Neural Chinese Address Parsing

Hao Li and **Wei Lu**
StatNLP Research Group
Singapore University of Technology and Design
hao_li@mymail.sutd.edu.sg
luwei@sutd.edu.sg

Pengjun Xie and **Linlin Li**
DAMO Academy
Alibaba Group
chengchen.xpj@alibaba-inc.com
linyan.lll@alibaba-inc.com

Abstract

This paper introduces a new task – Chinese address parsing – the task of mapping Chinese addresses into semantically meaningful chunks. While it is possible to model this problem using a conventional sequence labelling approach, our observation is that there exist complex dependencies between labels that cannot be readily captured by a simple linear-chain structure. We investigate neural structured prediction models with latent variables to capture such rich structural information within Chinese addresses. We create and publicly release a new dataset consisting of 15,000 Chinese addresses, and conduct extensive experiments on the dataset to investigate the model effectiveness and robustness. We release our code and data at http://statnlp.org/research/sp.

Figure 1: Two example Chinese addresses and the expected structures after parsing. Each chunk is underlined with its corresponding label in blue.

1 Introduction

Addresses play an important role in modern society. They are typically used as identifiers to locations and entities in the world that can be used to facilitate various social activities, such as business correspondences, meetings and events. Recent research efforts show that systems that perform *address parsing*, the task of automatically parsing addresses into semantically meaningful structures, can be useful for tasks such as building e-commerce or product recommendation systems (Jia et al., 2017; Avvenuti et al., 2018). Due to historical reasons, the English addresses come with a standardized format, mostly written in order from most specific to most general. Meaningful chunks in an English address are also separated by punctuation or the new-line symbols. Such characteristics make parsing English addresses a relatively easy task.

However, addresses written in eastern Asian languages such as Chinese present several unique challenges. Unlike English addresses, Chinese addresses are typically written in the form of a consecutive sequence of Chinese characters (possibly intermixed with digits and English letters). Figure 1 presents two example Chinese addresses and their desired output structures after parsing – chunks annotated with their labels indicating semantics (such as province, road, etc). The Chinese addressing system is also different from that of English. Though it is generally believed that the system uses the opposite ordering – starting from most general (e.g., province) and ending with most specific (e.g., room no.), in practice it can be observed that the format is far less rigorous than expected. The lack of rigor also leads to other issues – the addresses may come with incomplete, redundant or even inaccurate information, as we can see from the second example listed in Figure 1. Such unique challenges make the design of an effective Chinese address parser non-trivial.

Proceedings of NAACL-HLT 2019, pages 3421–3431
Minneapolis, Minnesota, June 2 - June 7, 2019.

Label	Order ID	Unique#	Train	Dev	Test	Interpretation	Example
COUNTRY	20	2	41	15	13	*name of a country*	中国(China)
PROVINCE	19	65	3,794	1,317	1,265	*name of a province*	浙江省(Zhejiang Province)
CITY	18	377	4,824	1,662	1,613	*name of a city*	北京市(Beijing)
DISTRICT	17	1137	5,881	2,027	1,921	*name of a district in a city*	朝阳区(Chaoyang District)
DEVZONE	16	297	330	118	107	*name of an economic development zone*	下沙开发区(Xiasha Development Zone)
TOWN	15	2,382	3,972	1,300	1,308	*name of a town or a boulevard*	乔司镇(Qiaosi Street)
COMMUNITY	14	1,867	1,279	415	416	*name of a community or a village*	荆山社区(Jingshan Community)
ROAD	13	5,037	5,410	1,788	1,801	*name of a road*	中山路(Zhongshan Road)
SUBROAD	12	486	333	109	130	*name of a lane*	丹心巷(Danxin Road)
ROADNO	11	2,676	4,316	1,435	1,401	*road number*	4-5号(#4-#5)
SUBROADNO	10	215	170	68	75	*road number for a subroad*	8号(#8)
POI	9	8,662	6,312	2,093	2,122	*name of the point of interest*	萧山医院(Xiaoshan Hospital)
SUBPOI	8	1,642	1,435	461	487	*name of the second point of interest*	西三苑(Xisan Sub-residence)
HOUSENO	7	1,309	2,993	978	943	*house number*	3幢(Block #3)
CELLNO	6	295	1,134	388	358	*cell number*	1单元(Unit #1)
FLOORNO	5	258	1,119	346	331	*floor number*	5层(Level 5)
ROOMNO	4	3,245	2,702	883	824	*room number*	402室(Room 402)
PERSON	3	903	650	207	208	*name of the third point of interest or a person*	大厅(the hall)
ASSIST	2	264	718	207	241	*a phrase for indicating relative position*	对面(opposite)
REDUNDANT	1	1,009	3,517	1,208	1,137	*redundant characters as well as repeated characters*	-,!
OTHERINFO	0	5	4	0	3	*a chunk which cannot be assigned any label above*	
Total Instances			8,957	2,985	2,985		
Total Chunks		32,133	50,934	17,024	16,704		

Table 1: Statistics of different labels in our *Chinese Address* corpus.

Parsing a Chinese address into semantically meaningful structures can be regarded as a special type of chunking task (Abney, 1991), where we need to perform address-specific Chinese word segmentation (Xue, 2003; Peng et al., 2004; Zhao et al., 2006) while assigning a semantic label to each chunk. However, existing models designed for chunking may not be readily applicable in this task. Our observations show that there are a few characteristics associated with the task. We found that while generally there exists certain ordering information among the chunks of different labels in the addresses, such ordering information is better preserved among the chunks that appear at the beginning of the addresses. For the chunks appearing towards the end of the addresses, chunks of different types often appear in more flexible order.

On top of the above observations, we propose a specific model based on neural networks for the task of Chinese address parsing. The model is able to encode the regular patterns among chunks that appear at the beginning of a Chinese address, while flexibly capturing the irregular patterns and rich dependencies among the chunks of different types that appear towards the end of the address. This is achieved by designing a novel structured representation integrating both a linear structure and a latent-variable tree structure.

Our main contributions in this work can be summarized as follows:

- We create and publicly release a new corpus consisting of 15K Chinese address entries fully annotated with chunk boundaries and address labels. To the best of our knowledge, this is the first and largest annotated Chinese address corpus.
- We introduce a novel neural approach to Chinese address parsing with latent variables to flexibly capture both prior ordering information and rich dependencies among labels.
- Through extensive experiments, we demonstrate the effectiveness of our approach. The experimental results show that our approach outperforms several baselines significantly.

2 Data

In this work, we created a *Chinese Address* corpus. To do so, we crawled a large number of publicly available addresses from the Chinese websites including online business directory websites (e.g., `b2b.huangye88.com`), social media websites (e.g., `www.dianping.com`), and an online API service translating a geo-location to a Chinese address (`lbs.amap.com`). In order to protect privacy, we discarded sensitive addresses (such as those involving military locations) and randomly altered the digits in the collected addresses.

Due to the lack of Chinese address standard format as well as complicated and different writing preferences in different regions (e.g., people living in southern China prefer the word "弄" as the suffix of the name of a lane or sub-road over the word "胡同" which is widely used in northern China), we create an annotation guideline[1] by summarizing different writing preferences. We proposed 21 chunk labels listed in Table 1. The meaning

[1]The annotation guideline can be found at `http://statnlp.org/research/sp`.

of most labels can be inferred from their names. We hire 3 annotators to annotate chunk boundaries and chunk labels for each Chinese address following the annotation guideline. In order to maintain high annotation quality, we also hire 2 additional quality controllers to sample 20 sentences from each batch of 1,000 annotated sentences for human evaluation. Re-annotation for that batch will be performed should the accuracy of human evaluation fall below 95%.

We randomly split the annotated data into 3 portions following the ratio of 60%, 20%, and 20%, yielding training, development, and test sets. The complete statistics of our data can be found in Table 1. From the table we can observe that the chunk label POI (point of interest) occurs most frequently. Indeed, such a label has a high level of importance. This is because location-based information can be extracted from such chunks, which is crucial for recommendation services (Gao et al., 2015; Xie et al., 2016). In addition, we report the number of distinctive chunks (unique#) that appear in the data for each label, from which we can see our corpus has a good coverage on PROVINCE, CITY, and DISTRICT[2].

We empirically assign each label a order ID indicating its level of specificity. For example, the label COUNTRY is used for describing a country, and is the most general concept. It is thus assigned the order ID 20, which is the highest among all labels. As another example, the label PERSON gets assigned an order ID 3, as it is used to describe one of the most specific concepts. Such order ID information will be useful later when designing our models for Chinese address parsing.

3 Approach

Our objective is to design a model for parsing Chinese addresses into semantically meaningful structures in the form of consecutive chunks, where each chunk is assigned a label as described in the previous section. As we have mentioned before, we believe there exist Chinese address-specific characteristics associated with address texts that can be exploited in designing a parsing model. Specifically, we argue there are two types of structured information within Chinese addresses that can be exploited when designing our parser – the *latent tree structures* and the *regular chain structures*. The former is used for capturing rich dependencies among chunks that appear towards the end of each address. The latter is used for capturing the structural patterns associated with chunks appearing at the beginning of each address.

3.1 Latent Tree Structures

We focus our discussions on the latent tree structures first. Given a consecutive sequence of labeled chunks, we can construct a binary tree structure whose yield exactly corresponds to the sequence of labeled chunks. We build the latent tree structures to capture complex dependencies based on the observation that chunks appearing towards the end of a given address do not follow a rigorous order. For instance, as we can see in the second example in Figure 1, chunks towards the end of the address consist of some labels related to numbers as well as the label REDUNDANT. These labels are either optional or do not follow some regular patterns in terms of order, which makes capturing dependencies among labels challenging.

We first introduce *auxiliary labels* based on the set of *original labels* we defined in the previous section. Such auxiliary labels are assigned to the internal nodes within a parse tree. Specifically, for each original label X, we introduce the auxiliary label $\overline{\text{X}}$. For example, the auxiliary label for ROAD would be $\overline{\text{ROAD}}$.

We now illustrate how a latent tree is constructed from a sequence of labeled chunks. These chunks will be regarded as a sequence of leaf nodes, each of which contains the corresponding chunk boundary and chunk type information. To simplify the construction process, we focus on building a specific type of binary trees with each non-leaf node containing at least 1 leaf node as one of its child.[3] We start the process by selecting any chunk first as one leaf node. Next we take a chunk that is either on the left or on the right of the selected chunk as its binary sibling node, and create a parent node by assigning the two selected leaf nodes as child nodes. To determine the label of the newly created parent node, we choose the auxiliary label based on the label with a higher order ID between the two of the child nodes. The newly created parent node will replace the 2 child nodes in the sequence and now the parent node becomes a selected node. We repeat this construction pro-

[2]We found these numbers are comparable with statistics on www.stats.gov.cn.

[3]Preliminary results show that considering arbitrary binary trees would lead to slightly worse results for our task.

cess until there is only 1 node left in the sequence. Note that the construction process makes use of the label order information.

Figure 2 shows an example tree that the gold chunks correspond to. From the example we can see that the non-leaf node label $\overline{\text{POI}}$ that appears twice has connections to other non-leaf node labels such as $\overline{\text{ROADNO}}$ and $\overline{\text{POI}}$. Such tree structures will allow us to capture rich Chinese address-specific structural information among labels. Since there are many latent trees corresponding to the given address consisting of consecutive labeled chunks, the model is facilitated to learn such complicated patterns, which is potentially beneficial for the address parsing task.

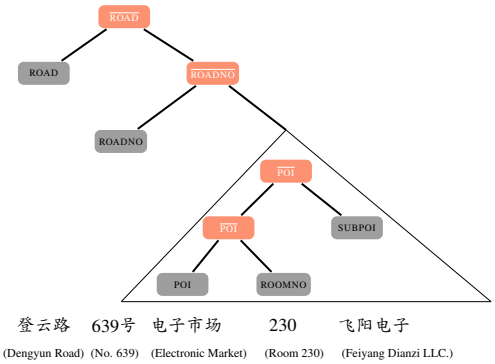

Figure 2: An example latent tree for given gold chunks where sp = POI. The English translation is listed below each chunk. Leaf nodes are in gray, and the internal nodes are in pink (labeled with white auxiliary labels). The tree structure within the triangle is latent – we show one of the many possible structures for illustration only.

3.2 Regular Chain Structures

The latent tree structures allow complex dependencies between different chunks to be captured within a Chinese address. Such dependencies would be helpful when there exist irregular patterns within an address. However, if we believe there are regular patterns among the labeled chunks, using an alternative assumption on the dependencies to properly capture such patterns may be more desirable. For instance, the first example in Figure 1 illustrates a common regular pattern at the beginning of the address, which is the order of (PROVINCE, CITY, DISTRICT). This motivates us to employ an alternative representation for capturing dependencies within chunks that appear at the beginning of the addresses, which are believed to exhibit more regular patterns.

Specifically, we employ a chain structure to capture the dependencies between adjacent labeled chunks. For example, given a sequence of chunks, we may always consider a right-branching tree structure to connect all these chunks. The resulting structure will be able to capture first-order dependencies between adjacent labeled chunks, which allows the regular orders among the labels to be learned. For example, consider the first two chunks that appear within the address as illustrated in Figure 2. The first two chunks form a right-branching tree structure. The construction process for such chain structures is similar to that of the latent trees, except that there is a single fixed (right-branching tree) structure for given labeled chunks.

Based on the observation that regular patterns appear mostly at the beginning of an address, we define the space $\mathcal{H}(x, y, sp)$ that consists of all latent tree structures that are consistent with the input character sequence x, the gold labeled chunks y and sp which determines the split point. Formally, we define the split point of a given address as specified by sp as the left boundary of the rightmost chunk whose label order ID is larger than or equal to sp. The split point divides the chunks into two groups – those appearing on the left of sp will form a chain structure while those on the right will form a tree structure where the correct construction is latent. Both structures are then merged to form a single representation, which is used for building our address parsing model.

Notice that when sp is set to -1 (denoted as sp = LAST), the split point is on the right of the last chunk. In this case the latent structured space $\mathcal{H}(x, y, sp)$ consists of only one single right-branching tree. On the other hand, when sp is set to its maximal value 20, the label order ID of COUNTRY (denoted as sp = COUNTRY), the latent structured space does not contain any structure that involves a partial regular chain component. Different values of sp leads to different interpolations between the two types of structural assumptions, resulting in different variants of our models. We will discuss the effect of different sp values in the experiments section.

3.3 Chunk Representation

A parse tree corresponds to a collection of labeled chunks as leaves. We adopt a bi-directional LSTM over a given input to compute the span-level representation. At each position i in the orig-

inal input consisting of a sequence of characters, we use $\mathbf{f}_i$ and $\mathbf{b}_i$ to denote the outputs of forward LSTM and backward LSTM respectively. We use $\mathbf{c}_{i,j} = [\mathbf{f}_j - \mathbf{f}_i; \mathbf{b}_i - \mathbf{b}_j]$ to denote the vector representation of the span covering characters from position i to position j (Wang and Chang, 2016). Motivated by Stern et al. (2017), we define the label score as follows:

$$\mathbf{s}(i, j) = F(\mathbf{c}_{i,j})$$

where F is a 2-layer feed-forward neural network with output dimension being the number of chunk labels. In addition, we denote the score of the span with a specific label l as the value of the l-th element in the vector $\mathbf{s}(i, j)$:

$$s(i, j, l) = [\mathbf{s}(i, j)]_l \tag{1}$$

3.4 Model

Inspired by Stern et al. (2017), we build a chart-based parsing model. Unlike that work, however, our model involves latent structures as mentioned in Section 3.1. For a given sequence of labeled chunks, our model considers all possible constituent trees whose yield are exactly the labeled chunks.

Consider a tree $\mathbf{t}$ that can be represented by a set of labeled spans, where each span is uniquely defined by the boundary (i, j) and the label l:

$$\mathbf{t} := \{(i_n, j_n, l_n) : n = 1, \ldots, |\mathbf{t}|\}. \tag{2}$$

The score of the tree $\mathbf{t}$ can be defined as follows:

$$\mathcal{S}(\mathbf{t}) = \sum_{(i,j,l)\in\mathbf{t}} [s(i, j, l)] \tag{3}$$

Similar to (Stern et al., 2017), we use a CKY-style algorithm to calculate the score of the optimal sub-tree that spans the interval (i, j) recursive using the following formula:

$$\begin{aligned}\pi(i, j) = &\max_l [s(i, j, l)] + \\ &\max_k \Big\{ \max_l [s(i, k, l)] + \pi(k, j), \\ &\quad \pi(i, k) + \max_l [s(k, j, l)] \Big\}\end{aligned} \tag{4}$$

The base case is when the text span (i, j) corresponds to a leaf node (a chunk) in the tree; in this case we have: $\pi(i, j) = \max_l [s(i, j, l)]$.

3.5 Training and Decoding

Inspired by the structural support vector machines with latent variables (Yu and Joachims, 2009), we employ a (per instance) hinge loss during training:

$$L = \max_{\mathbf{t}\in\mathcal{H}(x)} [0, \mathbf{\Delta}(\mathbf{t}, \mathbf{t}^*) + \mathcal{S}(\mathbf{t}) - \mathcal{S}(\mathbf{t}^*)] \tag{5}$$

where $\mathcal{H}(x)$ refers to the set of all possible trees for the given input x, and $\mathbf{t}^*$ denotes the best tree in the latent space $\mathcal{H}(x, y, sp)$:

$$\mathbf{t}^* = \max_{\mathbf{t}'\in\mathcal{H}(x,y,c)} \mathcal{S}(\mathbf{t}') \tag{6}$$

Here $\mathbf{\Delta}(\mathbf{t}, \mathbf{t}^*)$ represents the Hamming loss on labeled spans, measuring the similarity between the predicted tree and the best latent tree that corresponds to the gold chunks.

During decoding, we aim to obtain the best tree as the prediction $\hat{\mathbf{t}}$ for a new address x' among all the possible trees:

$$\hat{\mathbf{t}} = \arg\max_{\mathbf{t}\in\mathcal{H}(x')} [\mathcal{S}(\mathbf{t})] \tag{7}$$

The yield of the predicted tree $\hat{\mathbf{t}}$ gives us the list of labeled chunks.[4]

4 Experimental Setup

We call our model Address Parser with Latent Trees (**APLT**). We conducted experiments based on different settings of the sp values, leading to many model variants. We describe baselines, model hyperparameters as well as evaluation metrics in this section.

Baselines To understand the effectiveness of our models, we build the following baselines:

- ℓ**CRF** is the standard first-order linear CRF model (Lafferty et al., 2001) with discrete features for sequence labeling tasks.
- s**CRF** is based on the standard semi-Markov CRF (Sarawagi and Cohen, 2004) with discrete features[5].
- **LSTM** is the standard bi-directional LSTM model for sequence labeling tasks.

[4] In some cases, it is possible to predict a tree with one or more leaf chunks labeled with auxiliary labels (e.g., $\overline{\text{ROAD}}$). We have a post-processing step that converts such labels into their corresponding original labels (e.g., ROAD).

[5] See the supplementary material for details on the features for ℓ**CRF** and s**CRF**. For s**CRF** (and **LSTM-**s**CRF**), maximal chunk length is set to 36, which is the length of the longest chunk appearing in the training set.

- **LSTM-ℓCRF** is proposed by Lample et al. (2016) which is the state-of-the-art for many sequence labeling tasks
- **LSTM-sCRF** is based on segmental recurrent neural network (Kong et al., 2016) which is the neural network version of semi-Markov CRF (Sarawagi and Cohen, 2004).
- **TP** is a transition-based parser for chunking based on Lample et al. (2016), which makes use of the *stack LSTM* (Dyer et al., 2015) to encode the representation of the stack.

Hyperparameters We conducted all the experiments based on our *Chinese Address* corpus. We pre-trained Chinese character embeddings based on the Chinese Gigaword corpus (Graff and Chen, 2005), using the skip-gram model with hierarchical softmax implemented within the *word2vec* toolkit (Mikolov et al., 2013) where we set the sample rate to 10^{-5} and embedding size to 100.

We use a 2-layer LSTM (for both directions) with a hidden dimension of 200. For optimization, we adopt the Adam (Kingma and Ba, 2014) optimizer to optimize the model with batch size 1 and dropout rate 0.4. We randomly replace the low frequency words with the *UNK* token and normalize all numbers by replacing each digit (including Chinese characters representing numbers from 0-9) to 0. We train our model for a maximal of 30 epochs and select the model parameters based on the F_1 score after each epoch on the development set. The selected model is then applied to the test set for evaluation. Our model, as well as the baseline neural models, are implemented using DyNet (Neubig et al., 2017). All the neural weights are initialized following the default initialization method used in DyNet.

Evaluation Metrics We use the standard evaluation metrics from the CoNLL-2000 shared task (Tjong Kim Sang and Buchholz, 2000), reporting precision ($P.$), recall ($R.$) and F_1 percentage scores.

5 Result and Discussion

5.1 Main Results

We present our main results in Table 2, where we report the overall performance as well as specific results on the POI label. For our model, we report results for sp=20, -1 as two special cases – the former learns latent tree structures only and the latter assumes a single right-branching tree. We also report results for sp=7 which is selected based on the optimal results on the development set.

Model	POI			OVERALL		
	$P.$	$R.$	F_1	$P.$	$R.$	F_1
ℓCRF	69.76	72.68	71.19	87.78	85.33	86.53
sCRF	74.95	77.14	76.03	88.64	87.36	87.99
LSTM	70.11	76.90	73.35	85.63	88.11	86.85
LSTM-ℓCRF	77.94	75.62	76.76	88.83	88.88	88.86
LSTM-sCRF	77.80	77.84	77.82	89.21	88.52	88.86
TP	77.61	75.67	76.63	88.80	88.75	88.77
APLT $sp = 20$ (COUNTRY)	**80.36**	78.46	79.40	90.10	88.64	89.37
APLT $sp = -1$ (LAST)	79.64	78.89	79.26	90.06	89.07	89.56
APLT $sp = 7$ (HOUSENO)	79.75	**79.26**	**79.51**	**90.65**	**89.21**	**89.93**

Table 2: Main results.

Among all the baselines, **LSTM-ℓCRF** performs better than **LSTM** and **TP**, which is consistent with the finding reported in (Lample et al., 2016). The two models **LSTM-ℓCRF** and **LSTM-sCRF** both achieve similar results, which is also consistent with the finding reported in (Liu et al., 2016). The two non-neural models **ℓCRF** and **sCRF** perform substantially worse than their neural counterparts, which we believe is mainly due to the use of only handcrafted features in such systems. All these baseline models are capable of encoding transition patterns between neighboring chunks, which can partially capture certain structural information. However, certain Chinese address-specific structural information is not explicitly captured in such models.

Our model **APLT** (sp=7) achieves the best overall results, as well as the best results when evaluated on POI only. Compared with the strongest baselines **LSTM-ℓCRF** and **LSTM-sCRF**, **APLT** (sp=7) outperforms them significantly by more than 1 F_1 point overall ($p < 10^{-5}$)[6]. Furthermore, the **APLT** (sp=7) model obtains the best F_1 scores among all the models on POI. Note that our **APLT** model is able to learn richer dependencies among labels including label order information, regular patterns and irregular patterns among labels. Overall, the model **APLT** (sp=7) also outperforms both **APLT** (sp=-1) ($p < 0.05$) and **APLT** (sp=20) ($p < 0.005$) significantly. Such a result implies the importance of capturing the various Chinese address-specific structural information mentioned above within our model.

To understand the results better, we conduct detailed analysis of our results. Table 3 shows the F_1 scores of each label as well as the percentage of each label in the test data among four

[6]We perform the bootstrap resampling significant test.

Label	%	LSTM ℓCRF	LSTM sCRF	APLT sp=−1	APLT sp=7
Overall	100	88.86	88.86	89.56	**89.93**
POI	12.70	76.76	77.82	79.26	**79.51**
DISTRICT	11.55	95.04	95.04	95.46	**96.12**
ROAD	10.78	94.76	94.33	**95.28**	95.03
CITY	9.66	96.25	95.99	96.80	**97.03**
ROADNO	8.39	**95.32**	95.06	94.37	94.74
TOWN	7.83	92.07	92.05	**93.09**	92.90
PROVINCE	7.57	97.91	97.69	98.30	**98.42**
REDUNDANT	6.81	83.54	82.43	84.80	**85.98**
HOUSENO	5.65	90.62	90.07	89.83	**91.30**
ROOMNO	4.93	**91.15**	90.60	90.47	91.11
SUBPOI	2.92	57.70	**60.63**	60.58	59.41
COMMUNITY	2.49	74.79	75.06	76.53	**76.58**
CELLNO	2.14	92.29	91.01	**92.35**	90.78
FLOORNO	1.98	**98.03**	97.01	96.96	97.59
ASSIST	1.44	77.64	77.73	73.78	**78.32**
PERSON	1.25	61.58	61.68	61.69	**63.92**
SUBROAD	0.78	77.11	73.36	**80.00**	75.81
DEVZONE	0.64	63.85	63.11	**66.67**	64.13
SUBROADNO	0.45	70.50	63.24	**71.01**	67.65
COUNTRY	0.08	**96.00**	**96.00**	**96.00**	**96.00**
OTHERINFO	0.02	0.00	0.00	0.00	0.00

Table 3: F_1 score comparison on test data for each label among 4 models as well as the percentage of each label in the gold data.

models **LSTM-ℓCRF**, **LSTM-sCRF**, **APLT** (sp=−1) and **APLT** (sp=7). Note that the results for the top 4 labels POI, DISTRICT, ROAD and CITY, which take up 45% of total chunks, all get improved when using our **APLT** models. Moreover, it achieves better or comparable $F1$ scores on 15 labels in the table among the total 21 labels, especially on POI, DISTRICT, REDUNDANT, COMMUNITY and PERSON with at least 1 point improvement in F_1. Interestingly, our models perform worse than **LSTM-ℓCRF** on labels such as ROADNO, ROOMNO, and FLOORNO, which are mostly related to numbers. We note that, however, chunks with such labels do not constitute a large proportion of all chunks. Results suggest that our models somehow learned to focus on optimization performance for chunks with more prominent labels such as POI and DISTRICT.

5.2 Effectiveness of Structural Information

In order to investigate how tree structures affect the final performance, we also conducted experiments with different values for sp, which is used for determining the split point. Figure 3 shows the moving-averaged F_1 scores on the test set obtained when choosing sp around specific values (a similar distribution can be observed on the development set). From the bottom (COUNTRY,20) to the top (LAST,-1) along y axis, the lower the sp is, the more constraints are applied to the latent space $\mathcal{H}(x, y, sp)$. Note that when sp=−1 (LAST), the gold input only corresponds to a single right-branching tree. We exclude the following labels: REDUNDANT, ASSIST and OTHERINFO, because we found these labels may appear at any place within a given address, which make them unsuitable for determining the split point.

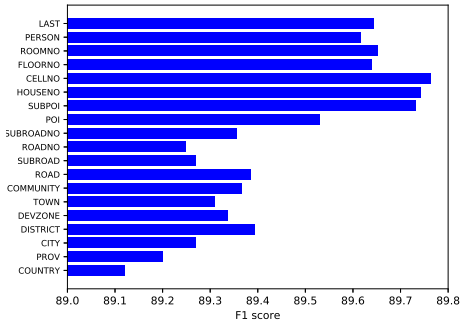

Figure 3: Effect of sp.

From Figure 3 we can observe that the F_1 score generally increases as we decrease sp, starting from COUNTRY(with order ID 20). The performance reaches the maximum when the sp is set to a value within the range [SUBPOI, CELLNO]. This observation implies that there does exist ordering information among labels, and introducing more constraints on the latent space will have the benefit of modeling the regular patterns around the beginning part of a given address. After reaching the best value, as we further decrease sp, the performance drops slightly and oscillates around the range [FLOORNO, LAST]. From here we can observe that the latent trees are able to help capture irregular patterns within labels that appear towards the end of the address. Overall, these results suggest the importance of designing a model like ours that is capable of capturing Chinese address-specific characteristics.

5.3 Error Analysis

We conduct error analysis on two strongest baselines **LSTM-ℓCRF** and **LSTM-sCRF** as well as two best-performing **APLT** models respectively. We examined the list of top-10 labels with most errors for each model, and found most of the errors come from labels such as POI, SUBPOI and REDUNDANT – this implies they are the most challenging labels for this task. We also found labels such as ROOMNO appear in the list for **APLT** models, but not for the **LSTM-ℓCRF** model, showing that **APLT** models are still not good at handling numbers as we discussed above.

There are two major types of errors. The type-

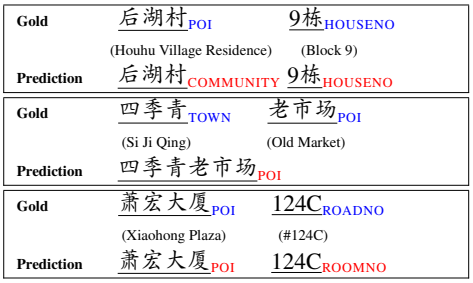

Figure 4: Example outputs from **APLT**(sp=7).

Length	%	**LSTM** ℓ**CRF**	**LSTM** s**CRF**	**APLT** sp=−1	**APLT** sp=7
1	9.39	**92.16**	92.14	91.65	91.77
2	**23.73**	86.69	86.13	87.16	**87.77**
3	**44.60**	92.26	92.04	93.03	**93.51**
4	**13.31**	86.49	87.48	88.05	**88.18**
5	3.70	74.57	76.41	77.55	**79.43**
6	2.14	68.88	70.19	70.87	**73.73**
7	1.16	64.61	68.14	67.59	**68.22**
≥8	1.97	63.31	62.57	**63.33**	60.19

Table 4: Results for different chunk lengths.

I error refers to the case where the boundary of a chunk is predicted correctly but not its label. The type-II error is the case where even the boundary of a predicted chunk is incorrect. We found that **APLT** (sp=−1) and **APLT** (sp=7) produce less type-I errors (45.04% and 42.95% respectively) than **LSTM-**ℓ**CRF** and **LSTM-**s**CRF** (49.87% and 47.26% respectively). Moreover, we find that **APLT** (sp=7) model produces the least number of type-I errors as well as type-II errors.

Looking into the type-I errors of both two **APLT** models, we find chunks with label POI are often incorrectly labeled as COMMUNITY, which is a major source of errors (9% of total errors). As a typical example, we show a partial prediction in Figure 4, where our model fails to recognize "后湖村"(Houhu Village Residence) as a POI. Here the character "村"(Village) is a common suffix for the name of either a village or a residence, hence the confusion.

The second example in Figure 4 demonstrates another typical kind of errors produced by our models around the POI labels. Here, "四季青(Si Ji Qing)" is actually the name of a town. However, as most names of towns end with "镇(Town)" as the suffix, our models as well as baseline models all fail to identify the correct chunk boundaries.

We also investigate the errors around the number labels. We choose to look into the results on ROADNO because it is the fifth most popular label in the test data. Based on the error analysis, we found that many chunks of label ROADNO were incorrectly assigned other types of number labels. As we can see from the third example in Figure 4, the ROADNO "124C" is incorrectly predicted as a ROOMNO. Indeed, this chunk does look like a room number, though in fact it refers to a road within a "plaza" (大厦) rather than an office within a "building" (another interpretation of 大厦). From these examples we can observe that many ambiguities may not be easily resolved without further background knowledge.

5.4 Robustness Analysis

We analyze the model robustness by assessing the performance on chunks of different lengths for each of the four models discussed above. We group chunks into 8 categories based on their lengths and present the results in Table 4 where the distribution information is also included. As we can see, all the models achieve at least a F_1 score of 86 when considering chunks whose lengths are less than 5. As the length increases, the performance of all models drop gradually. For chunks whose lengths are at least 8, the F_1 score is around 60-63 for all models. Considering chunks whose lengths are either 2, 3, or 4 only (such chunks constitute over 80% of total chunks), we can observe that **APLT** (sp=7) outperforms two baselines significantly by more than 1 point for each category. These results demonstrate the robustness of our model when handling chunks of different lengths.

Comparing the two **APLT** models, we can see the model **APLT** (sp=7) outperforms **APLT** (sp=−1) for each chunk category, except for chunks whose lengths are greater than or equal to 8. These two models differ in their latent spaces. **APLT** (sp=7) with a richer latent space appears to be better at handling chunks with short or medium lengths.

In addition, we conducted a further experiment to understand how each model is able to handle new chunks – the chunks that appear in the test set (according to the gold labels) but do not appear in the training set. We found empirically there are 31% of the chunks in the test set that are new chunks. Such an experiment allows us to assess the robustness of each model when new data is available. We report the accuracy for the new chunks in Table 5. As we can see, two **APLT** models outperform two baselines, indicating our **APLT** models appear to be better at handling new chunks. We believe this is due to the tree models

LSTM ℓCRF	LSTM sCRF	APLT sp=−1	APLT sp=7
80.17	79.92	80.94	80.94

Table 5: Accuracy on test data for the new chunks.

that we used, which are capable of capturing complex dependencies among chunks.

6 Related Work

While the Chinese address parsing task is new, it is related to the following traditional tasks within the field of natural language processing (NLP) – chunking, named entity recognition, word segmentation and parsing. We briefly survey research efforts which are most related to our task below.

Chunking as a fundamental task in NLP has been investigated for decades (Abney, 1991). Chunking for Chinese can typically be regarded as a sequence labeling problem solvable by models such as conditional random fields (Chen et al., 2006; Tan et al., 2005; Zhou et al., 2012), hidden Markov models (Li et al., 2003), support vector machines (Tan et al., 2004) and the maximum entropy model (Wu et al., 2005). Our task can also be regarded as a chunking task where we need to assign an address-specific label to each chunk.

Named entity recognition (NER) is another fundamental task close to chunking within the field of NLP, which focuses on the extraction of semantically meaningful entities from the text. The state-of-the-art approach by Lample et al. (2016) employs a LSTM-CRF model. Ma and Hovy (2016) proposed a LSTM-CNNs-CRF model that utilizes convolutional neural networks (CNNs) to extract character-level features besides word-level features. Zhai et al. (2017) suggested a neural chunking model based on pointer networks (Vinyals et al., 2015) to resolve the issue of being difficult to use chunk-level features such as the length of the chunk for segmentation. Zhang and Yang (2018) tackled the problem of Chinese NER by deploying a lattice LSTM leveraging lexicons.

Another task closely related to our task is the Chinese word segmentation task which at least dates back to the 1990s (Sproat et al., 1994). The segmentation task is typically casted as a character-based sequence labeling problem (Xue, 2003) which can be solved by CRF based models (Peng et al., 2004; Zhao et al., 2006), their latent-variable variants (Sun et al., 2009), or max-margin based models (Zhang and Clark, 2007). Recently, Zhang et al. (2016) proposed a neural transition-based segmentation approach by encoding both words and characters as well as the history action sequence. Yang et al. (2017) suggested to perform segmentation with a neural transition-based method with rich pre-training.

Constituent parsing is another line of work that is related to our task. The state-of-the-art approaches to parsing include transition-based models (Dyer et al., 2016) and chart-based models (Stern et al., 2017; Kitaev and Klein, 2018).Our model is motivated by the latter approaches, where we additionally introduce latent variables for capturing complex dependencies among chunks.

7 Conclusion

In this work, we introduce a new task – *Chinese address parsing*, which is to segment a given Chinese address text into chunks while assigning each chunk a semantically meaningful label. We create and publish a *Chinese address* corpus that consists of 15K fully labeled Chinese addresses. We identify interesting characteristics associated with the task and design a novel neural parsing model with latent variables for this task, which is able to capture Chinese address-specific structural information. We conduct extensive experiments and compare our approach with strong baselines through detailed analysis. We show that our proposed model outperforms baseline approaches significantly, due to its ability in capturing rich structural information present in the Chinese addresses.

Future work includes leveraging external knowledge bases to disambiguate chunks and entities that appear within Chinese addresses, as well as designing algorithms that are able to capture longer-range dependencies among chunks using alternative structures.

Acknowledgments

We would like to thank the anonymous reviewers for their constructive comments on this work. This work is done under a collaborative agreement between SUTD and Alibaba on an Alibaba Innovative Research (AIR) Program funded by Alibaba, where Alibaba provided data. We appreciate Alibaba's generosity in the agreement that makes it possible for us to make all data and code in this research publicly available upon acceptance of this paper. This work is also partially supported by SUTD project PIE-SGP-AI-2018-01.

References

Steven P Abney. 1991. Parsing by chunks. In *Principle-based parsing*, pages 257–278. Springer.

Marco Avvenuti, Stefano Cresci, Leonardo Nizzoli, and Maurizio Tesconi. 2018. Gsp (geo-semantic-parsing): Geoparsing and geotagging with machine learning on top of linked data. In *Proc. of ESWC*.

Wenliang Chen, Yujie Zhang, and Hitoshi Isahara. 2006. An empirical study of chinese chunking. In *Proc. of ACL*.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *Proc. of ACL*.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A Smith. 2016. Recurrent neural network grammars. In *Proc. of NAACL*.

Huiji Gao, Jiliang Tang, Xia Hu, and Huan Liu. 2015. Content-aware point of interest recommendation on location-based social networks. In *Proc. of AAAI*.

David Graff and Ke Chen. 2005. Chinese gigaword. *LDC Catalog No.: LDC2003T09, ISBN*, 1:58563–58230.

Ya-Hui Jia, Wei-Neng Chen, Tianlong Gu, Huaxiang Zhang, Huaqiang Yuan, Ying Lin, Wei-Jie Yu, and Jun Zhang. 2017. A dynamic logistic dispatching system with set-based particle swarm optimization. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *Proc. of ICLR*.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proc. of ACL*.

Lingpeng Kong, Chris Dyer, and Noah A Smith. 2016. Segmental recurrent neural networks. In *Proc. of ICLR*.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proc. of NAACL*.

Heng Li, Jonathan J Webster, Chunyu Kit, and Tianshun Yao. 2003. Transductive hmm based chinese text chunking. In *Natural Language Processing and Knowledge Engineering, 2003. Proceedings. 2003 International Conference on*, pages 257–262. IEEE.

Yijia Liu, Wanxiang Che, Jiang Guo, Bing Qin, and Ting Liu. 2016. Exploring segment representations for neural segmentation models. In *Proc. of IJCAI*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proc. of ACL*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proc. of NIPS*.

Graham Neubig, Chris Dyer, Yoav Goldberg, Austin Matthews, Waleed Ammar, Antonios Anastasopoulos, Miguel Ballesteros, David Chiang, Daniel Clothiaux, Trevor Cohn, et al. 2017. Dynet: The dynamic neural network toolkit. *arXiv preprint arXiv:1701.03980*.

Fuchun Peng, Fangfang Feng, and Andrew McCallum. 2004. Chinese segmentation and new word detection using conditional random fields. In *Proc. of ACL*.

Sunita Sarawagi and William W Cohen. 2004. Semi-markov conditional random fields for information extraction. In *Proc. of NIPS*.

Richard Sproat, Chilin Shih, William Gale, and Nancy Chang. 1994. A stochastic finite-state word-segmentation algorithm for chinese. In *Proc. of ACL*.

Mitchell Stern, Jacob Andreas, and Dan Klein. 2017. A minimal span-based neural constituency parser. In *Proc. of ACL*.

Xu Sun, Yaozhong Zhang, Takuya Matsuzaki, Yoshimasa Tsuruoka, and Jun'ichi Tsujii. 2009. A discriminative latent variable chinese segmenter with hybrid word/character information. In *Proc. of NAACL*.

Yongmei Tan, Tianshun Yao, Qing Chen, and Jingbo Zhu. 2004. Chinese chunk identification using svms plus sigmoid. In *Proc. of IJCNLP*.

Yongmei Tan, Tianshun Yao, Qing Chen, and Jingbo Zhu. 2005. Applying conditional random fields to chinese shallow parsing. In *Proc. of CICLing*.

Erik F Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the conll-2000 shared task: Chunking. In *Proc. of CoNLL*.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Proc. of NIPS*.

Wenhui Wang and Baobao Chang. 2016. Graph-based dependency parsing with bidirectional lstm. In *Proc. of ACL*.

Shih-Hung Wu, Cheng-Wei Shih, Chia-Wei Wu, Tzong-Han Tsai, and Wen-Lian Hsu. 2005. Applying maximum entropy to robust chinese shallow parsing. In *Proc. of IJCLCLP*.

Min Xie, Hongzhi Yin, Hao Wang, Fanjiang Xu, Weitong Chen, and Sen Wang. 2016. Learning graph-based poi embedding for location-based recommendation. In *Proc. of CIKM*.

Nianwen Xue. 2003. Chinese word segmentation as character tagging. *International Journal of Computational Linguistics & Chinese Language Processing, Volume 8, Number 1, February 2003: Special Issue on Word Formation and Chinese Language Processing*, 8(1):29–48.

Jie Yang, Yue Zhang, and Fei Dong. 2017. Neural word segmentation with rich pretraining. In *Proc. of ACL*.

Chun-Nam John Yu and Thorsten Joachims. 2009. Learning structural svms with latent variables. In *Proc. of ICML*.

Feifei Zhai, Saloni Potdar, Bing Xiang, and Bowen Zhou. 2017. Neural models for sequence chunking. In *Proc. of AAAI*.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2016. Transition-based neural word segmentation. In *Proc. of ACL*.

Yue Zhang and Stephen Clark. 2007. Chinese segmentation with a word-based perceptron algorithm. In *Proc. of ACL*.

Yue Zhang and Jie Yang. 2018. Chinese ner using lattice lstm. In *Proc. of ACL*.

Hai Zhao, Chang-Ning Huang, Mu Li, and Bao-Liang Lu. 2006. Effective tag set selection in chinese word segmentation via conditional random field modeling. In *Proc. of PACLIC*.

Junsheng Zhou, Weiguang Qu, and Fen Zhang. 2012. Exploiting chunk-level features to improve phrase chunking. In *Proc. of EMNLP*.