Computer Modeling in
Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2021.014802

ARTICLE

Number Entities Recognition in Multiple Rounds of Dialogue Systems

Shan Zhang[1], **Bin Cao**[1], **Yueshen Xu**[2,*] **and Jing Fan**[1]

[1]School of Computer Science and Technology, Zhejiang University of Technology, Hangzhou, 310023, China

[2]School of Computer Science and Technology, Xidian University, Xi'an, 710071, China

[*]Corresponding Author: Yueshen Xu. Email: ysxu@xidian.edu.cn

Received: 30 October 2020 Accepted: 02 February 2021

ABSTRACT

As a representative technique in natural language processing (NLP), named entity recognition is used in many tasks, such as dialogue systems, machine translation and information extraction. In dialogue systems, there is a common case for named entity recognition, where a lot of entities are composed of numbers, and are segmented to be located in different places. For example, in multiple rounds of dialogue systems, a phone number is likely to be divided into several parts, because the phone number is usually long and is emphasized. In this paper, the entity consisting of numbers is named as *number entity*. The discontinuous positions of number entities result from many reasons. We find two reasons from real-world dialogue systems. The first reason is the repetitive confirmation of different components of a number entity, and the second reason is the interception of mood words. The extraction of number entities is quite useful in many tasks, such as user information completion and service requests correction. However, the existing entity extraction methods cannot extract entities consisting of discontinuous entity blocks. To address these problems, in this paper, we propose a comprehensive method for number entity recognition, which is capable of extracting number entities in multiple rounds of dialogues systems. We conduct extensive experiments on a real-world dataset, and the experimental results demonstrate the high performance of our method.

KEYWORDS

Natural language processing; dialogue systems; named entity recognition; number entity; discontinuous entity blocks

1 Introduction

With the rapid development of E-Commerce, many functions have emerged, such as product recommendation systems [1–3], mobile marketing apps [4], product intelligent transportation systems [5] and intelligen customer service systems. In intelligent customer service systems, named entity recognition is a widely-used technique to extract entities as important information in a conversation. There is one type of entities consisting of numbers, such as telephone number, street number and ID number. Such entities exist widely in conversations and typically contain key information, and in this paper, we name such type of entity as number entity. Based on applications of intelligent customer service systems, we have the following observations from multiple rounds of dialogues between a user and a customer service agent (CSA). In a conversation, number entities are always divided into several parts, because they are usually long and emphasized. Meanwhile, in many cases, due to the interception of mood words, the divided number entity blocks are distributed in discontinuous positions. The extraction of number entities is quite useful in many tasks, such as user information completion, service requests correction and user information retrieval. However, the existing named entity recognition methods fail to extract number entity blocks in discontinuous positions and further fail to generate a complete entity.

In this paper, we aim to extract number entities that consist of entity blocks at discontinuous locations in multiple rounds of dialogues systems. Because the positions of number entity blocks are not continuous, we cannot directly extract entity blocks using traditional named entity recognition methods. We solve this extraction task as a sequence labeling task and model the determination of whether a word in a dialogue belongs to the number entity block as a classification problem. In a dialogue, besides of number entity blocks, there are other number entities that are typically not separated, such as currency entities and time entities. So we extract dialogue fragments containing number entity blocks to avoid interference from the unseparated number entities. Based on these ideas, we propose a holistic solution for number entities recognition, which is built based on hidden Markov SVM (HM-SVM)[1] [6] and conditional random field (CRF) [7]. We use HM-SVM model to locate the dialogue segments that contain number entity blocks. The dialogue texts need to be transformed as the input of HM-SVM model, and we use the BOW (Bag-of-word) model and Doc2vec (Document-to-vector) [8] model to convert sentences of dialogue texts into feature vectors, and we select the BOW model for dialogue segments extraction. The HM-SVM model extracts dialogue segments, and CRF model recognizes the number block entities. The number entity blocks are out of order, but the entity blocks that form the number entities are in order. Therefore, we propose rules to concatenate the partial number entities to produce complete number entities. In general, the contributions in this paper can be summarized as follows:

1. In multiple rounds of dialogue systems, we observe that the entities that are composed of numbers are divided into several parts and usually distributed in discontinuous positions. We find the reasons behind the observations and formally define the problem, i.e., how to effectively and efficiently extract the number entities.
2. We compare the impact of BOW model and Doc2Vec model on performance of dialogue fragments extraction. We find that although Doc2Vec model leverages more information of words than BOW model, the combination of BOW model and HM-SVM model has a higher performance on dialogue fragments extraction.
3. We conduct extensive experiments on a real-world dataset from China Telecom call center, and experimental results demonstrate the effectiveness of our method in number entity recognition.

The rest of this paper is organized as follows. The study of related work is given in Section 2. Section 3 introduces preliminaries and defines our task. Section 4 presents an overview of the proposed method. Sections 5 and 6 introduce the implementation details of our method. Experimental results are presented in Section 7. Finally, Section 8 concludes the paper.

[1] SVM is short for support vector machine.

2 Related Work

As an important task in natural language processing (NLP), named entity recognition is widely used in many tasks, such as information extraction, question-and-answering system, syntax analysis and machine translation. In addition to the traditional extraction of named entities, including persons, locations and organizations, new named entity recognition also exist and one representative is the extraction of number entity, such as phone numbers, ID numbers and dates [9]. As our work aims to extract number entities, in this section, we discuss the related work of number entities recognition.

Rule-based approaches. In early days, researchers developed rule-based and dictionary-based methods for named entity recognition [10]. von Brzeski et al. [11] used the regular expression to detect all types of entities, including phone numbers and e-mail addresses. In [12], the authors developed a system to extract information by dividing sentences into noun groups and verb groups. Using the rule-based approach for date and time entities, Stern [13] provided a calendar on the visual interface and allowed users to find relevant messages according to date. However, rule-based approach typically require a lot of manual work to prepare rules and does not take all features of entities into account.

Machine learning approaches. As a machine learning technique, the CRF model is preferred for named entity recognition [14–16]. To identify named entities in Bulgarian, Georgiev et al. [17] presented a CRF model based on rich features that were already established in other languages. A named entity recognition system was constructed in [18] with CRF model. The system extracted many types of named entities, including date, currency and numbers based on contextual information and features in text. However, the existing CRF-based methods only can identify named entities that consist of consecutive entity blocks, For example, the exiting methods can extract number entity "*32454678865423*" from text "My ID number is *32454678865423*." Our proposed method can also identify the entity blocks in inconsecutive positions, such as a multiple rounds of dialogue "My ID number is *3245*," "ok, then?," "*4678*," "and?," "*865427*." The existing entity recognition methods can only extract "*3245*," "*4678*" and "*865427*" as three complete number entities. Our proposed method can extract them as a complete number entity "*32454678865423*."

Deep learning approaches. In recent years, for the power in deep features learning, the deep learning methods [19,20] have been used in named entity recognition tasks. Chiu et al. [21] combined LSTM (long short-term memory) and CNN (convolutional neural network) to learn words features, and extracted date, time, concurrency and other entities. Luoma et al. [22] improved the performance of named entity recognition by adding context to the input of BERT model. There are many approaches that combine the machine learning methods with the deep neural networks [23–27]. However, deep learning methods require a large amount of corpora to train models, which is not suitable in many real-world cases. The data we obtained from China Telecom call center is not enough to train a deep model based on the following two reasons. The first reason is the lack of manpower with professional background knowledge to label data. The other reason is that only in a few cases, clients need to inform or be informed phone numbers, ID numbers, account numbers, etc.

3 Preliminaries

In this section, we present a set of preliminaries that are important to understand our method. We first present the concept of bag-of-words (BOW) model, Doc2vec model and sequence labeling. We briefly introduce the HM-SVM model and CRF model, which are used to solve the sequence labeling problem. Finally, we define the task solved in the paper.

3.1 Bag-of-Words Model and Doc2vec Model

The bag-of-words (BOW) model is used to represent text data when we model text with machine learning models. Text is made up of a large number of words, so BOW model aims to use words to represent text. BOW model does not utilize the contextual information between words in the text and only considers the weight of words. The weight is related to how often a word appears in the text. Based on such an idea, BOW model builds a dictionary for N words that appear in all texts, and each word in the dictionary has a fixed position. We use an N-dimensional vector to represent a text. Each element of the vector represents the times that each word in the dictionary appears in the text.

The Doc2vec is an unsupervised model that can generate vector representation of sentences and documents, and can be regarded as an extension of word2vec [28]. The learned vector can find the similarity between sentences and documents by calculating the similarity. Take the document vector representation as an example. During training, each document is mapped into a vector space and is represented by a column of a matrix D. Each word in the document is also mapped into a vector space, which is represented by a column of a matrix W. The document vectors and word vectors are as the input to predict the next word in the document. The final document representation is generated by iteratively updating the document vector.

3.2 Sequence Labeling, HM-SVM and CRF

Sequence labeling refers to the assignment of labels to each member in a sequence of observed values. We formalize the sequence labeling as follows. We have the observation sequence $X = x_1, x_2, \ldots, x_n$ and label collection L. We perform sequence labeling for X to generate the results $Y = y_1, y_2, \ldots, y_n,\ y_i \in L$. Sequence labeling can be used in many typical tasks, such as word segmentation, part-of-speech tagging, named entity recognition, keyword extraction and word sense role tagging.

Hidden Markov Support Vector Machine (HM-SVM) is used to solve sequence labeling problem. The HM-SVM model combines hidden Markov model (HMM) and support vector machine (SVM). The basic HMM model only learns the relationship between the observation sequence and the corresponding label, and the dependencies between adjacent labels. Based on the basic HMM model, the HM-SVM model also considers the dependencies between the observation sequence and other labels within a range of the corresponding label. Besides, different from the basic HMM that defines a proper joint probability model, the HM-SVM model uses a kernel-centric approach inherited from SVM learning non-linear discriminant functions.

Conditional random fields (CRF) model has outstanding performance in named entity recognition. The CRF model uses feature functions to represent features more accurately. A feature function can be expressed as $f(X, i, y_i, y_{i-1})$, where X represents the input sequence, i is the current position, y_i is the current state and y_{i-1} represents the previous state. The feature function can learn the relationship between the current state and any observation, and the relationship between the current state and other states.

3.3 Task Definition of Number Entity Recognition

Based on the observation of conversations between users and customer service agents (CSA), we find that there are many number entities existing in real-world cases, and typical number entities include currency, time, ID numbers and phone numbers. Some number entities do not need to be divided due to their short content and can be identified as a complete entity. In this paper, we aim to extract complete number entities that are divided into number entity blocks at discontinuous positions in multiple rounds of dialogue systems. Fig. 1 shows an example of extracting phone number entity.

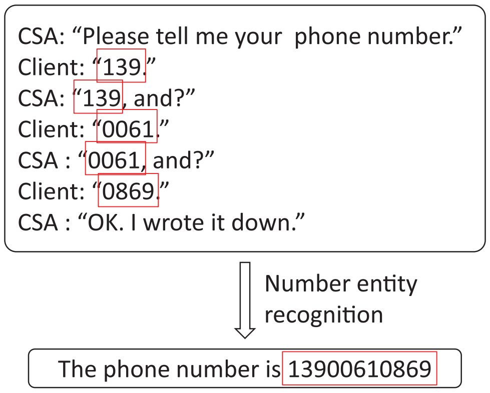
CSA: "Please tell me your phone number."
Client: "139."
CSA: "139, and?"
Client: "0061."
CSA : "0061, and?"
Client: "0869."
CSA : "OK. I wrote it down."
Number entity recognition
The phone number is 13900610869

Figure 1: A toy example of number entity extraction

4 The Whole Process of the Proposed Method

In this section, we introduce the complete process of the proposed method. The idea of our method is that we first locate the fragment in a dialogue text where number entity blocks are located, and we extract entity blocks from the fragment. We combine the entity blocks to generate the complete number entities. Based on the idea, the number entity recognition is divided into three phases shown in Fig. 2, which are *fragment locating*, *partial entity recognition* and *entity recovery*. First, we perform some preprocessing on noisy dialogue text. In Phase 1, we judge whether there is any situation of speaking numbers during a conversation. If the situation does not exist, we return the result that there is no number entities. If it exists, we locate the fragments which include number entity blocks, and further extract the number entity blocks in Phase 2. In Phase 3, we splice the obtained entitie blocks to obtain complete number entities and return them as the result. The details are described as follows.

The conversations between users and CSA are converted into texts, and we extract the number entities based on the conversation texts. As the conversation texts contain a lot of noise, we first preprocess the dialogue texts. There is a large number of stop words in dialogue texts. We leverage context information of a word when we decide whether the word belongs to a number entity block, so the stop words are likely to impact the identification accuracy of number block entities. Before we train models and extract number entities, we remove stop words. In the first phase, we locate the fragment that contains number entity blocks from a dialogue text. Our object is to extract number entities that are divided into several parts at discontinuous locations in a multiple rounds of dialogue systems. Meanwhile, a dialogue text also contains some number entities that do not need to be divided. For example, number entities that represent currency are usually short, so they usually are not separated in a conversation. Therefore, we locate fragments from a dialogue text to avoid interference with the extraction of number entity blocks. As shown in Fig. 2, if there is a fragment in a dialogue text, we locate the fragment. Otherwise, we directly return the result without number entity. We aim to extract partial entities that compose number entities in Phase 2. The number entity blocks in the fragment that is generated in *fragment locating* phase can be in many discontinuous locations. The entity blocks may also have the following situations. 1) To confirm the information, number entity blocks may be repeated. 2) Although we have done fragment locating, there may still be number entities that are not necessary to be divided. 3) The length of number entity blocks is not fixed, which is likely to cause confusion with other number entities. Fig. 3 shows an example of number entity blocks. We extract number entity blocks by fully considering all situations. Based on *partial entity recognition* phase, we obtain the partial entities that compose number entities. The number entity blocks contain the repeated entity blocks and are out of order, and the real number entities are in order. In Phase 3, we propose to design the concatenation rules, and combine the number entity blocks to generate complete and correct number entities with the rules.

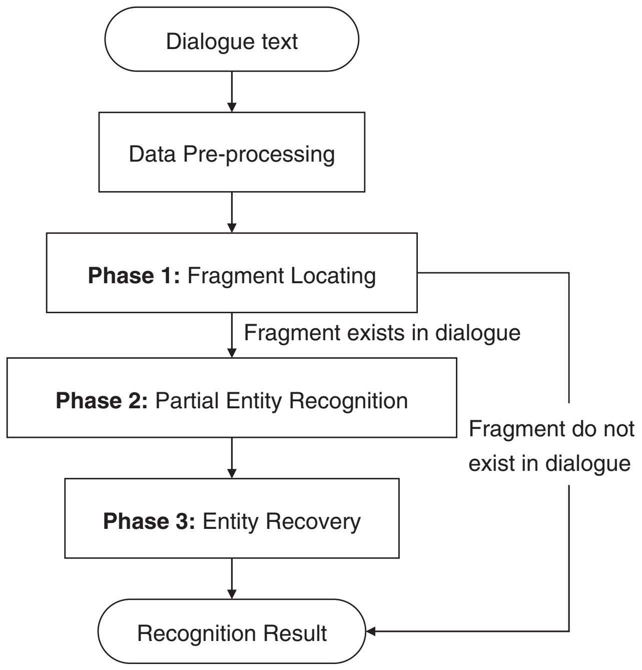
Dialogue text
Data Pre-processing
Phase 1: Fragment Locating
Fragment exists in dialogue
Phase 2: Partial Entity Recognition
Fragment do not exist in dialogue
Phase 3: Entity Recovery
Recognition Result

Figure 2: The process of number entity extraction

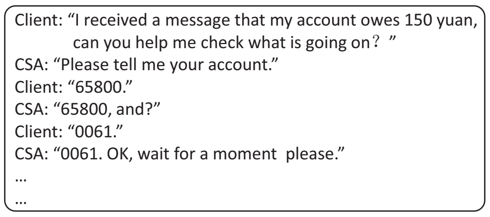
Client: "I received a message that my account owes 150 yuan, can you help me check what is going on？"
CSA: "Please tell me your account."
Client: "65800."
CSA: "65800, and?"
Client: "0061."
CSA: "0061. OK, wait for a moment please."
...
...

Figure 3: An example of situations of number entity blocks

5 Fragment Locating

In this section, we give the details of *fragment locating*. The goal is to determine whether there are number entity blocks in a dialogue text, if there is, the fragment will be extracted. We regard the fragment extraction task as a classification problem. The HM-SVM model is used to classify each sentence of a dialogue text to determine whether the sentence belongs to the number entity dialogue fragment. The input of the model is a conversation text between a user and the CSA. The output is classification result of each sentence in the text. The model building is as follows.

5.1 Dialogue Sequence Labeling

Before training the HM-SVM model, we need to label the data in training set. In sequence labeling tasks, there are many labeling schemes. As dialogue fragments have more than one sentence, we use the labeling scheme "BIOE" to label each sentence of dialogue texts in training set. "B" denotes the first sentence of number entity dialogue fragment. "I" denotes the sentence in the middle of fragment. "E" denotes the last sentence of fragment. "O" means that this sentence does not belong to the fragment. Tab. 1 shows an example of labeling results for one dialogue text in the training set. Because the sentence labeled "O" is not useful for number entity dialogue fragment extraction, we filter the sentence labeled "O", and output the sentence containing labels "B", "I" and "E" in order. The sentences compose the number entity dialogue fragment.

Table 1: Labeling results for the training data of HM-SVM model

"I want to check the phone bil"	O
"Please tell me your phone number"	B
"130"	I
"130, then?"	I
"1699"	I
"and?"	I
"2345"	E
"Ok, please wait a moment, your monthly consumption is 50 rmb"	O

5.2 Number Entity Dialogue Fragment Recognition

Each sentence of dialogue texts is necessary to convert into the input format of HM-SVM model. In this paper, we use the following two models to finish sentence embedding and train HM-SVM model.

Bag-of-words model. According to bag-of-words (BOW) model, we perform word segmentation on sentences in all texts and use the resulting words to build a dictionary. Based on the dictionary, we use the BOW model to convert each sentence into a sentence feature vector. Each dimension in the vector corresponds to each word in the dictionary, and the value of each dimension is the times that each word appears in the sentence.

Doc2vec model. We train the Doc2vec model to generate the word vector, the parameter of softmax and the sentence vector. We use the model to produce the vector of the new sentence.

The feature vector of each sentence in every conversation and the label are as the input of HM-SVM model. Note that, in the experiment, we compared the recognition performance of HM-SVM model with two embedding models. The one with the better performance is the number entity dialogue fragment extraction model.

6 Partial Entity Recognition and Entity Recovery

In this section, we give the details of *partial entity recognition* phase and *entity recovery* phase. Through the number entity dialogue fragment extraction, we obtain the fragment as the input. The next task is to extract number entity blocks from the fragment and combine them into complete number entities as the output. With sequence labeling, we classify each word in the fragment and decide whether it belongs to number entity blocks. In this paper, we use CRF model to extract number entity blocks. CRF model is to recognize entity according to feature functions that are built for features of the entity categories. Therefore, it is necessary to extract features of number entity blocks for CRF model to construct feature functions. First, we analyze the characteristics of number entity blocks in dialogue texts. We extract features for number entity blocks according to the characteristics. We design feature templates for the model to build feature functions. Finally, we train CRF model to extract number block entities.

After the *partial entity recognition* phase, we obtain disordered number entity blocks. However, the entity blocks that compose the complete number entity are in order. In *entity recovery* phase, we design concatenation rules to combine the number entity blocks to produce number entities. To facilitate understanding, we use the example of phone number entity recognition throughout the following section.

6.1 Characteristics Analysis and Summarization

In this step, we summarize the characteristics of number entity blocks to extract features. The characteristics of number entity blocks in multiple rounds of dialogue are as follows. 1) When people say a number entity, the number entity is probably divided into number entity blocks with different lengths. For example, the phone number entity "13955678845" might be divided into "139 5567 8845", "1395 5678 845". 2) To confirm that the phone number is recorded correctly, the number block may be repeated by the CSA. 3) More than one number entity may exist in a dialogue. 4) In addition to the number entity blocks that compose the number entity, a dialogue text may also contain number entities that do not need to be split. An example is shown in Fig. 4.

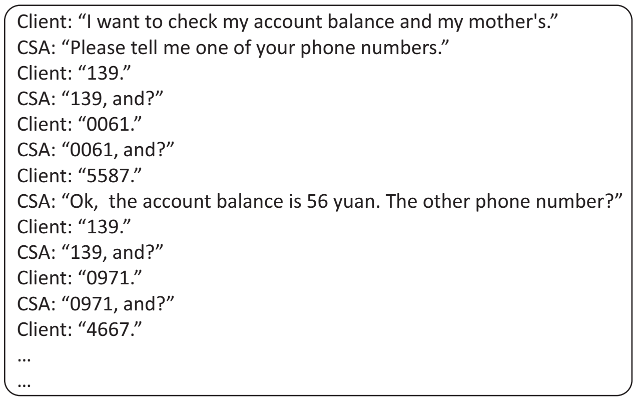
Client: "I want to check my account balance and my mother's."
CSA: "Please tell me one of your phone numbers."
Client: "139."
CSA: "139, and?"
Client: "0061."
CSA: "0061, and?"
Client: "5587."
CSA: "Ok, the account balance is 56 yuan. The other phone number?"
Client: "139."
CSA: "139, and?"
Client: "0971."
CSA: "0971, and?"
Client: "4667."
...
...

Figure 4: An example of phone number blocks in a dialogue

6.2 Labeling Dataset

We regard number entity recognition as a sequence labeling task, so we first break down sentences of texts in the training set into words and label each word. Because a number entity is usually divided into two or more number entity blocks, we use label "B", "I", "E" to represent the first block, the middle block and the last block. Label "O" represents non-phone number entity word. Due to the repetition of the number entity blocks, we use label "R" to denote a repeat. For example, label "BR" indicates that the first block of the number entity is repeated. Tab. 2 shows an example of labeling results for number entity blocks. The first number entity block of the number “13016992345” is “130”, we label the first “130” as “B”. Because “130” is repeated, we label the second “130” as “BR”. “1699” is the middle block, so we label “1699” as “I”.

Table 2: Labeling result of training data for CRF

“I want to check the phone bill”	
“Please tell me your phone number”	
“130”	130-B
“130, then?”	130-BR
“1699”	1699-I
“and?”	
“2345”	2345-E
“Ok, please wait a moment, your monthly consumption is 50 yuan”	

6.3 Feature Building

As for CRF model, we need to extract the features of number entity blocks to construct feature functions. By analyzing the characteristics of number entity blocks in dialogue texts, we leverage the following features to determine whether a word is a number entity block. 1) Part-of-speech to decide whether the word is a number. 2) How many digits does the word consist of? 3) Times of numbers that appear before the word. 4) Whether the word is the same as its previous number. 5) Whether there is a demonstrative before the word. Demonstratives are words that often appear before entities. In this paper, we have six demonstrative dictionary corresponding to “B”, “I”, “E”, “BR”, “IR”, “ER”. 6) Whether number entity blocks exist before the word. Based on the six situations, we build a 16-dimensional feature vector for each word. Situations 1, 2, 3, and 4 are represented in one dimension. Situation 5 needs six dimensions because we need to determine whether the previous word of the current word is demonstrative of the six entity categories. Situation 6 also needs six dimensions to present whether the six labels already appear before the word.

6.4 Template Building

In CRF model, we do not need to construct feature functions ourselves, and we only need to provide feature template to the model. The feature template is designed to define different features and the degree of dependence between words. For example, the model should consider the impact of the label of the previous word on the current word. We need to define feature templates “U00:%[x, y].” “U00” is the sequence number of the feature template. We can define any number of feature templates. “x” represents the relative position to the current word. “y” represents a feature. For example, “U00:%[−1, 0]” means that we consider the dependence of the feature at position “0” of the previous word on the label of the current word. CRF model automatically generates feature functions based on the feature template. We take the feature vector of each word as input to train CRF model.

6.5 Number Entity Block Concatenation

The number entity blocks that compose number entities are in order. However, the extracted number entity blocks are disordered and there are duplicate entity blocks. In this step, we design rules to splice the number entity blocks to generate complete number entities. The rules are as follows. 1) Splice the entity blocks marked with label “B”, “I”, “E” in sequence. 2) If the label

"B" is not recognized, but the label "BR is recognized", we use label "BR" instead of label "B". The same case is for the label "I" and "E". 3) If the label "B", "I" or "E" is not recognized, we still splice the identified entity blocks in sequence. 4) For label "B" and label "E", we select the entity block where the label appears for the first time. 5) For label "I", we splice the entity blocks where identification result is the label "I" in sequence. Based on these rules, if a dialogue contains number entity blocks and the number entity blocks are recognized through CRF model, we splice the number entity blocks.

An example is shown in Tab. 3. Through CRF model, we can get the following recognition results. In Result 1, we splice the entity blocks corresponding to label "B", "I" (a total of two entities) and "E" in sequence. In Result 2, CRF model did not recognize the label "B", but the label "BR" was recognized. We use the entity block corresponding to label "BR" as the first number entity block. In Result 3, CRF model did not recognize the label "E", and we splice the entity blocks corresponding to the labels "B" and "I" in sequence. There are two labels "E" in Result 4. We select the number entity block corresponding to label "E" that appears for the first time.

Table 3: Number entity block recognition results

Entity block	Result 1	Result 2	Result 3	Result 4
754	B	O	B	B
754	BR	BR	BR	BR
7300	I	I	I	I
7300	IR	IR	IR	IR
8973	I	I	I	I
459	E	E	I	E
459	ER	ER	IR	E
Concatenation result	75473008973459			

7 Experiment and Evaluation

This section reports the experiment for evaluating the performance of our method. We compare the prediction performance of the BOW + HM-SVM model and the Doc2vec + HM-SVM model, and we select the model with better performance to extract dialogue fragments. We tune parameters of HM-SVM model based on the results of extracting dialogue fragments. We evaluate the impact of each feature in CRF model on number entity block recognition. Finally, we compare our method with CRF model.

7.1 Experimental Setup

Data set. The data set used in experiment is voice conversation text between users and the CSA from China Telecom call center[2]. Due to user privacy, the data has been decrypted. There are 100 dialogue texts in the data set and 70 dialogues have the intention related to phone numbers. We manually label these texts and randomly select 70 dialogues as training data, and the other 30 dialogues as test data. Each group of experiments is tested 10 times and we report the average result.

[2] http://www.189.cn/zj/

We use the Chinese stopword list released by Harbin Institute of Technology as the basic stopword list. In addition, we add stop words summarized by telecommunication experts.

Evaluation Measure. We use F-Score as evaluation metric. F-Score is the weighted harmonic average of precision and recall. Different issues have different emphasis on precision and recall, F-Score comprehensively combines accuracy and recall. Eq. (2) shows the computation of F-Score, where P denotes precision and R denotes recall. If "$0 < \beta < 1$," the accuracy rate has a larger impact. If "β is 1," it is the F1 value. If "$\beta > 1$," the recall rate has a larger impact. Because we aim to extract the number entities to help the CSA confirm that the record of the number is correct, we pay more attention to the precision. In this experiment, we set β to 0.5.

$$F = \frac{((\beta)^2 + 1) \times P \times R}{(\beta)^2 \times P + R} \tag{1}$$

7.2 *Dialogue Fragment Extraction*

In this section, we compare the impact of BOW model and Doc2vec model on dialogue fragment extraction. We tune parameters of HM-SVM model according to the extraction performance on dialogue fragments.

We use BOW model and Doc2vec model to convert the dialogue texts into the input of HM-SVM model, and we use HM-SVM model to extract the dialogue fragments. The experimental results are shown in Tab. 4. The accuracy, recall and F-score of BOW model are higher than those of Doc2vec model. Although the Doc2vec model can leverage more contextual information of sentences, the experimental results show that BOW + HM-SVM model has a better performance. The reason is that Doc2Vec is a shallow neural network, and as a machine learning model, the HM-SVM model cannot extract more semantics.

Table 4: Performance comparison of BOW model and Doc2vec model on dialogue fragment extraction

"Model"	Precision (%)	Recall (%)	F-score (%)
"BOW"	87.45	76.79	85.09
"Doc2vec"	79.68	37.27	64.91

Based on the experiments, we use BOW + HM-SVM model for dialogue fragment extraction and tune parameters of HM-SVM model. The parameters to be fine-tuned are as follows. 1) Parameter C. Typical SVM parameter C trading-off slack compared to magnitude of the weight-vector. 2) Parameter $ORDER_t$. Order of dependencies of transitions in HMM. 3) Parameter $ORDER_e$. Order of dependencies of emissions in HMM. As we only pay attention to whether the model extracts the number dialogue fragments, we assume that the recognition result is correct if the extracted dialogue fragments contain the complete number information. Fig. 5 shows the performance of each parameter on recognition results of HM-SVM.

Parameter C. As shown in Fig. 5a, parameter C increases from 0 to 20. When C is 0, the recall is 1 because the label of each sentence in texts is identified as the dialogue fragment. The precision is the highest when C is 10. When C is 15, the recall is the highest. The F-score is the same when C increases from 10 to 20. Because we focus more on precision, we set C to 10 as the optimal solution.

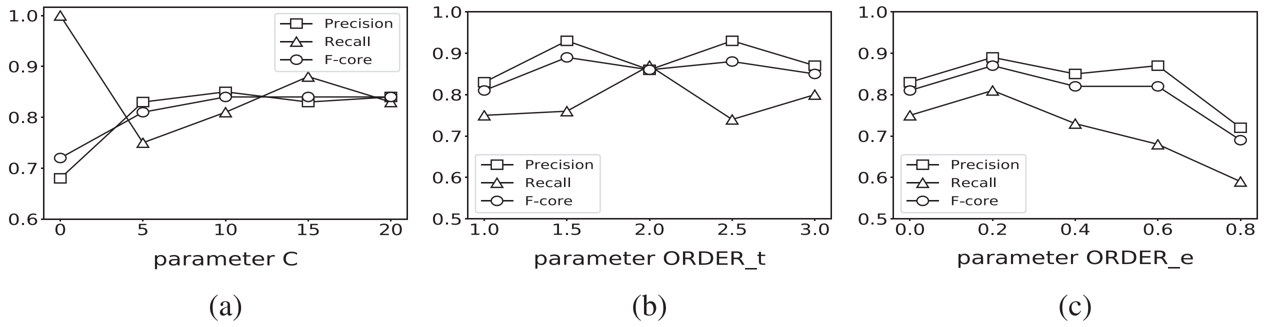

Figure 5: Precision, recall and F-score for HM-SVM on dialogue fragment extraction. (a) Parameter C, (b) parameter $ORDER_t$, (c) parameter $ORDER_e$

Parameter $ORDER_t$. Parameter $ORDER_t$ is the order of dependencies of transitions in HMM and is required to be less than 3. We set t in the range of 1 to 3. In Fig. 5b, when $ORDER_t$ is 2.0, the recall is the highest. The precision is the same where $ORDER_t$ is 1.5 and 2.5, but the F-score of 1.5 is the highest. We set $ORDER_t$ to 1.5 as the optimal solution.

Parameter $ORDER_e$. Fig. 5c shows the impact of parameter $ORDER_e$ that is set less than 1 on number dialogue fragment extraction. When parameter $ORDER_e$ is 0.8, the precision, recall and F-score are all the smallest. As the parameter $ORDER_e$ increases from 0 to 0.6, the precision and F-score change slightly. 0.2 is set as the optimal solution of parameter $ORDER_e$.

7.3 Number Entity Block Recognition

In this section, we evaluate the impact of each feature in CRF model on number entity blocks recognition. The training data and test data are dialogue fragments from the number dialogue fragment extraction. We evaluate the performance of one feature by only utilizing the other five features. Because a complete number is spliced by entity blocks, we evaluate the extraction results of the spliced complete numbers. Tab. 5 shows the impact of each feature.

Table 5: Extraction performance of each feature on number entity blocks extraction

“Feature”	Precision (%)	Recall (%)	F-score (%)
“All feature”	84.97	79.72	83.87
“Without feature 1”	80.95	78.21	80.39
“Without feature 2”	79.08	71.39	77.41
“Without feature 3”	80.89	75.51	79.75
“Without feature 4”	80.05	79.93	80.03
“Without feature 5”	82.07	75.05	80.57
“Without feature 6”	77.27	65.99	74.71

Feature 1-part of speech. Feature 1 is the part-of-speech. When we extract numbers from the dialogue fragments without considering Feature 1, the recall declining is not obvious, but the accuracy and the F-score decrease significantly. So Feature 1 has impact on the performance and should be utilized.

Feature 2-number of digits. Feature 2 is the number of digits that form a word. The recall, accuracy and F-score decrease significantly when we extract number entities from the dialogue fragments without considering Feature 2. Such a result means that the number of digits has a great impact on extraction, so Feature 2 should be utilized in CRF model.

Feature 3-times of numbers. Feature 3 is the times of numbers that appear before the word. When we do not utilize Feature 3, the recall, accuracy, and F-score all decreases. So Feature 3 is useful and should be utilized in CRF model.

Feature 4-whether the words are the same. Feature 4 refers to whether a word is the same as its previous number. The experiment result shows that although the recall improves, the precision and F-score decrease. The result means that Feature 4 has an impact on extraction, so we cannot remove the Feature 4 in CRF model.

Feature 5-demonstrative. Feature 5 is the demonstrative before the word. The experimental result shows that without Feature 5, the precision, recall and F-score all decrease. Such a result means that the demonstrative has an impact on the extraction performance, so we utilize Feature 5 in CRF model.

Feature 6-whether number entity blocks exist. Feature 6 is whether number entity blocks exist before the word. Without Feature 6, the precision, recall, and F-score all decrease significantly. So we utilize Feature 6 in CRF model.

Based on the experiments, we can draw the following conclusion. With the six features, the number entity blocks extraction performance of CRF model is the optimal.

7.4 Number Entity Extraction

We evaluate the performance of the number entities extraction in multiple rounds of dialogues. CRF model has superior performance for named entity recognition tasks. Our method locates the fragments in the dialogue before using CRF model to extract the number entity blocks. In this section, we compare the proposed method with CRF model without dialogue fragments extraction. The neural network models can also complete named entity recognition, but neural network model needs a large amount of training data, so our experiment does not involve the deep learning methods. The result is shown in Tab. 6. The precision of our method is 91.97%, which is higher than CRF model. The result shows that the recall is higher when we only use CRF model. The reason is that we first use HM-SVM model to extract the dialogue fragments that contain the number intent from the texts. The recall of HM-SVM model is around 84.43%, which means 25.57% of dialogue fragments will not be identified. It leads to an impact on the overall recognition performance of our method. After the CSA records the number, we help him/her to check whether the record is correct by extracting the numbers. We pay more attention to the impact of our method on precision. According to F-score, our method is superior to CRF model.

Table 6: Comparison experimental results

"Method"	Precision (%)	Recall (%)	F-score (%)
"CRF"	84.06	79.68	83.15
"HM-SVM + CRF"	91.97	69.81	86.48
"HM-SVM in HM-SVM + CRF"	93.10	84.43	91.22
"CRF in HM-SVM + CRF"	91.97	82.64	89.94

8 Conclusion

In this paper, we propose a method based on HM-SVM model and CRF model, which is capable of extracting number entities in multiple rounds of dialogue systems. The extraction is divided into three modules, i.e., fragment locating, partial entity recognition and entity recovery. We use HM-SVM model extracting dialogue fragments that contain number entity blocks. We use CRF model to extract number entity blocks at discontinuous positions from dialogue fragments. We concatenate number entity blocks into complete number entities according to concatenation rules. We conduct extensive experiments on real-world data. In the experiments, we compare the prediction performance of "BOW + HM-SVM" model and "Doc2vec + HM-SVM" model on dialogue fragment extraction. We tune three parameters of HM-SVM model based on precision, recall and F-score. Besides, we evaluate the impact of six features for CRF model on number entity blocks recognition. We compare our method with CRF model, and the experimental results verify that our method can successfully extract number entities in multiple rounds of dialogue systems.

Funding Statement: This research was partially supported by: Zhejiang Laboratory (2020AA3AB05), and the Fundamental Research Funds for the Provincial Universities of Zhejiang (RF-A2020007).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Yang, X., Zhou, S., Cao, M. (2020). An approach to alleviate the sparsity problem of hybrid collaborative filtering based recommendations: The product-attribute perspective from user reviews. *Mobile Networks and Applications, 25(2),* 376–390. DOI 10.1007/s11036-019-01246-2.
2. Yin, Y., Huang, Q., Gao, H., Xu, Y. (2020). Personalized APIS recommendation with cognitive knowledge mining for industrial systems. *IEEE Transactions on Industrial Informatics,* 1. DOI 10.1109/TII.2020.3039500.
3. Yin, Y., Cao, Z., Xu, Y., Gao, H., Li, R. et al. (2020). QoS prediction for service recommendation with features learning in mobile edge computing environment. *IEEE Transactions on Cognitive Communications and Networking, 6(4),* 1136–1145. DOI 10.1109/TCCN.2020.3027681.
4. Gao, H., Kuang, L., Yin, Y., Guo, B., Dou, K. (2020). Mining consuming behaviors with temporal evolution for personalized recommendation in mobile marketing apps. *Mobile Networks and Applications, 25(1),* 1–16. DOI 10.1007/s11036-019-01249-z.
5. Gao, H., Liu, C., Li, Y., Yang, X. (2020). V2vr: Reliable hybrid-network-oriented V2v data transmission and routing considering rsus and connectivity probability. *IEEE Transactions on Intelligent Transportation Systems,* 1–14. DOI 10.1109/TITS.2020.2983835.
6. Altun, Y., Tsochantaridis, I., Hofmann, T. (2003). Hidden markov support vector machines. *Proceedings of the 20th International Conference on Machine Learning*, Washington DC.
7. Lafferty, J., McCallum, A., Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proceedings of 18th International Conference on Machine Learning*, Williamstown, MA, USA.
8. Le, Q., Mikolov, T. (2014). Distributed redatespresentations of sentences and documents. *International Conference on Machine Learning*, Beijing, China.
9. Irmak, U., Kraft, R. (2010). A scalable machine-learning approach for semi-structured named entity recognition. *Proceedings of the 19th International Conference on World Wide Web*, Raleigh, North Carolina, USA.

10. Appelt, D. E., Hobbs, J. R., Bear, J., Israel, D., Kameyama, M. et al. (1995). SRI International FASTUS systemMUC-6 test results and analysis//Sixth Message Understanding Conference (MUC-6). *Proceedings of a Conference Held in Columbia*, Maryland.
11. von Brzeski, V., Irmak, U., Kraft, R. (2007). Leveraging context in user-centric entity detection systems. *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management,* New York, NY, USA. Association for Computing Machinery.
12. Appelt, D. E., Hobbs, J. R., Bear, J., Israel, D., Tyson, M. (1993). Fastus: A finite-state processor for information extraction from real-world text. *IJCAI*, *93*. Chambéry, France.
13. Stern, M. K. (2004). Dates and times in email messages. *Proceedings of the 9th International Conference on Intelligent User Interfaces,* New York, NY, USA. Association for Computing Machinery.
14. Saha, S. K., Mitra, P., Sarkar, S. (2008). Word clustering and word selection based feature reduction for MaxEnt based Hindi NER. *Proceedings of ACL-08: HLT,* Columbus, Ohio. Association for Computational Linguistics.
15. Özkaya, S., Diri, B. (2011). Named entity recognition by conditional random fields from turkish informal texts. *2011 IEEE 19th Signal Processing and Communications Applications Conference (SIU)*, Ankara, Turkey.
16. Fissaha Adafre, S., de Rijke, M. (2005). Feature engineering and post-processing for temporal expression recognition using conditional random fields. *Proceedings of the ACL Workshop on Feature Engineering for Machine Learning in Natural Language Processing,* Ann Arbor, Michigan. Association for Computational Linguistics.
17. Georgiev, G., Nakov, P., Ganchev, K., Osenova, P., Simov, K. (2009). Feature-rich named entity recognition for Bulgarian using conditional random fields. *Proceedings of the International Conference RANLP-2009,* Borovets, Bulgaria. Association for Computational Linguistics.
18. Ekbal, A., Haque, R., Bandyopadhyay, S. (2008). Named entity recognition in Bengali: A conditional random field approach. *Proceedings of the Third International Joint Conference*, vol. *2*. Hyderabad, India.
19. Li, P. H., Dong, R. P., Wang, Y. S., Chou, J. C., Ma, W. Y. (2017). Leveraging linguistic structures for named entity recognition with bidirectional recursive neural networks. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark.
20. Yadav, V., Bethard, S. (2018). A survey on recent advances in named entity recognition from deep learning models. *Proceedings of the 27th International Conference on Computational Linguistics*, Santa Fe, New Mexico, USA.
21. Chiu, J. P., Nichols, E. (2016). Named entity recognition with bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics, 4(1),* 357–370. DOI 10.1162/tacl_a_00104.
22. Luoma, J., Pyysalo, S. (2020). Exploring cross-sentence contexts for named entity recognition with bert. *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 904–914, Barcelona, Spain (Online). DOI 10.18653/v1/2020.coling-main.78.
23. Jie, Z., Lu, W. (2019). Dependency-guided lstm-crf for named entity recognition. *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, Hong Kong, China.
24. Ren, J., Liang, J., Zhao, C. (2020). Chinese named entity recognition with changed bilstm and CRF. *Artificial Intelligence in China, 572,* 398–406. DOI 10.1007/978-981-15-0187-6_47.
25. De Souza, F. B., Nogueira, R., de Alencar Lotufo, R. (2019). Portuguese named entity recognition using bert-crf. *arXiv preprint*, arXiv: 1909.10649.
26. Bokaei, M. H., Mahmoudi, M. (2018). Improved deep persian named entity recognition. *2018 9th International Symposium on Telecommunications*, Iran, Islamic Republic of Tehran.
27. Liu, M., Tu, Z., Wang, Z., Xu, X. (2020). LTP: A new active learning strategy for bert-crf based named entity recognition. *arXiv preprint*, arXiv: 2001.02524.
28. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, Lake Tahoe, Nevada, USA.