaches, such as Bayesian networks and hidden Markov models, focus on maximising the joint distribution $p(Y, X)$; which is usually computed by applying Bayes' rule:

$$p(Y, X) = p(X|Y)p(Y). \tag{1}$$

$P(Y)$ is a prior distribution and $P(X|Y)$ is a likelihood function that governs how the feature X is generated given a state Y. In Generative models, an essential question that needs to be answered is how to decompose $P(X|Y)$ into factor representations. Typically, this requires assumptions of independence of the features (such as naive Bayes) or structual learning to find the best decomposition.

CRF models, on the other hand, do not use this decompostion but directly compute the conditional probability $p(Y|X)$. As HMMs can be considered as an extension of Naive Bayes for sequential data, so CRF models can be considered as an extension of logistic regression. More detailed comparison of generative and discriminative approaches can be found in [17].

When the data sequence is restricted to just one item ($T = 1$), logistic regression can be used to compute the conditional probability $P(y_1|\mathbf{x}_1)$ as follows[1]:

$$p(y_1 = i|\mathbf{x}_1) = \frac{1}{Z(X)} exp(\sum_{k=1}^{K} \theta_{ik} x_{1k}) \tag{2}$$

where $Z(X) = \sum_{i=1}^{|S|} exp(\sum_{k=1}^{K} \theta_{ik} x_{1k})$ and is a normalization constant (we ignore a state bias θ_i).

The simplest form of CRFs for sequential data are *linear-chain* CRFs, which only assume conditional probability between adjacent items. Thus, Equation 2 is rewritten as follows:

$$p(y_t = i|y_{t-1} = j, \mathbf{x_t}) \quad = \quad \frac{1}{Z(X)} exp(\sum_{k=1}^{K} \theta_{ijk} x_{tk}) \tag{3}$$

and taking the product of such terms, over the sequence, to form $P(\mathbf{Y}|\mathbf{X})$.

In linear-chain CRFs, the dimensionality of parameter θ is $M = |S|^2 \times K$. However, not all θ_{ijk} need to be learned from the training data $D = \{\mathbf{X}^{(n)}, \mathbf{Y}^{(n)}\}_{n=1}^{N}$, because some combinations of y_{t-1},y_t and x_t never occur. Parameter estimation for linear-chain CRFs is usually done by maximising log likelihood function $l(\theta) = \log P(\mathbf{Y}|\mathbf{X})$; typically using numerical optimisation such as gradient ascent or Newton's method.

IV. Address Parser

We are presenting an address parser using a linear-chain CRF model described above to semantically annotate addresses that come from databases or the Web. The task of semantic annotation of an address is different from other natural language applications because the texts are less regular in an address. In real world applications, addresses often have imperfect field alignment and noisy data values. For example, information about "country" could be missing for an address in a Web or a "city" column from a database might store data other than values of "city". Our goal is to parse these addresses into corresponding fields so that a better data quality can be maintained.

[1]We assume variable $\mathbf{x}_{1k}$ is a binary variable.

TABLE II
A SAMPLE OF REFERENCE DATA FOR GB ADDRESSES.

Lookup Tables	Sample Words
PostDistrict	London, West Bromwich
County	London, West Midlands
Geolocation Qualifier	West, South

A. Extraction of Features

One of the advantages of using CRFs is its expressiveness of features, allowing rich and overlapping features to be extracted from the raw data. For instance, as shown in Table II, word "London" can be found in both GB's post district and county tables, features for token "London" are "*GB_pdis_nm*" and "*GB_cnty_nm*", saying "London" can be found in both post district and county reference tables.

We also explicitly distinguish data in the reference tables with more than one word so that *X_part* feature could be generated to represent that the current token can be found in a token sequence of reference table "X". As you can see from Table II, token "West" is not only part of a post district and a county but also a complete geolocation qualifier. Automatically generated features for the token "West" would be "*GB_pdis_nm_part*", "*GB_cnty_nm_part*" and "*Geo _qualifier*".

The addresses we intend to parse could be incomplete (missing data), inaccurate (typographic error), or even have redundant information (repetitions). Take the following address for example: "Flat 2 Monet Court Monet Court 2 Stubbs Drive London SE16 3EG UK". Premises name "Monet Court" appears twice in a row and the second appearance is considered as "junk". With our CRFs address parses, it is desirable to annotate "junk" label for such redundant information. Thus, we add a feature function that can look at previous tokens in order to determine if the current token is a repetition. Therefore, tokens "Monet" and "Court" in the second appearance will be assigned a feature "*repetition*".

B. Semi-Structured Data

The addresses we considered in this work are not always unstructured, but could have semi-structure. Köpcke and Rahm [2] reviewed numerous studies of record linkage which mainly focused on structured and often relational data, while semi-structured and unstructured data received much less attention. Note that the difference between fully structured and semi-structured data is not strictly determined and can differ across domains and data representation. We focus on relational structured and semi-structured data as defined below, following the definitions in which are defined in [18].

Structured data: Fully structured data is considered to be relational data where each field has a designated meaning. For example, if a field is designated for the house number of the address, then the corresponding field in each record should only contain this part of the address.

Semi-structured data: Semi-structured data is data that has some degree of flexibility, such as imperfect field alignment where the data might appear in any of a number of fields which is not necessarily designated to these data values or where these field values could be furthered parsed into a set of elementary attributes. For example, the whole address may be stored textually in a single field or may be assigned to multiple fields without any particular designation of purpose; so that, the postal town coupled with post code may appear as a single filed.

For instance, a semi-structured company address from the web looks as follows:

```
<br> "1600 Amphiteatre ParkWay"
<br> "Mountain View, CA 94043"
<br> "USA"
```

There is already a **br** tag between token "Parkway" and "Mountain", which indicates it is unlikely for "Parkway" and "Mountain" to be in the same semantic category after parsing. Similar semi-structure could be found in databases where data is separated into columns. In order to leverage this data structure, we generate a "*field_separator*" feature whenever there is a boundary between field values.

C. Stochastic Regular Grammar

Our lexicon-based address parser relies on generating features mainly from a set of reference tables. An important feature could be missing for a token in an address for three reasons:

(i) Missing reference data; for example where the locality table (for small sub-districts, e.g. villages) is incomplete;
(ii) Peculiar entity references; such as "long acre" (a road in London) that omits a street identifier;
(iii) Omitted identifiers in the data; such as "X Y" instead of "X Y industrial park". With key words "industrial park" missing, it is more likely for the CRF parser to assign **unknown** state to the arbitrary tokens "X" and "Y", because tokens labelled as **unknown** in the training data also often lack any useful features apart from whether they are alphabetic or alphanumeric.

In order to reduce the effect of missing features, we proposed to directly learn a stochastic regular grammar (SRG) from the *lifted* label sequences, where a sequence of the same labels is lifted as a macro label. For example, an address sequence {"Road Road Road", "City City Postcode Postcode"} will become {Road City Postcode} in the training examples for learning the grammar. The learned grammar serves the same purpose as the transition matrix in HMM, but it is intended to capture the dependencies between labels at the segment-level rather than at the token-level and is more sensitive to the examples presented in the data. For instance, given two state sequences $\{BCBD, BC\}$, transition probability from B to D learned in HMM is $1/3$, so a new sequence $\{BD\}$ will have $1/3$ probability being accepted, while the stochastic regular grammar described later will reject

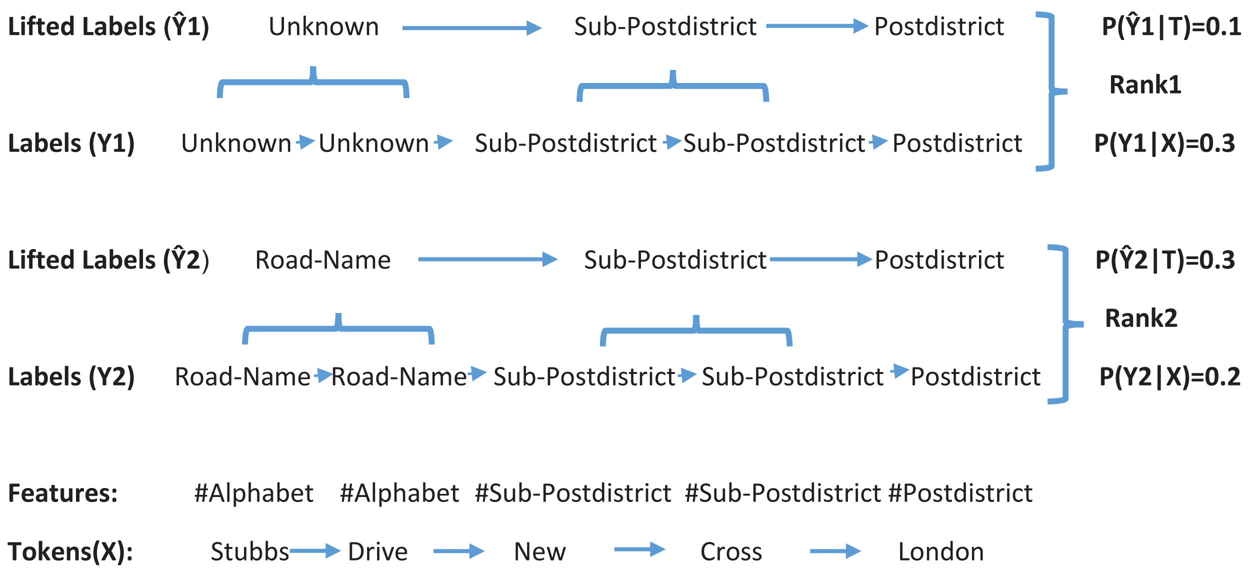

Fig. 1. Two best output sequences $Y1$ and $Y2$ from the CRF parser and their conditional probabilities $P(Y|X)$. The Corresponding lifted label sequences are $\hat{Y}1$ and $\hat{Y}2$ with probability $P(\hat{Y}|T)$.

the sequence $\{BD\}$ because there is no transition from an initial B to a D in the examples.

Let A be a finite alphabet which contains all semantic labels for addresses. An address language L contains all possible lifted label sequences of the address. The learned regular grammar is a minimal stochastic finite automata (SFA), which not only can decide whether a sequence of labels can be accepted by L, but also can assign a probability distribution to the sequences in the language L. The learning can be done by firstly creating a prefix tree acceptor from a set of examples and then merging similar states using an algorithm called ALEGRIA [19]. The algorithm can identify the canonical acceptor of a language with polynomial complexity in the size of the training data.

The minimal SFA is used to re-rank the outputs of the linear-chained CRFs. Given the K-top candidate sequences $Y^i, i \in 1, \ldots, K$, generated from the CRF address parser, each candidate sequences Y^i has its own conditional probability $P(Y^i|X)$. The best sequence that combines CRF and SFA is selected as the one that maximises the joint probability:

$$Y^* = \arg\max_{Y^i} P(Y^i|X) * score(Y^i|T) \tag{4}$$

where T is the minimal SFA that is learned from the examples and the score is computed as follows:

$$score(Y^i|T) = \frac{P(\hat{Y^i}|T)}{|\hat{Y^i}|(1 + |Y^i| - |\hat{Y^i}|)} \tag{5}$$

$\hat{Y^i}$ is the lifted label sequence, $P(\hat{Y^i}|T)$ is the acceptance probability given the learned grammar T. The length $|\hat{Y^i}|$ is used so that average transition probability is computed. Variable $|Y^i| - |\hat{Y^i}|$ is added to penalise the difference between actual sequence length $|Y^i|$ and lifted sequence length $|\hat{Y^i}|$. In an extreme situation, if every token in an address is assigned the same label in an output Y, there is going to be only one lifted label left after lifting ($|\hat{Y}| = 1$), which would have a rather large average score $\frac{P(\hat{Y}|T)}{|\hat{Y}|}$. In summary, the function $score(Y^i|T)$ gives a large score to a sequence with larger average transition probabilities and more distinct labels.

Figure 1 shows an example that combines the CRF conditional probability and score function to reorder the output sequences. The best two sequences for the address *"Stubbs Drive New Cross London"* are $Y1$ and $Y2$. Because both token "Stubbs" and "Drive" do not possess any informative features apart from "Alphabet", the best sequence $Y1$ of the CRF parser is *"Unknown, Unknown, Sub-Postdistrict, Sub-Postdistrict and Postdistrict"* with conditional probability $P(Y1|X) = 0.3$. The lifted sequence $\hat{Y}1$, in this case, is *"Unknown, Sub-Postdistrict and Postdistrict"* with length $|\hat{Y}1| = 3$ and $|Y1| - |\hat{Y}1| = 2$. The $score(Y1|T)$ computed by Equation (5) is $0.1/(3 * (1 + 2)) = 0.01$ and the score of $Y2$ is $0.3/(3 * (1 + 2)) = 0.03$. Thus, the best sequence is therefore $Y2$ using Equation (4).

As we mentioned before, the learned regular grammar is sensitive to the examples presented in the data. If only label sequences in the training data are used to learn the grammar, a large number of test sequences will be rejected. Thus, the examples we used to learn the grammar come from two sources. One is training data where each label sequence has a probability 1 because we are sure the appearance of the sequence in the address language L. The other source is k-top state sequences from the CRF parser associated with their conditional probability $P(Y|X)$. For example, $Y1$ and $Y2$ in Figure 1 are both presented as the examples for learning the grammar with probability 0.3 and 0.2 respectively.

V. Experiments

In this section, we compared our CRF address parser to a HMM address parser [4] and a semi-Markov CRF address parser [10] which allows each label to persist for a non-unit length of time. In the linear-chain CRF, multiple features can be extracted for a token, while there is only one feature for each token in HMM. As for the semi-Markov CRFs, the features are extracted on the segment-level, that is, all previously mentioned token-level features are combined with the indicators for the begin, the middle and the end of a segment. The inference of semi-Markov CRFs is more expensive than conventional linear-chained CRFs because it has to search over all possible segmentations. Two *real world* datasets were tested in the experiment. One is a *postal* dataset [2], which contains GB addresses randomly generated from Google. The other is a *company* dataset which contains GB company addresses.

Postal Dataset

The addresses in this dataset have much less noise and imperfect field alignment. For example, most of the abbreviations, such as “st” or “rd”, have already been standardised and the token “UK” always appears in the last field of an address. Another thing worth noting here is that the addresses in this dataset do not have information below house level, such as floor number, or a flat indicator. When field separators are not considered, semi-structured data are concatenated across all fields to form a single string. As you can see from Table III, the CRF and semi-Markov CRF address parsers have better token-wise accuracies than the HMM parser when the field separator is not taken into account, because they can deal better with the overlapping features. The performance of all three models improve with the introduction of the field separator, which demonstrated a general advantage of making use of the semi-structure of the data. Linear-chained models, such as HMM and CRF, also benefit from the additional SFA that learns segment-level dependency. As we discusses before, the semi-Markov CRF does not perform well because there is no sufficient segment-level features to support the global dependency in the model and the overlapping in entity values and features make the inference of semi-Markov CRF more difficult.

The performance of the parsers on individual labels is shown in Table V. In this experiment, we use the F_1 score as our evaluation criterion, which takes both precision and recall into account and is computed as follows:

$$F_1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{6}$$

where precision $= \frac{TP}{TP+FP}$ and recall $= \frac{TP}{TP+FN}$. TP, FP, and FN stands for the number of true positives, false positives and false negatives receptively. The CRF has a higher (or equal) F_1 score for most of the labels than the HMM and the semi-Markov CRF.

TABLE III
Performance of CRFs and HMM address parser on the postal dataset.

Methods	Token-Wise Accuracy
HMM	91.8%
Semi-Markov CRF	93.0%
CRF	95.6%
HMM+FieldSeparator	93.1%
Semi-Markov CRF+FieldSeparator	93.7%
CRF+FieldSeparator	97.2%
HMM+FieldSeparator+SFA	94.4%
CRF+FieldSeparator+SFA	**97.7%**

TABLE IV
Comparison results of CRFs and HMM address parser on the company dataset.

Methods	Token-Wise Accuracy
HMM	85.7%
Semi-Markov CRF	87.8%
CRF	90.9%
HMM+FieldSeparator	93.8%
Semi-Markov CRF+FieldSeparator	88.7%
CRF+FieldSeparator	95.9%
HMM+FieldSeparatar+SFA	93.7%
CRF+FieldSeparator+SFA	**96.2%**

Company Dataset

There are 433 records of GB companies in this dataset, which were randomly selected from a private company’s database [3]. Since both company names and addresses appear in the database, we also included a state label “CO” to represent company names. The database has one column to store the company name and six columns to store the address. Compared to the postal dataset, the imperfect field alignment problem is much more severe and data values are much more noisy, such as abbreviations, misspellings and missing values. We assigned label “JK” to tokens that are clearly redundant and use label “UN” for tokens that cannot be decided by our domain experts. Another thing worth pointing out here is company addresses in this dataset have much richer information below road level, such as “X industrial park” or “X department in a building”. The goal of our address parser is to assign a semantic label to each token of an address so that data belonging to the same semantic field can be placed in the same column. In addition, data labelled as “junk” can be cleaned up so that a better data quality can be maintained. Both the training data and test data are manually labelled by domain experts. There are 200 addresses in the training data and 233 addresses in the test data.

Table IV compares the performance between different address parsers for the company dataset. When the boundary of

[2]Addresses are generated from the website: https://www.doogal.co.uk/RandomAddresses.php. Labelled data is publicly available at https://www.dropbox.com/sh/zaoqpuc0gawxpvx/AABHbikbe49ON2xPdEf7kwo7a?dl=0

[3]Because of the company’s policy, the dataset is not publicly available.

TABLE V
PRECISION, RECALL AND F1-SCORE OF INDIVIDUAL LABELS FOR THE POSTAL DATASET.

Label	**CRFs + SFA**			**HMMs**			**Semi-Markov CRFs**		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score	Precision	Recall	F1 Score
House Number(HN)	1	1	**1**	1	1	**1**	1	1	**1**
Road (RD)	1	0.92	**0.96**	0.94	0.92	0.93	0.88	0.97	0.92
Sub District (SD)	1	1	**1**	1	1	**1**	0.80	0.80	0.80
Post District (PD)	0.94	0.94	0.94	1	0.94	**0.97**	0.94	0.94	0.94
County (CN)	0.96	0.92	**0.94**	1	0.64	0.78	0.91	0.8	0.85
PostCode (PC)	1	1	**1**	1	1	**1**	0.97	1	0.99
Country (CY)	1	1	**1**	1	1	**1**	1	1	**1**
Junk (JK)	0.5	1	**0.67**	0.09	1	0.17	0	0	
Average	0.98	0.96	**0.97**	0.98	0.95	0.95	0.93	0.94	0.93

TABLE VI
PRECISION, RECALL AND F1-SCORE OF INDIVIDUAL LABELS FOR THE COMPANY DATASET.

Label	**CRFs + SFA**			**HMMs**			**Semi-Markov CRFs**		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score	Precision	Recall	F1 Score
Company (CO)	0.97	0.99	0.98	0.99	0.99	**0.99**	0.73	0.83	0.77
Post box (PB)	0.67	1	0.80	1	1	**1**	1	1	**1**
Postbox number (PN)	0.5	1	0.67	1	1	**1**	1	1	**1**
Sub Building (SB)	1	0.35	0.52	1	0.53	**0.70**	1	0.47	0.64
SubBuilding (SN) Number	0.86	0.86	**0.86**	0.56	0.71	0.61	0.86	0.86	**0.86**
Building (BU)	0.81	0.96	**0.88**	0.86	0.81	0.83	0.73	0.83	0.78
House Number (HN)	0.97	0.96	**0.97**	0.91	0.96	0.93	0.94	0.96	0.95
Sub Road(SR)	0.94	0.90	**0.92**	0.91	0.76	0.83	0.88	0.84	0.86
Road (RD)	0.98	0.95	**0.97**	0.75	0.89	0.81	0.93	0.89	0.91
Sub District(SD)	0.82	0.85	**0.84**	0.70	0.70	0.70	0.67	0.51	0.58
Post District (PD)	0.92	0.98	**0.95**	0.85	0.99	0.91	0.71	0.91	0.80
County (CN)	0.98	1	**0.99**	0.97	1	0.98	0.84	0.74	0.79
PostCode (PC)	1	0.98	**0.99**	0.99	0.97	0.98	0.99	0.93	0.96
Country (CY)	1	1	**1**	0.92	1	0.96	0.83	0.76	0.79
Special (SP)	1	0.5	0.67	0.8	1	**0.89**	0.2	0.13	0.15
Junk (JK)	0.75	0.2	**0.28**	0.29	0.06	0.10	0	0	
Unknown (UN)	0.2	0.1	0.15	0	0		0.16	0.21	**0.19**
Average	**0.94**	**0.94**	**0.94**	0.89	0.90	0.90	0.81	0.83	0.82

field values is not considered, the CRF achieved 90.9% accuracy, which is better than two baselines. Additional boundary features improve the performance of both CRF and HMM to 95.9% and 93.8% respectively, while the accuracy of semi-Markov CRFs is only 88.7%. Finally, the linear-chain CRF, coupled with the field separator and the learned grammar, achieved the best performance with 96.2% accuracy.

The performance for individual labels with this dataset is shown in Table VI. The CRF has higher F_1 scores in the majority of cases, in particular with values that are overlapping, e.g. sub road, road, post district and so on. As you can see from Table VI, the CRF, as compared to HMM, was also able to generate much more correct "Junk" labels for the redundant data because of the proposed repetition features. HMM predicts 100% correctly on the label PB because the corresponding feature has no overlapping with any other.

VI. CONCLUSION AND FUTURE WORK

We presented a lexicon-based probabilistic address parser based on conditional random fields that can automatically annotate semi-structured addresses. Boundaries between fields are taken into account to leverage the semi-structure of the data. In addition, a stochastic grammar learned from the lifted labels is used to capture the segment-level dependencies. Experiments on two real-world datasets demonstrated a better annotation ability of the linear-chained CRF coupled with the learned SFA compared to the baselines. One thing worth noting here is that leveraging the semi-structure of the data and the learned SFA are two *general* enhancement techniques that can be applied to improve any linear-chained models.

Human labelling is time consuming when generating both training and test data, we would like to extend our CRF models in a semi-supervised setting. Generative models, such as Bayesian networks or HMMs, have natural ways of handling unlabelled data in the training data. For example, standard Baum-Welch algorithm can be applied with HMMs for learning parameters from partially labelled data. However, it is not straight-forward to apply CRFs for semi-supervised learning as discussed in [20], [21].

Another direction of our future work is making use of external sources for data cleaning and validation [22]. There might be more than one postcode appearing in an address, e.g. a company's current postcode and previous postcode are both appearing in the record. Without referring to external sources, it is extremely difficult for the current address parser to decide which postcode is more likely to be correct.

REFERENCES

[1] E. Rahm and P. A. Bernstein, "A survey of approaches to automatic schema matching," *The VLDB Journal*, vol. 10, no. 4, pp. 334–350, 2001.

[2] H. Köpcke and E. Rahm, "Frameworks for Entity Matching: A Comparison," *Data & Knowledge Engineering*, vol. 69, no. 2, pp. 197–210, 2010.

[3] W. E. Winkler, "String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage," 1990.

[4] T. Churches, P. Christen, K. Lim, and J. X. Zhu, "Preparation of Name and Address Data for Record Linkage using Hidden Markov Models," *BMC Medical Informatics and Decision Making*, vol. 2, no. 1, p. 1, 2002.

[5] V. Borkar, K. Deshmukh, and S. Sarawagi, "Automatic segmentation of text into structured records," in *ACM SIGMOD Record*, vol. 30, no. 2. ACM, 2001, pp. 175–186.

[6] X. Li, H. Kardes, X. Wang, and A. Sun, "HMM-based Address Parsing with Massive Synthetic Training Data Generation," in *Proceedings of the 4th International Workshop on Location and the Web*. ACM, 2014, pp. 33–36.

[7] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr, "Conditional Random Fields as Recurrent Neural Networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1529–1537.

[8] N. V. Cuong, N. Ye, W. S. Lee, and H. L. Chieu, "Conditional Random Field with High-Order Dependencies for Sequence Labeling and Legmentation," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 981–1009, 2014.

[9] F. Peng and A. McCallum, "Information Extraction from Research Papers using Conditional Random Fields," *Information processing & management*, vol. 42, no. 4, pp. 963–979, 2006.

[10] S. Sarawagi and W. W. Cohen, "Semi-Markov Conditional Random Fields for Information Extraction," in *Advances in neural information processing systems*, 2004, pp. 1185–1192.

[11] D. Okanohara, Y. Miyao, Y. Tsuruoka, and J. Tsujii, "Improving the Scalability of Semi-Markov Conditional Random Fields for Named Entity Recognition," in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2006, pp. 465–472.

[12] Z. Ghahramani, "An Introduction to Hidden Markov Models and Bayesian Networks," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 15, no. 01, pp. 9–42, 2001.

[13] P. Christen and D. Belacic, "Automated Probabilistic Address Standardisation and Verification," in *Australasian Data Mining Conference (AusDM05)*, 2005, pp. 53–67.

[14] A. McCallum, D. Freitag, and F. C. Pereira, "Maximum Entropy Markov Models for Information Extraction and Segmentation," in *Proceedings of 17th International Conference on Machine Learning*, 2000.

[15] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," in *Proceedings of the 8th International Conference on Machine Learning*, 2001.

[16] K. Yoshida and J. Tsujii, "Reranking for biomedical named-entity recognition," in *Proceedings of the Workshop on BioNLP 2007: Biological, Translational, and Clinical Language Processing*. Association for Computational Linguistics, 2007, pp. 209–216.

[17] C. Sutton and A. McCallum, "An Introduction to Conditional Random Fields," *Machine Learning*, vol. 4, no. 4, pp. 267–373, 2011.

[18] C. Batini, C. Cappiello, C. Francalanci, and A. Maurino, "Methodologies for Data Quality Assessment and Improvement," *ACM computing surveys (CSUR)*, vol. 41, no. 3, p. 16, 2009.

[19] R. C. Carrasco and J. Oncina, "Learning Stochastic Regular Grammars by Means of a State Merging Method," in *Grammatical Inference and Applications*. Springer, 1994, pp. 139–152.

[20] F. Jiao, S. Wang, C.-H. Lee, R. Greiner, and D. Schuurmans, "Semi-supervised Conditional Random Fields for Improved Sequence Segmentation and Labeling," in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2006, pp. 209–216.

[21] G. S. Mann and A. McCallum, "Generalized Expectation Criteria for Semi-Supervised Learning of Conditional Random Fields," in *The 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Columbus, OH*, 2008.

[22] K. Gomadam, P. Yeh, and K. Verma, "Data Enrichment Using Data Sources on the Web," in *AAAI Spring Symposium Series*, 2012.