Neural Computing and Applications (2024) 36:1815–1832
https://doi.org/10.1007/s00521-023-09152-1

ORIGINAL ARTICLE

SSR-TA: Sequence-to-Sequence-based expert recurrent recommendation for ticket automation

Chenhan Cao[1] · Xiaoyu Fang[1] · Bingqing Luo[1] · Bin Xia[1]

Received: 31 January 2023 / Accepted: 16 October 2023 / Published online: 17 November 2023

Abstract
Ticket automation plays a crucial role in ensuring the normal operation of IT software systems. One of the key tasks of ticket automation is to assign experts to resolve incoming tickets. However, when facing a large number of tickets, inappropriate expert assignments can result in frequent ticket transfers between experts, leading to time delays and wasted resources. Therefore, it is essential to find an appropriate expert efficiently and effectively with minimal steps. To address this challenge, we propose a sequence-to-sequence-based translation model that is combined with a recurrent recommendation network to recommend suitable experts for tickets. The sequence-to-sequence model transforms the ticket description into the corresponding resolution for capturing the potential and useful features of representing tickets. The recurrent recommendation network recommends the appropriate expert based on the assumption that the previous expert in the recommendation sequence cannot solve the ticket. To evaluate the performance of our proposed model, we conducted experiments on two real-world datasets and compared SSR-TA with several baselines. The experimental results demonstrate that our proposed model outperforms the baselines.

Keywords Ticket automation · Seq2seq · Recurrent recommendation · Description translation

1 Introduction

In an IT software system, maintaining the normal operation is the essential requirement to provide high-quality services. In practice, the relationship among components of the IT software system is complicated. Once an exception occurs in a component, the exception may spread rapidly to other components and generate many concurrent warnings, causing the system to be unable to provide service. For example, IBM cloud suffered two outages within five days in 2021 [1]. The services were hit in the UK, the USA, Sydney, Tokyo, and more for several hours due to incorrect routing settings by the external network provider. However, manually assigning an expert to address the exception is time-consuming and expensive. In order to catch exceptions and solve problems in time, the ticket automation is widely used in IT software systems. Figure 1 illustrates the workflow of the ticket automation. In detail, when an exception is detected by the engineer/monitor system, a ticket containing the exception description and the corresponding system information will be submitted to the ticket system. Further, the ticket system assigns an expert to solve the ticket according to the submitted information (i.e., the description and the system information). If the current expert cannot solve the ticket, another expert will be assigned by the ticket system until the ticket is solved. However, the inappropriate assignment (i.e., the assigned expert cannot solve the ticket) will cause serious problems such as time delays, wasted resources, and service failures. It is crucial to assign an appropriate expert to solve the ticket in time.

✉ Bin Xia
bxia@njupt.edu.cn

Chenhan Cao
1220044906@njupt.edu.cn

Xiaoyu Fang
1220044932@njupt.edu.cn

Bingqing Luo
luobq@njupt.edu.cn

[1] Jiangsu Key Laboratory of Big Data Security and Intelligent Processing, Nanjing University of Posts and Telecommunications, Nanjing, China

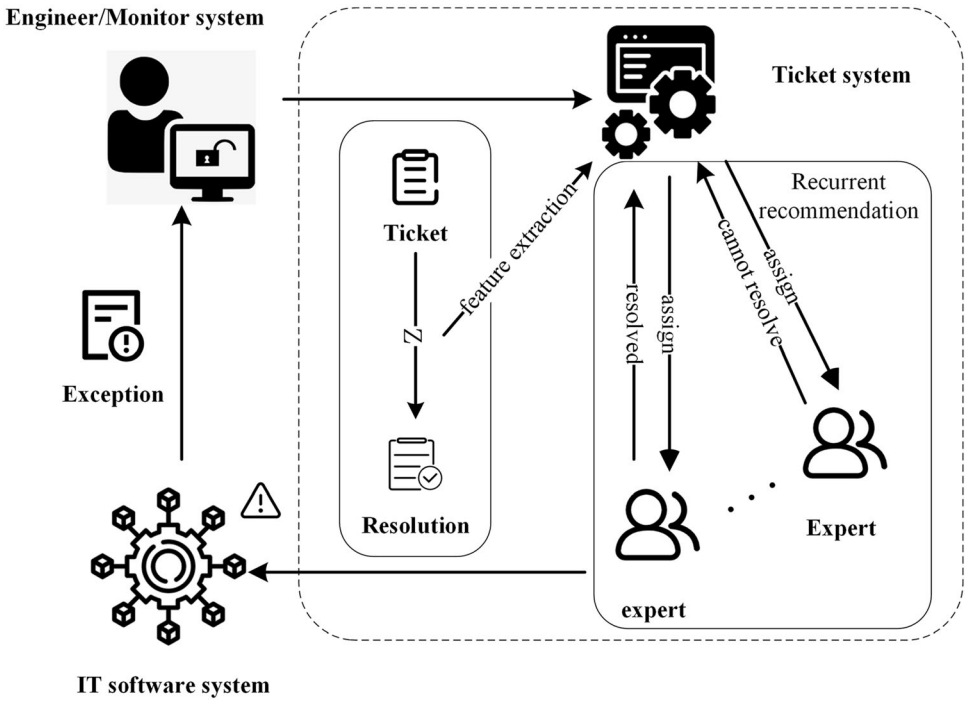


Fig. 1 The workflow of the ticket automation

Many efforts have been made by researchers to improve the performance of ticket automation at different stages. For the ticket generation, the previous works focused on generating tickets automatically [2] and reducing ticket submitting delays [3]. To represent ticket description, the approaches based on the service catalog mapping [4] and the entity name extraction [5] were introduced. Additionally, several methods were proposed for the ticket classification, including the entity recognition for ticket description [6], the similarity-based ticket clustering [7], and the hierarchical multi-label classification [8]. Furthermore, researchers proposed two types of ticket recommendation methods, including resolution and expert recommendation. The resolution recommendation is the methods based on the ticket description similarity [9] or the hierarchical multi-armed bandit [10] recommend a corresponding resolution for the ticket. The expert recommendation is proposed to find an appropriate expert to solve the ticket based on the ticket description and the ticket transfers among experts [11]. Topic models such as LDA [12] and deep semantic models like DSSM [13] are used for feature representation of ticket descriptions. However, the recommendation method based on similarity between ticket and expert features relies on matching the similarity between the current ticket and expert features, which may result in recommending experts with similar abilities who are unable to solve the ticket. Moreover, the method based on historical ticket transfer relationships recommends experts based on the transfer path of historical tickets, which may result in invalid recommendations if the previous experts in the transfer path were unable to solve the ticket, leading to longer resolution time.

To overcome the aforementioned problems, we propose a sequence-to-sequence (seq2seq)-based model SSR-TA which is combined with a recurrent recommendation network to recommend appropriate experts for the ticket automation. For training SSR-TA, the description of solved tickets is translated into the corresponding resolution, where the potential features of ticket description and resolution are captured to improve the representation of the ticket description. The expert recommendation will be generated based on the effective representation of the ticket description. Because the traditional Top-N recommendation generates expert sequences based on the similarity, which may cause that if the current expert cannot solve the ticket, other similar experts in the sequence cannot solve the ticket either. By introducing an attention mechanism, SSR-TA recurrently recommends the next appropriate expert based on the assumption that the previous expert in the recommended sequence failed to solve the ticket. The main contributions of this paper are summarized as follows:

1. To the best of our knowledge, SSR-TA is the first seq2seq-based model that combined the features from description and resolution to assist the expert recommendation, and the model can be used for ticket resolution generation.
2. We propose a recurrent recommendation network adopting an attention mechanism to generate the expert recommendation sequences, while the generation is

based on the assumption that the previous expert in the sequence failed to solve tickets, the results show that the recurrent recommendation network improves the ranking of the true resolver in the recommendation sequence.
3. We conduct experiments to discuss the effectiveness of description translation and the recurrent recommendation network, and compare the performance of several baselines with SSR-TA.

The rest of the paper is organized as follows: Section 2 summarizes research related to ticket automation. Section 3 describes the details of SSR-TA. Section 4 conducts experimental comparisons and results analysis. Finally, Sect. 5 concludes the paper.

2 Related work

This section introduces the related research on the ticket automation and the application of seq2seq technology. In addition, the similarities and differences between these approaches and ours are presented briefly.

2.1 Ticket automation

2.1.1 Ticket data analysis

For automatic management of cellular networks, Palacios et al. [2] proposed a big data-empowered framework using the anomaly detection and root cause analysis tools to identify the network state and automatically generate trouble tickets. To reduce the time of interaction during resolving a ticket, Gupta et al. [3] analyzed the historical ticket interactions between analysts and users, and proposed a system to requesting additional information when the user was raising a ticket based on the ticket description. Moreover, Shimpi et al. [4] proposed an integrated framework to extracting issues from tickets under different scenarios, including system-generated and user-generated tickets. Han et al. [5] studied the problem of extracting software product names that are represented by various surface forms and informal abbreviations. They designed features to precisely map abbreviations to the product and version by analyzing ticket language patterns. These approaches improved the efficiency of ticket analysis by extracting effective features of ticket descriptions. To fully exploit the information in the ticket data, our approach combines the description with the corresponding resolution to capture the effective and potential representation of tickets for ticket analysis.

2.1.2 Ticket classification

Potharaju et al. [6] proposed NetSieve which was a framework to infer problem symptoms, troubleshooting activities, and resolution actions based on analyzing the steps performed on each network entity from tickets. Zeng et al. [8] explored the hierarchical multi-label ticket classification problem and proposed GLabel, which was a greedy algorithm to search the optimal hierarchical multi-label for each ticket, where a novel contextual hierarchy loss was applied to maintain the accordance of labels assigned to each ticket. Zhou et al. [14] proposed an unsupervised ticket classification, which used the classic canonical analysis to extract ticket features for ticket labeling. Xu et al. [7] constructed signatures to represent each type of ticket and proposed an algorithm to identifying the type of incoming tickets by comparing the similarity between the representation of incoming tickets and each type signature. To improve the effectiveness of measuring similarities among tickets, they also proposed a multi-view similarity measure framework that integrated the traditional similarity measures (e.g., Jaccard similarity, NLCS) [15]. Piero et al. [16] proposed two machine learning and natural language processing techniques COTA v1 and COTA v2 to converts the multi-classification task into a ranking problem, and use encoder-combiner-decoder get heterogeneous input and output feature types. The method was validated the real-world impact of COTA in reducing issue resolution time by 10 percent without reducing customer satisfaction. Fuchs et al. [17] conducted a literature review in the field of automated support ticket systems, they found creating an automated incident management tool being the majority topic in the field and identified Random Forrest and Support Vector Machine as best performing algorithms for classification in the field. Recently, Alessandro et al. [18] conducted in-depth research on multi-level ticket classification and used a pre-trained BERT language model to classify tickets by topic, which improved the accuracy of ticket classification. These methods exploited semantic or syntactic analysis of ticket descriptions to improve the performance of ticket classification. However, in this paper, we considers the ticket automation as a Q&A problem by translating the description into the corresponding resolution. Our proposed approach fully utilizes the sequential information to learn the representation of tickets which is capable of improving the performance of ticket classification.

2.1.3 Ticket recommendation

Ticket recommendation is mainly categorized as resolution recommendation and expert recommendation. Resolution recommendation is to provide appropriate historical ticket

resolutions for the incoming ticket, while expert recommendation is to assign potential experts to address the incoming ticket. For resolution recommendation, Zhou et al. [19] used the k-nearest neighbors algorithm to find out the historical tickets which are similar to the incoming ticket, and recommend the corresponding resolutions. However, the same exception would sometimes generate tickets with different descriptions due to the changes of system environment, which may weaken the effectiveness of the resolution recommendation. To solve this problem, they utilized structural corresponding learning-based feature adaptation to maintain the coincidence of representations to these difference tickets [20]. Zhou et al. [9] also used the siamese neural network to compare the similarity between historical tickets and the incoming one, and recommended the resolution which belonged to the most similar historical ticket. Wang et al. [10] cast resolution recommendation as a reinforcement learning problem and proposed an integrated framework based on hierarchical multi-armed bandits with the arm dependencies considered as a tree-structured hierarchy, while the framework recommends the resolution with the maximum reward at leaf node. For expert recommendation, Xu et al. [21] proposed a two-stage expert recommendation model that if the initial expert cannot solve the ticket, the unsolved ticket will be transferred to the next expert. Furthermore, Han et al. [22] proposed a unified ticket routing framework that incorporated four types of features (i.e., ticket, group, ticket-group, and group-group) for ranking the list of the expert recommendation. Recently, they also proposed a multi-view model to further improve the performance of the expert recommendation in their previous work, where the graph convolutional network was used to generate expert graph-view representation and the deep neural network was used to obtain the text-view feature [11]. However, the traditional methods often ranked the expert and resolution recommendation list based on the similarity (e.g., the similarity between tickets, or the similarity between experts and tickets). In other words, if the first recommendation in the list fails, the second one will probably fail due to the incorrect representations of tickets or experts. In addition, these expert recommendation approaches relied on the ticket transfers among experts which emphasized the ability of experts while ignoring the root cause of tickets. The ticket recommendation is similar to the Q&A task, which retrieves or generates the appropriate answer to the given question. Recently, Raheja et al. [23] proposed a context-aware self-attention mechanism coupled with a hierarchical recurrent neural network to classify Dialogue Acts, effectively capturing utterance-level semantic text representations while maintaining high accuracy. In addition, Deng et al. [24] proposed a unified model to exploit the opinion information from the reviews to facilitate the opinion-aware answer generation for a given product-related question, effectively generating opinionated and informative answers. These approaches performed well for the Q&A task mainly comprising contextual natural language texts, however, they are ineffective for understanding ticket descriptions that are context-free and without polarity. To overcome these problems, SSR-TA is trained only based on the historical ticket descriptions and resolutions and is designed to recurrently generate the expert recommendation based on the assumption that the previous expert cannot solve the ticket.

Table 1 shows the summary of the related ticket automation works and the proposed method in this paper:

2.2 Sequence to Sequence

Inspired by the recurrent neural network-based encoder-decoder model, Sutskever et al. [25] proposed the seq2seq model, which is now widely used for many scenarios such as machine translation, image caption generation, and speech-to-text conversion. To improve the performance of seq2seq model, Bahdanau et al. [26] modified the original seq2seq model using the attention mechanism to reduce the information loss caused by compressing the source sentence into a hidden vector. Furthermore, Kenesh et al. [27] constructed the seq2seq model based on the reinforcement learning and tried to address the exposure bias and inconsistency between train/test measurement in the original seq2seq model. Pent et al. [28] proposed a seq2seq-based approach for mapping natural language sentences to abstract meaning representation semantic graphs. Also, seq2seq was used in aspect term extraction, where the source sequence and target sequence is composed of words and labels, respectively [29]. Huang et al. [30] proposed an end-to-end approach which can recognize multiple languages in images considering data imbalance between languages. Lewis et al. [31] proposed a denoising autoencoder named BART, which combines bidirectional and auto-regressive transformers for pretraining sequence-to-sequence models, and BART is effective for text generation after fine-tuning. In addition, Mao et al. [32] proposed GAR (generation-augmented retrieval) for answering open-domain questions, which enriches the semantics of the queries by generating contexts for answer retrieval. These methods are effective for text generation and augmentation based on contextual information; however, ticket descriptions are context-free, which is difficult for relevant information extraction. The principle of seq2seq model is to transform the sequential source data into the target data (e.g., English to Chinese, image to text, and question to answer). SSR-TA is to transform the ticket description into the corresponding resolution for capturing the potential and

Table 1 The summary of the related ticket automation works and the proposed method in this paper

Article	Task	Methods
Zeng et al. [8]	Ticket classification	Greedy algorithm
Xu et al. [7]	Ticket classification	Signature matching
Alessandro et al. [18]	Ticket classification	BERT, multi-level
Piero et al. [16]	Ticket classification	RF, encoder-combiner-decoder
Zhou et al. [19]	Resolution recommendation	KNN
Zhou et al. [9]	Resolution recommendation	Siamese neural network
Wang et al. [10]	Resolution recommendation	Hierarchical multi-armed bandits
Xu et al. [21]	Expert recommendation	Two-stage expert recommendation
Han et al. [11]	Expert recommendation	GCN, DNN
This paper	Expert recommendation	Encoder–decoder, attention

useful features which is capable of representing tickets and improve the performance of the expert recommendation.

3 Method

SSR-TA mainly consists of three components: a description encoder, a resolution decoder, and an expert recommendation network. Figure 2 shows the overall structure of SSR-TA. The description encoder is used to transform the description of ticket into the potential representation of knowledge to the ticket (i.e., the hidden state). The resolution decoder, which helps the description encoder improve the effectiveness of representing the ticket, is to generate the corresponding resolution of the ticket based on the hidden state. The expert recommendation network will recurrently generate appropriate expert to solve the ticket based on the the hidden state which is fine-tuned using an attention mechanism.

3.1 Data preprocessing

Table 2 illustrates the primary information of a ticket, mainly including structured and unstructured fields. The structured field contains ID, Type, Status, Datetime, and Expert, where these fields are generated automatically. In detail, ID is the index of the ticket, Type is the problem category, Status indicates whether the ticket is solved, Datetime is the generation time, and Expert is the expert who solved the ticket. The unstructured field contains the exception description generated by user/monitor system (i.e., Description) and the corresponding resolution proposed by the expert (i.e., Resolution). The notations mentioned in this paper are summarized in Table 3.

Suppose each ticket can be represented by a triple $(\boldsymbol{D}, \boldsymbol{R}, \boldsymbol{E})$, where $\boldsymbol{D}$ is the description of the ticket, $\boldsymbol{R}$ is the corresponding resolution, and $\boldsymbol{E}$ is the expert who solved the ticket. In detail, $\boldsymbol{D}$ can be represented by a sequence $[d_1, d_2, ..., d_i, ..., d_m, <\text{EOS}>]$, where d_i means the i_{th} word in the description and '$<\text{EOS}>$' is a special end-of-

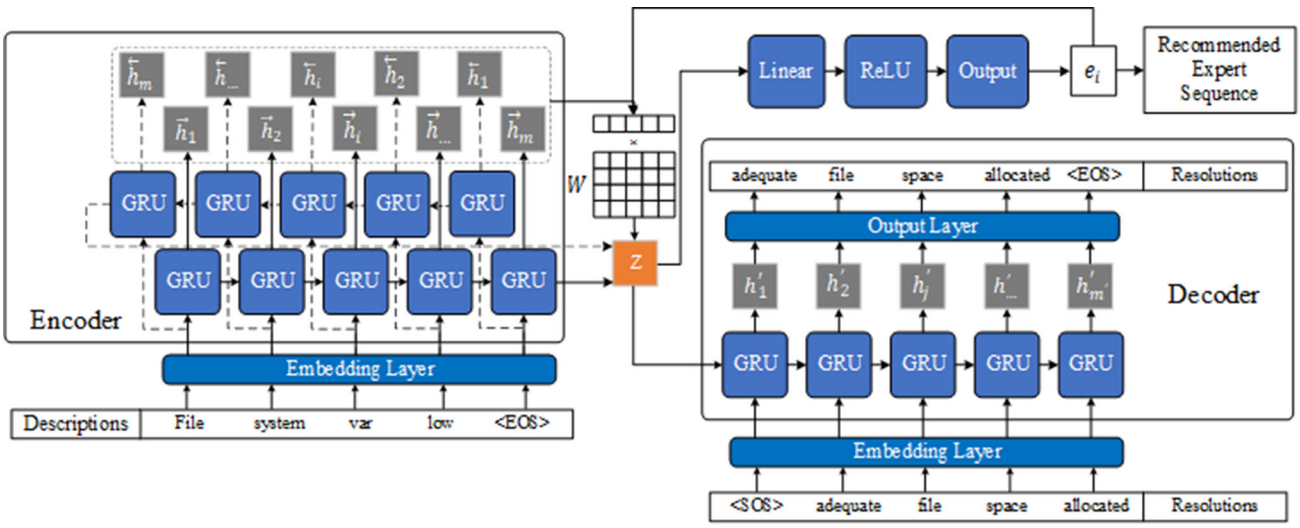


Fig. 2 The overview of SSR-TA

Table 2 A ticket example

Fields	Values
ID	19027525
Datetime	2014-04-29 06:00:12
Type	Connectivity
Status	Closed
Expert	Lynn.Ridge
Description	Failed to reconnect to PatrolAgent on host AVPMD623, port 3181 Will retry in 3 timer ticks
Resolution	Job failed due to GFT connectivity issue. We have checked and restarted the job. It completed successfully

Table 3 Notations mentioned in this paper

Notation	Description
D	The description of ticket
R	The resolution of ticket
E	The expert label sequence of ticket
m	The length of the description sequence
m'	The length of the resolution sequence
$<\text{EOS}>$	The special end-of-sentence symbol
k	The kth expert in the sequence
K	The number of experts
$\overline{E}$	The recommended expert sequence of ticket
$\overline{E}_{n-1}$	The $(n-1)_{\text{th}}$ recommendation result
h_i	The ith hidden state output by encoder
z	The hidden state output by encoder last layer
H	All hidden states output by encoder
$\tilde{W}$	The attention parameter of the recommendation network
W_n	The attention weight for the n_{th} recommendation
N	The number of experts recommended by model for a ticket
p_k	The probability of the kth expert being recommended by the model
set()	The same number of experts in the recommended sequence

sentence symbol. Similarly, $\boldsymbol{R}$ is denoted as $[r_1, r_2, ..., r_j, ..., r_{m'}, <\text{EOS}>]$, where r_j means the j_{th} word in the resolution. m is the length of the description sequence and m' is the length of the resolution sequence. Each word is represented by a vector, which can be learned through the embedding layer based on the word embedding mechanism. Also, $\boldsymbol{E} = [e_1, e_2, ..., e_k, ..., e_K]$ is the expert label of the ticket, where $e_k = 1$ indicates that the k_{th} expert can solve the ticket and $e_k = 0$ means the expert cannnot solve the ticket.

3.2 Model architecture

3.2.1 Description encoder

We use a bidirectional gate recurrent unit (GRU) network to construct the description encoder, and consider the word embedding of the ticket description $\boldsymbol{D}$ as the input. The output of the encoder h_m is considered as the hidden state vector $\boldsymbol{z}$, which contains the contextual information of the description. At each step i, the corresponding hidden states $\overleftarrow{h}_i$ and $\overrightarrow{h}_i$ are generated based on the corresponding input d_i and the previous hidden states $\overleftarrow{h}_{i-1}$ and $\overrightarrow{h}_{i-1}$ from two directions, respectively:

$$\overleftarrow{h}_i = f(d_i, \overleftarrow{h}_{i-1}), \tag{1}$$

$$\overrightarrow{h}_i = f(d_i, \overrightarrow{h}_{i-1}). \tag{2}$$

The output of the encoder h_m is generated as below:

$$h_m = [\overleftarrow{h}_m, \overrightarrow{h}_m]. \tag{3}$$

3.2.2 Resolution decoder

The resolution decoder of SSR-TA is an original GRU that is trained to generate the resolution sequence **R** based on the hidden state vector z. In detail, the decoder generates the current hidden state $h_j^{'}$ and the prediction of current word r_j based on the previous hidden state $h_{j-1}^{'}$ and the previous word r_{j-1} at each step j, where the hidden state vector z is considered as the initial hidden state $h_1^{'}$. The conditional probability of resolution sequence is defined as:

$$P(r_1, r_2, ..., r_{m'}|d_1, d_2, ..., d_m) = \prod_{j=1}^{n} P(r_j|z, r_1, r_2, ..., r_{j-1}), \tag{4}$$

where the left-hand side of Eq. (4) represents the probability of the model generating the corresponding resolution for each word based on the ticket description. The right-hand side represents the product of probabilities of generating each word of the resolution based on the intermediate variables generated by the encoder.

3.2.3 Expert recommendation

The expert recommendation network will generate a sequence of experts recurrently based on the hidden state vector z from the description encoder. SSR-TA is designed to generate the next expert based on the assumption that the previous expert cannot solve the ticket. We adopt an attention mechanism to implement this recurrent recommendation process, defined as Eqs. (5) and (6):

$$\boldsymbol{W}_n = \tanh\left(\tilde{W} \times \left[\overline{E}_{n-1}, H\right]\right), \tag{5}$$

$$z_n = \boldsymbol{W}_n \times H, \tag{6}$$

where $\boldsymbol{W}_n$ is the attention weight for the n_{th} recommendation, $\overline{E}_{n-1}$ is the $(n-1)_{\text{th}}$ recommendation result, H is all the hidden states of the description encoder, z_n is the nth hidden state vector, and the $\tilde{W}$ is a trained parameter. For example, the first hidden state vector z_1 is used to generate the first recommended expert e_1. The attention weight $\boldsymbol{W}_2$ is generated based on the current recommendation result $\overline{E}_1$ and all hidden states from the description encoder H. Then, the attention weight will be combined with all the hidden state states H to obtain the hidden state vector z_2, which is used to recommend another appropriate expert e_2. The expert recommendation network generates the next recommendation result based on the assumption that the currently recommended expert is unable to solve the ticket and finally obtains a specified number of expert recommendation sequences. Then, the new ticket will be assigned to the first expert in the recommended sequence. If the expert cannot solve the ticket, the ticket will be passed on to the subsequent experts in the sequence until the ticket is resolved or the time limit is exceeded.

3.3 Objective function

SSR-TA has three objectives, including the description translation, the accuracy of recommendation, and the disparity of recommendation. Therefore, three objective functions are designed to optimize our proposed model. The objective function theoretically aims to improve the accuracy of expert recommendations and shorten the steps to resolve tickets. The objective function for the description translation is to make the output of the resolution decoder as close to the real resolution as possible, which is defined as:

$$\text{Objective}_{dt} = -\log P(\boldsymbol{R} \mid \boldsymbol{D}), \tag{7}$$

where the $\boldsymbol{D}$ and $\boldsymbol{R}$ are the ticket description and the corresponding resolution, respectively.

For the recurrent recommendation network, the accuracy of recommendation means the probability of the expert recommended by SSR-TA each time is the true expert who solved the ticket, and the disparity of recommendation denotes the experts should be unique in the recommendation sequence. The objective function for the accuracy of recommendation is defined as Eq. (8):

$$\text{Objective}_{ar} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K} e_k \log\left(p_k\right), \tag{8}$$

where e_k is the kth expert and p_k denotes the probability of recommending the expert, K is the number of experts, and N indicates the length of the recommended expert sequence. The objective function for the disparity of recommendation is defined as Eq. (9):

$$\text{Objective}_{dr} = N - set(\overline{E}), \tag{9}$$

where $\overline{E}$ is the recommended expert sequence and $set()$ represents the number of unique experts in the sequence.

Finally, three hyperparameters α_1, α_2 and α_3 are introduced to balance SSR-TA among the description translation, the accuracy of recommendation, and the disparity of recommendation. A joint objective function is defined as below:

$$\text{Objective} = -\alpha_1 \log P(R, D) - \alpha_2 \frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K} e_k \log\left(p_k\right) + \alpha_3 (N - set(\overline{E})). \tag{10}$$

The values of α_1, α_2, and α_3 are set to 0.2, 0.3, and 0.5, respectively.

3.4 Train and prediction

The training process of SSR-TA is described in Algorithm 1. The inputs of SSR-TA are the ticket description $\boldsymbol{D}$, the corresponding ticket resolution $\boldsymbol{R}$, the length of the recommended expert sequence N, and the expert label of the corresponding ticket $\boldsymbol{E}$. The output is the recommended expert sequence S. In detail, the model will generate an empty recommendation sequence S, and the first hidden state vector z_1 is obtained from the description encoder. Then, based on z_1, the resolution decoder predicts the corresponding resolution of the ticket, while the first recommendation result $\overline{E_1}$ is generated by the model, and the most probable expert e_1 in $\overline{E_1}$ is added to the sequence S. For the loop (i.e., Step 6 to 10), the model will generate the next hidden state vector z_n based on the previous recommendation result $\overline{E}_{n-1}$, and obtain the next recommendation expert e_n. This loop will be repeated until the recommendation sequence S is filled by N recommended experts. During the training process, the teach-forcing [33] is used to guarantee the efficient convergence while ensuring the robustness of the model. The parameters of the SSR-TA model are initialized based on the distribution characteristics of the ticket data. Subsequently, the parameters are fine-tuned during the subsequent training process based on the experimental results. The model was trained on two separate ticket datasets for 400 and 300 epochs, respectively.

Algorithm 1 Seq2seq-based Recurrent Expert Recommendation

Input: a description of the current ticket D; the resolution of the current ticket R; the length of recommended expert sequence N; the expert label of the ticket E.
Output: the recommended expert sequence S.
1: $S = \emptyset$ //generate a empty sequence
2: $z_1 = encoder(D)$
3: $\overline{R} = decoder(z_1)$ //$\overline{R}$ is predicted resolution by decoder
4: $\overline{E_1} \Leftarrow recommendation(z_1)$ // get the first recommendation result
5: $e_1 \Leftarrow get(\overline{E_1})$ // get the most probable expert
6: $S.append(e_1)$
7: **while** $S.length < N$ **do** // get the remaining recommended experts
8: $\quad \boldsymbol{W}_n = tanh\left(\tilde{W} \times \left[\overline{E}_{n-1}, H\right]\right)$
9: $\quad \boldsymbol{z}_n = \boldsymbol{W}_n \times H$ //get new hidden state vector
10: $\quad \overline{E_n} \Leftarrow recommendation(z_n)$ // get the n_{th} recommendation result
11: $\quad e_n \Leftarrow get(\overline{E_n})$ // get the most probable expert
12: $\quad S.append(e_n)$
13: **end while**

The trained model can be used to recommend experts for incoming tickets. The process of prediction is similar to the training process, the description of a new ticket will be input into a pre-trained model. The model generates the next recommendation result based on the assumption that the currently recommended expert is unable to solve the ticket, and finally obtains a specified number of expert recommendation sequences. Then, the new ticket will be assigned to the first expert in the recommended sequence. If the expert cannot solve the ticket, the ticket will be passed on to the subsequent experts in the sequence until the ticket is resolved or the time limit is exceeded. If the incoming ticket can be solved by an expert in the recommendation sequence, we consider that the recommendation is effective. Furthermore, the true resolver is ranked higher in the recommendation sequence, which means the model is more efficient. After the prediction, the resolution proposed by the true resolver for the corresponding ticket can be used to improve the robustness of the model. In other words, if the description translation loss of new tickets reaches the predefined threshold, the model can be updated based on these tickets.

4 Experiments

In this section, we conduct experiments on two real-world datasets to evaluate the performance of SSR-TA for the expert recommendation and mainly discuss the following issues:

1. *Ablation experiment*: Whether the resolution decoder and recurrent recommendation network are effective for the expert recommendations?
2. *Model exploration*: Do the different structures of the seq2seq model and length of sequence have impacts on the performance of the expert recommendations?
3. *Performance evaluation*: How does the performance of SSR-TA compare to the performance of baselines?
4. *Case study*: What is the difference between the effectiveness of SSR-TA and baselines on the real-world datasets?

The implementations of SSR-TA and baselines are presented at Github: https://github.com/coderxor/SSR-TA.

4.1 Datasets

The ticket datasets, TDS-a and TDS-b, were collected from different IBM IT service system. Table 4 shows the number of tickets and experts in the corresponding datasets. Both datasets contain duplicate tickets and unsolved tickets, 15,188 and 10,693 tickets over two datasets were selected for experiments, and the experts solved tickets less than 50 are removed. The training and testing datasets were split in a ratio of 9:1. All training data for tickets includes a description and its corresponding resolution. Incomplete description texts were not specially processed. Because almost all ticket descriptions contain numbers, dates, and special symbols (such as 'WVPMA584 | 03/02/2014', '%', '#'), and the description of the ticket submitted by the user may contain abbreviations and spelling errors (such as 'immed' means immediate). Then we performed symbol removal, stop word removal, and lemmatization for description and resolution for all selected tickets. Some selected ticket examples from two datasets are shown in Table 5. Due to the sensitive nature of operations and maintenance (O&M) ticket data related to core business operations, there are currently no publicly available O&M ticket datasets. Therefore, the experimental section of this paper utilized ticket data provided by IBM, which primarily originated from the software systems domain and is considered representative to some extent. In the future, if there are publicly available O&M ticket datasets, we will conduct experiments to validate the proposed methods in this paper.

4.2 Metrics

To evaluate the performance of the expert recommendation, three metrics are used: resolution rate (RR) [34], mean step to resolver (MSTR) [35], and mean reciprocal rank (MRR) [11].

RR represents the proportion of the tickets that can be solved. For example, if the true resolver (i.e., expert) to a ticket can be found in the recommendation sequence within the length N, it means the ticket can be solved. RR is defined as follows:

$$\mathrm{RR}(T) = \frac{\sum_{t \in T} R(t)}{\mid T \mid}, \tag{11}$$

where $R(t)$ is 1 if the ticket t is solved and 0 otherwise, while T is the set of tickets.

MSTR represents the average number of steps to reach the resolver in the recommendation sequence. The smaller MSTR is, the more efficient the system is. The minimum value of MSTR is 1, and MSTR is defined as:

$$\mathrm{MSTR}(T) = \frac{\sum_{t \in T} P(t)}{\mid T^* \mid}, \tag{12}$$

where T^* represents the set of tickets solved, and $\mid T^* \mid$ is the number of tickets in the set. For each ticket t, $P(t)$ means the position of the true resolver in the recommendation sequence. For example, if the recommendation sequence is $S = [e_1, e_2, e_n, ..., e_N]$ and true resolver is e_n, then $P(t) = 3$.

MRR represents the ranking of the true resolver corresponding to the ticket within the recommendation sequence:

$$\mathrm{MRR}(T) = \frac{1}{\mid T^* \mid} \sum_{t=1}^{\mid T^* \mid} \frac{1}{rank_t}, \tag{13}$$

where $rank_t$ denotes the position of the corresponding true resolver in the recommendation sequence for the ticket t.

4.3 Baselines

To evaluate the effectiveness of SSR-TA, we compare SSR-TA with several baselines.

1. *SVM* [36]. SVM is a classic linear classifier, which is used to predict the probability of an expert solving a ticket. TF-IDF is used to obtain the representation

Table 4 The number of tickets and experts on two datasets

Dataset	All ticket	Distinct description	Distinct resolution	Expert
TDS-a	72,977	15,188	8574	64
TDS-b	23,935	10,693	7452	256

Table 5 Some ticket examples from two datasets

Dataset	Description	Resolution
TDS-a	sppwa928: Available Real Memory is low. The percentage of available real memory is 5.0 percent	ProblemSolutionText: we have validated the application and its BAU
	AVPMD504: File system / is low. The percentage of used space in the file system is 90 percent. Threshold: 90 percent	Removed excess files and current filesystem usage now lower than threshold. According to System Operations Procedures closing the ticket accordingly
	lvpmi027: A high percentage of system CPU is being used.(99.98 percent)	Server CPU usage was checked and is within normal parameters of operation. This incident will fall into the Defect Prevention Process
	Patrol Agent Offline: Failed to reconnect to PatrolAgent on host AVPMD580, port 3181. Will retry in 3 timer ticks	verified connectivity
	lppww665: The syslogd process is not running	No actions taken since the process is running as expectedThis incident will fall into the Defect Prevention Process
TDS-b	b1pavreconapp01(161.178.193.234) is unreachable. The host has failed to respond to the ping request	Server was rebooted under change # CH165505
	01.11 ET Job DOSW05P Abended at step DOS28090 with CC=SB37	Insufficient space in output dataset
	Corrective Action ACR_AC_SVC_WIN (101) failed. (Received during suppression) Service in alert state	Service will be started automatically whenever required
	174312 9015.00 SEV-2 TRANSFER STEP: FAILED FILE: TRANSEAPPRD. EPTRN3 #M06EC122355Z9Q	File has been re-transmitted successfully
	PowerHA Event reported by AVPMD592 AVPMD592 Mar 1 02:19:22 AVPMD592 user:notice HACMP for AIX: EVENT COMPLETED:network_up_complete AVPMD592 net_ether_02 0	Issue being worked on imr 19027973

vector of the ticket description which is considered as the input of SVM.

2. *XGBoost* [37]. XGBoost is a scalable machine learning system for tree boosting, which is to predict whether the expert can solve the ticket. TF-IDF is still used to learn the representation of ticket descriptions.
3. *RNN-uniform* [38]. RNN-based text classification is a deep learning approach, and the LSTM layer can capture semantic information in variable-length sequences. The word embedding of ticket description is considered as the input of the RNN model which is to provide the expert recommendation sequence.
4. *CNN-rand* [39]. This model casts the words in a ticket description as an embedding matrix, where the matrix will be classified through convolutional and max-pooling layers.
5. *DeepRouting* [11]. DeepRouting applies the text and graph similarity to predict the true resolver of the ticket. The deep structured semantic model is used to obtain the text-view representation of a ticket, and the graph convolutional network is adopted to learn the graph-view representation of expert relationships. For each expert, we sample 40 tickets it solved for training the GCN model. Moreover, according to the parameter settings in the original text, for each positive pair (i.e., a ticket which is not solved by the expert), 19 negative experts are sampled.
6. *BART* [31]. BART is a denoising autoencoder combining bidirectional and auto-regressive transformers for pretraining sequence-to-sequence model. We applied pre-trained BART to recommend the ticket expert sequence by giving the ticket description.

4.4 Ablation experiment

In this section, we investigate whether the resolution decoder and the recurrent recommendation network are effective for the expert recommendation.

Figure 3 shows the resolution rate and Table 6 shows MRR and MSTR of SSR-TA using different combinations of components, where ① is the description encoder, ② is the resolution decoder, and ③ is the recurrent recommendation network. As observed from Fig. 3 and Table 6, the models without the resolution decoder (i.e., ① and ①+③) perform worse than the seq2seq-based models (i.e., ①+② and ①+②+③). In other words, the description translation loss is capable of improving the representation of ticket description which is effective for the expert recommendation. Moreover, the models with ③ outperform those without ③ (especially when the recommended sequence

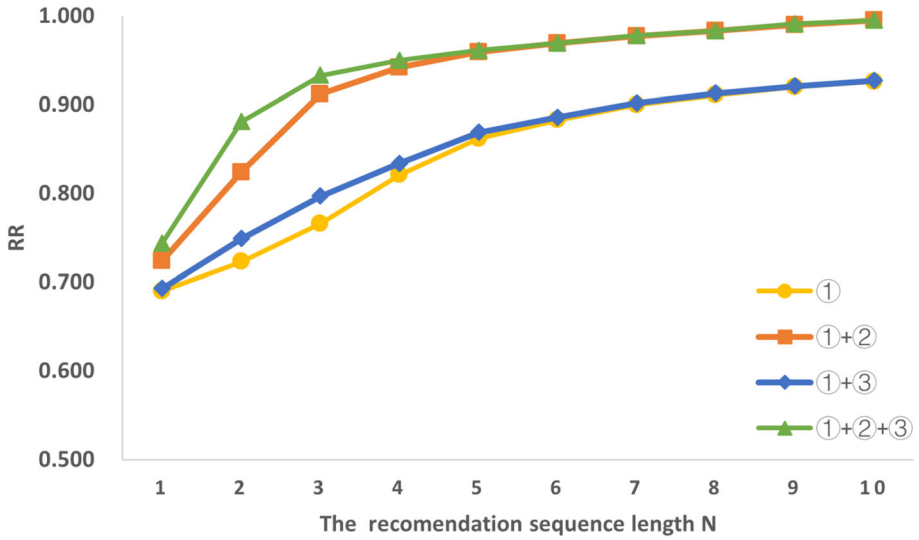
length is less than 5), which means that the recurrent recommendation network improves the ranking of the true resolver in the recommendation sequence. The phenomenon further supports our assumption that the model will recommend another appropriate expert when the previous experts in the sequence cannot solve the ticket. In summary, the experimental results show that the seq2seq structure improves the effectiveness of the expert recommendation, while the recurrent recommendation network improves the efficiency by reducing the number of steps to find true experts.

Table 6 The MRR and MSTR of ablation experiment

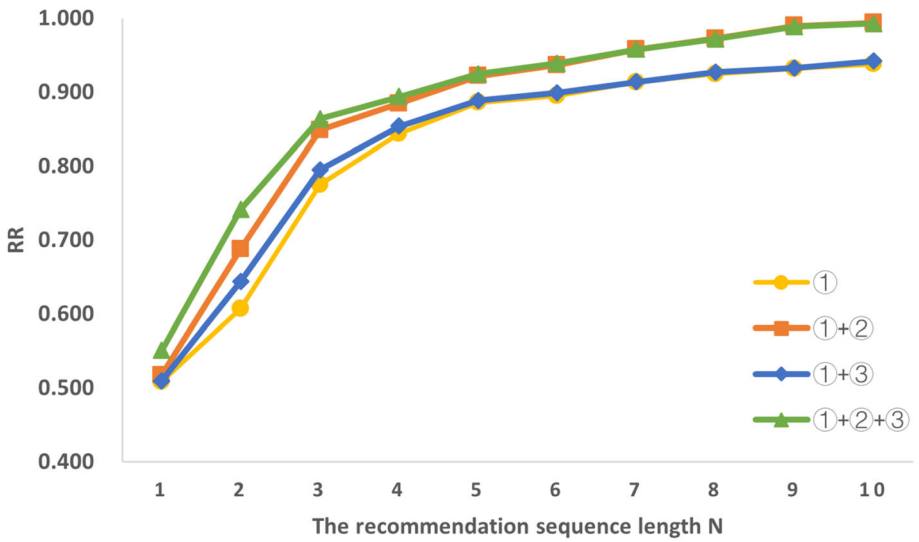
Model	TDS-a		TDS-b	
	MRR	MSTR	MRR	MSTR
①	0.719	1.912	0.670	2.464
①+②	0.828	1.474	0.730	2.117
①+③	0.731	1.899	0.681	2.422
①+②+③	**0.843**	**1.352**	**0. 743**	**2.043**

The representations of terms are: ① the description encoder, ② the resolution decoder, ③ the recurrent recommendation network

Bold indicates the best performance in the corresponding column (i.e., metric)

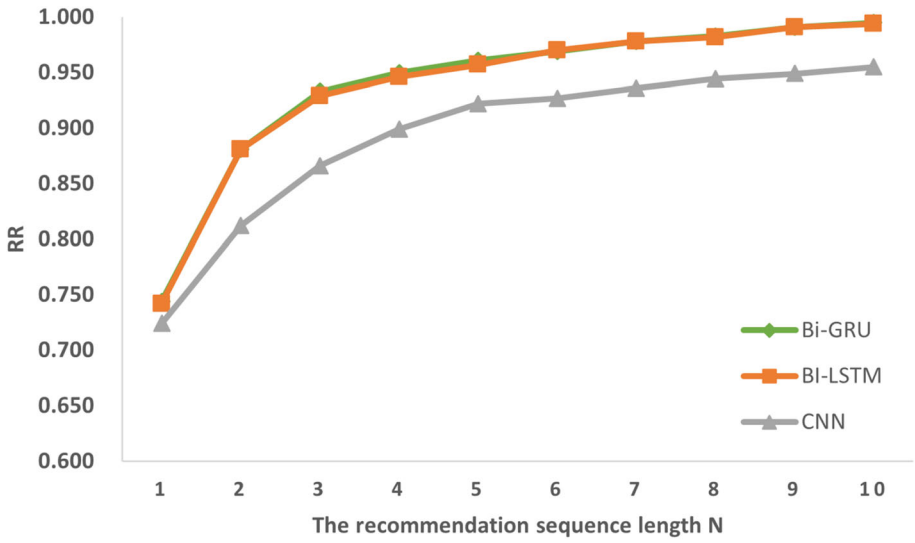


(a) TDS-a

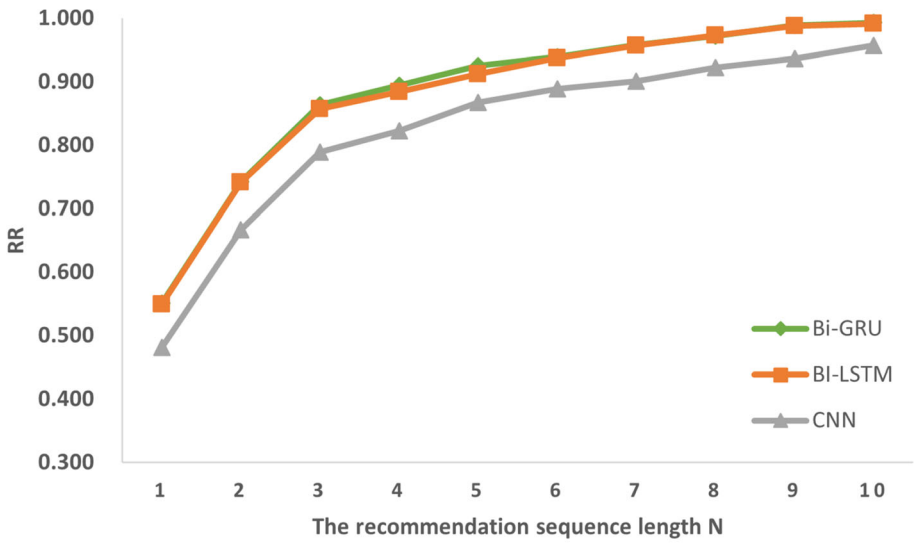


(b) TDS-b

Fig. 3 The RR of ablation experiment



(a) TDS-a



(b) TDS-b

Fig. 4 The resolution rate of model exploration

Table 7 The MRR and MSTR of model exploration

Model	TDS-a		TDS-b	
	MRR	MSTR	MRR	MSTR
Bi-GRU	**0.843**	**1.352**	**0.743**	**2.043**
Bi-LSTM	0.842	**1.352**	0.741	2.044
CNN	0.815	1.619	0.691	2.379

Bold indicates the best performance in the corresponding column (i.e., metric)

4.5 Model exploration

In this section, we explore if the different structures of the seq2seq model have impacts on the performance for the expert recommendations and the impact of the length of the ticket text sequence on the recommended results.

Figure 4 represents the resolution rate and Table 7 shows MRR and MSTR of SST-TA with different seq2seq structures, where Bi-GRU and Bi-LSTM are RNN-based seq2seq models, and CNN-based seq2seq model is composed of convolutional layers. The result shows that the RNN-based models perform better than CNN-based model (especially on TDS-a), which means the sequential features

Table 8 The RR@10 of different sequence length

RR@10	TDS-a			TDS-b		
	LoD@10	LoD@20	LoD@30	LoD@20	LoD@40	LoD@60
LoR@10	0.877	0.803	0.974	0.955	0.958	0.982
LoR@20	0.902	0.887	0.896	0.957	0.956	0.965
LoR@30	0.937	0.815	0.995	0.968	0.960	0.993

LoD length of ticket description, **LoR** length of ticket resolution, *RR* resolution rate

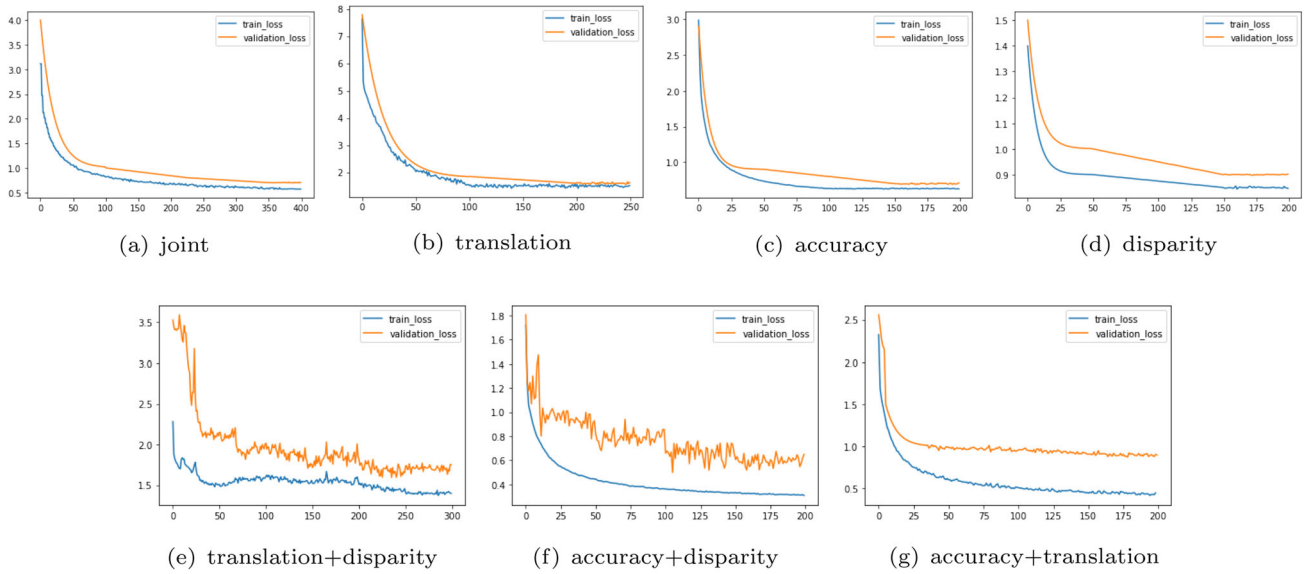


Fig. 5 Convergence status of the model and the combination of various components on TDS-a

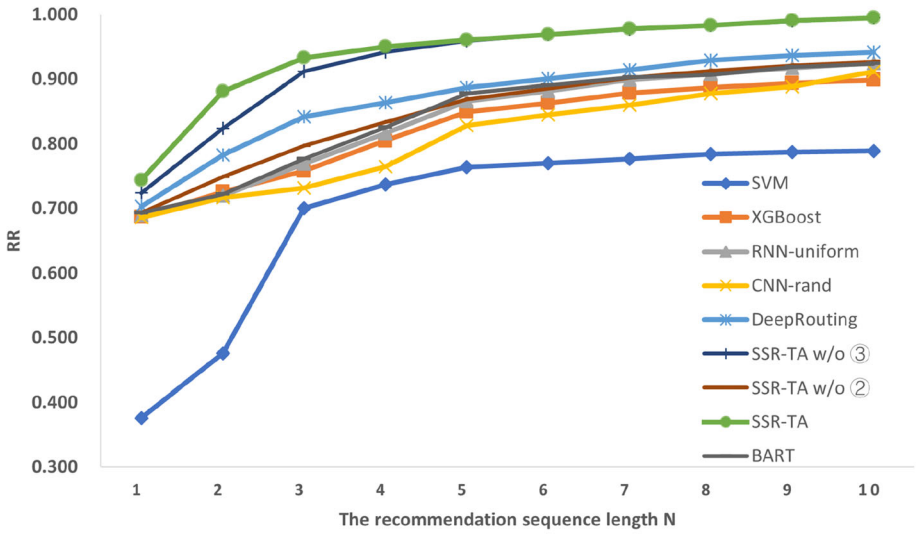
captured by the description translation are effective for the expert recommendation. In addition, the performance of RNN-based models is close due to the similar structure of Bi-GRU and Bi-LSTM, while Bi-GRU is slightly better than Bi-LSTM. Overall, Bi-GRU is applied to construct SSR-TA because Bi-GRU has better performance and fewer parameters for faster training.

Table 8 shows the impact of different ticket description and resolution sequence lengths on the ticket resolution rate (with the recommended sequence length set to 10). In dataset TDS-a, we selected data with ticket description length less than 30 for experimentation, the impact of ticket description lengths of 10, 20, and 30 on the ticket resolution rate were explored. The experimental results show that, overall, the longer the ticket description length, the more ticket information it contains, and the higher the ticket resolution rate. In dataset TDS-b, all tickets with description length less than 60 was used for experimentation, the impact of ticket description lengths of 20, 40, and 60 on the ticket resolution rate were explored. The experimental results show that the ticket resolution rate increases with the increase in ticket description length, but the overall change is not significant, possibly due to the weaker ticket description sequence information in this dataset.

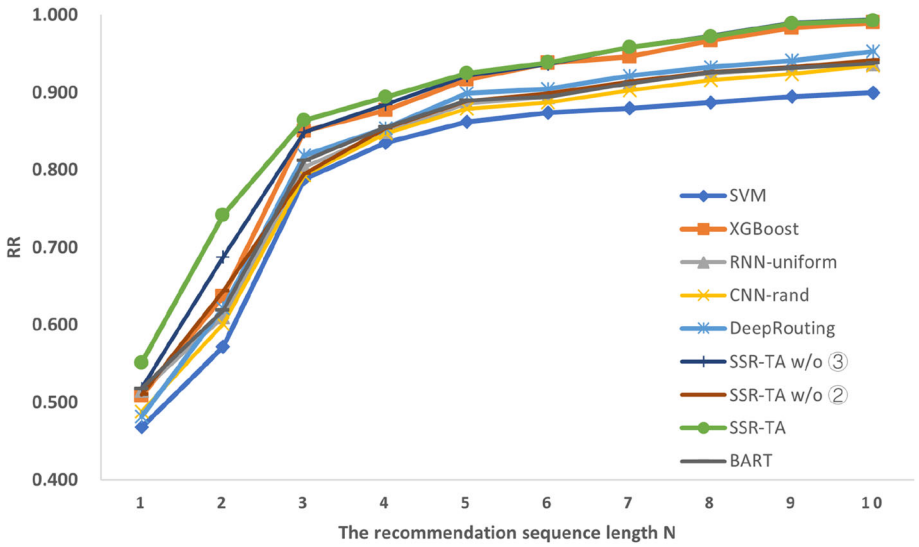
Figure 5 shows the convergence status of the model and the combination of various components on TDS-a. Furthermore, Fig. 5(a)-(d) illustrates the performance of the joint objective function, the objective function for the description translation, the objective function for the accuracy of recommendation, and the objective function for the disparity of recommendation, respectively. In addition, Fig. 5(e)-(g) show the performance of the combination of corresponding objective functions. As observed from Fig. 5, the initial loss value of the objective function for the disparity of recommendation is the smallest, we hope that the model can pay more attention to the disparity of experts in the recommended sequence. Therefore, we set α_3 to 0.5, and fine-tuned α_1 and α_2 through grid search based on the experimental results. In addition, the training of the combination of various components converged after 300 epochs, indicating the effectiveness of each component in SSR-TA.

4.6 Performance evaluation

In this section, we compare SSR-TA with the baselines on the real-world datasets, where Fig. 6 illustrates the resolution rate and Table 9 shows MRR and MSTR. For SVM



(a) TDS-a



(b) TDS-b

Fig. 6 The RR comparisons of SSR-TA with baselines

and other classifiers, the historical ticket descriptions are used as model inputs, while the experts are treated as classification labels. The classifier predicts the probabilities of the ticket belonging to each expert based on the ticket description. The expert recommendation sequence is obtained by sorting the probabilities, and the MRR and MSTR metrics are calculated based on the position of the true expert in the sequence.

As observed from Fig. 6 and Table 9, SSR-TA outperforms other baselines. Notice that the RR of SSR-TA and DeepRouting increases faster than other baselines on TDS-a, especially when the length of the recommendation sequence is less than 3 (i.e., $N \leq 3$). The experimental results show that the recurrent recommendation of SSR-TA and the ticket transfer of DeepRouting effectively improve the ranking of the true resolver in the recommendation sequence. In addition, on TDS-b, different from DeepRouting, SSR-TA also performs better in RR when $N \leq 3$, which means that the generalization ability of the recurrent recommendation network is better than that of the ticket transfer. Additionally, the pre-trained BART is outperformed by DeepRouting and SSR-TA. Compared to BART, which only makes one-time recommendations based on ticket descriptions, SSR-TA adjusts the subsequent recommendations with the recurrent recommendation network, which reduces the steps of resolving tickets. Furthermore, the performance of SSR-TA w/o ② is close to that of other baselines, which further shows that the

Table 9 The MSTR and MRR comparisons of SSR-TA with baselines

Model	TDS-a		TDS-b	
	MRR	MSTR	MRR	MSTR
SVM	0.575	2.788	0.629	2.614
XGBoost	0.704	1.972	0.719	2.197
RNN-uniform	0.720	1.947	0.676	2.451
CNN-rand	0.692	2.123	0.669	2.485
DeepRouting	0.767	1.793	0.686	2.401
BART	0.723	1.923	0.678	2.450
SSR-TA w/o ③	0.828	1.474	0.730	2.117
SSR-TA w/o ②	0.731	1.899	0.681	2.422
SSR-TA	**0.843**	**1.352**	**0.743**	**2.043**

Bold indicates the best performance in the corresponding column (i.e., metric)

Table 10 The training time comparisons of SSR-TA with baselines

Model	Training time (s)	
	TDS-a	TDS-b
SVM	321	68
XGBoost	10	3
RNN-uniform	1635	685
CNN-rand	807	316
DeepRouting	5913	2581
BART	11,843	5059
SSR-TA w/o ③	1832	747
SSR-TA w/o ②	1683	664
SSR-TA	1866	794

Table 11 The number of user-generated and system-generated tickets on two datasets

Dataset	User-generated	System-gengerated	Total
TDS-a	13,680	1508	15,188
TDS-b	4103	6590	10,693

Table 12 The RR@10 of different type tickets on two datasets

RR@10	User-generated	System-gengerated	Total
TDS-a	0.998	0.987	0.995
TDS-b	0.996	0.980	0.993

resolution decoder is necessary and effective for the expert recommendation.

In addition, Table 10 shows the training time of SSR-TA and baselines. All models use the same datasets(TDS-a and TDS-b) and experimental environment and are implemented with PyTorch, and the test system is Linux with 32 GB of RAM and 1080 Ti GPU. The results show that the training time of SSR-TA is more than that of traditional classification models (e.g., SVM and XGBoost), close to other RNN-based methods (e.g., RNN-uniform), but less than that of DeepRouting and BART. DeepRouting is time-consuming due to the application of the graph model GCN, while BART is constructed based on BERT, which is a model comprising millions of parameters. In summary, compared to DeepRouting and BART, the performance of SSR-TA is more effective and efficient, while comparing to the traditional methods (i.e., SVM and XGBoost) and neural network-based methods (i.e., RNN-uniform and CNN-rand), SSR-TA is more effective with appropriate efficiency.

In conclusion, the description translation (i.e., the description encoder and the resolution decoder) improves the effectiveness, while the recurrent recommendation network improves the efficiency (i.e., the ranking of the true resolver in the recommendation sequence) of the expert recommendation.

4.7 Case study

As observed from the comparison between SSR-TA and the baselines in the real-world datasets, we notice that the performance of the baselines and SSR-TA are similar on TDS-b, which is different from that on TDS-a. Therefore, we further investigate the difference between TDS-a and TDS-b in this section.

As mentioned earlier, TDS-a and TDS-b are collected from different IT managed service systems; therefore, there exist some differences between tickets. For example, the description of a user-generated ticket "AGPMA506: A high percentage of CPU is being used. (93 percent)," which is similar to the natural language. However, the system automatically generated tickets such as "EXPCHNG_8193_WVPMA582 WVPMA582 06:05:43 Message Error Information: 1) Exception Details: Error Code:8193 Severity Code:2 Message:Exception occurred at Business Layer while Adding SourceRequired Error Type," which are mainly constructed by a template containing a large number of non-English words (e.g., EXPCHNG). In other words, the RNN-based model demonstrates proficiency in learning from user-generated tickets that contain substantial sequential information. However, it faces challenges in capturing the underlying representation of system-generated tickets that lack meaningful sequential information, resulting in ineffective representation of such tickets. Table 11 presents the proportion of the user-generated tickets and the system-generated tickets in TDS-a and TDS-b. As observed from Table 11, the proportion of system-generated tickets in

TDS-b is much higher than that in TDS-a. Table 12 shows that when the recommended sequence length is 10, the RR of the model for user-generated tickets is slightly higher than that for system-generated tickets on both datasets. Moreover, Fig. 6a shows that RNN-uniform performs better than other traditional top-N recommendation models, while RNN-uniform performs similar to other baselines in Fig. 6b. In other words, the RNN-based models (i.e., SSR-TA and RNN-uniform), which are capable of capturing the sequential feature, perform better on TDS-a containing more user-generated tickets.

4.8 Discussion

The proposed SSR-TA in this paper is used to recommend experts to solve tickets. The seq2seq model captures feature information by translating ticket descriptions into corresponding resolutions, and the recurrent recommendation network is designed to obtain expert recommendation sequences.

For ticket recommendation, SSR-TA applied the seq2seq model to combine the features from ticket description and resolution to assist the expert recommendation, and it also can be used for ticket resolution generation. Moreover, the recurrent recommendation network adopts an attention mechanism to generate the expert recommendation sequences based on the assumption that the previous expert in the sequence failed to solve tickets. The recurrent recommendation network improves the ranking of the true resolver in the recommendation sequence, which means the model reduces the steps to find appropriate experts for solving tickets. In other words, the application of SSR-TA is capable of solving anomalies more quickly and saving more human costs in practice.

Finally, there exist some limitations to our proposed method. The performance of expert recommendations for the system-generated tickets is worse than that for the user-generated tickets. Because the seq2seq model (i.e., SSR-TA) is specialized in capturing the sequential information from natural language (i.e., user-generated tickets), the system-generated tickets mainly consist of unfamiliar words and specific terms without contextual information. SSR-TA is also affected by the quality of the ticket data. Unknown tickets and data imbalance can lead to inaccuracies in the model's recommendation results. In addition, training SSR-TA is much more time-consuming than training those traditional machine learning methods (i.e., SVM and XGBoost), which is difficult to apply in practice. Furthermore, limited to the architecture of SSR-TA, SSR-TA can select and recommend experts or existing resolutions for the known tickets; however, SSR-TA cannot generate personalized resolutions for unknown tickets.

5 Conclusion

In this paper, we proposed a seq2seq-based model SSR-TA combined with a recurrent recommendation network to recommend appropriate experts for the ticket automation. The description of solved tickets is translated into the corresponding resolution for capturing the potential and useful features, which is capable of representing tickets and improving the effectiveness of the expert recommendation. In addition, the recurrent recommendation network improves the ranking of the true resolver in the recommendation sequence based on the assumption that the previous expert cannot solve the ticket. The experimental results show that SSR-TA improves the effectiveness and the efficiency of the expert recommendation.

In future work, we will continue to study the following issues. First, we are going to improve the initial resolution rate of the model in processing system-generated tickets. Second, the decoder of SSR-TA is not used when recommending an expert sequence for an incoming ticket. We will recommend historical resolutions for the incoming ticket using the similarity between historical resolutions and the decoder output. Finally, we plan to explore more variants of seq2seq for ticket expert recommendation.

Acknowledgements This work was supported by the National Natural Science Foundation of China under Grant No. 61872186, the Natural Science Foundation of Nanjing University of Posts and Telecommunications (Grant No. NY221070).

Data availibility The datasets analyzed during the current study are not publicly available due it involves private information from IBM but are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Moss S (2021) IBM cloud suffers second outage in five days. DataCenterDynamics. https://www.datacenterdynamics.com/en/news/ibm-cloud-suffers-second-outage-in-five-days/. Accessed 26 May 2021
2. Palacios D, Morillas C, Garcés M et al (2019) Big data-empowered system for automatic trouble ticket generation in IoT networks. In: 2nd IEEE 5G World Forum, 5GWF 2019, Dresden, Germany, 30 Sept–2 Oct, 2019. IEEE, pp 63–68. https://doi.org/10.1109/5GWF.2019.8911636
3. Gupta M, Asadullah A, Padmanabhuni S et al (2018) Reducing user input requests to improve it support ticket resolution process. Empir Softw Eng 23(3):1664–1703
4. Shimpi V, Natu M, Sadaphal V et al (2014) Problem identification by mining trouble tickets. In: Paper presented at the 20th international conference on management of data, pp 76–86

5. Han J, Goh KH, Sun A et al (2018) Towards effective extraction and linking of software mentions from user-generated support tickets. In: Paper presented at the 27th ACM international conference on information and knowledge management, pp 2263–2271
6. Potharaju R, Jain N, Nita-Rotaru C (2013) Juggling the jigsaw: towards automated problem inference from network trouble tickets. In: Paper presented at 10th USENIX symposium on networked systems design and implementation (NSDI 13), pp 127–141
7. Xu J, Zhang H, Zhou W et al (2018) Signature based trouble ticket classification. Future Gener Comput Syst 78:41–58. https://doi.org/10.1016/j.future.2017.07.054
8. Zeng C, Zhou W, Li T et al (2017) Knowledge guided hierarchical multi-label classification over ticket data. IEEE Trans Netw Serv Manag 14(2):246–260
9. Zhou W, Xue W, Baral R et al (2015) Star: a system for ticket analysis and resolution. In: Paper presented at the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, pp 2181–2190
10. Wang Q, Zeng C, Iyengar S et al (2015) Aistar: an intelligent system for online it ticket automation recommendation. In: Paper presented at 2018 IEEE international conference on big data (Big Data), pp 1875–1884
11. Han J, Sun A (2020) Deeprouting: a deep neural network approach for ticket routing in expert network. In: Paper presented at 2020 IEEE international conference on services computing (SCC), pp 386–393
12. Blei DM, Ng AY, Jordan MI (2003) Latent dirichlet allocation. J Mach Learn Res 3:993–1022
13. Huang P, He X, Gao J et al (2013) Learning deep structured semantic models for web search using clickthrough data. In: He Q, Iyengar A, Nejdl W et al (eds) 22nd ACM international conference on information and knowledge management, CIKM'13, San Francisco, CA, USA, 27 Oct–1 Nov, 2013. ACM, pp 2333–2338. https://doi.org/10.1145/2505515.2505665
14. Zhou W, Zhu X, Li T et al (2014) Multi-view feature selection for labeling noisy ticket data. In: Paper presented at NOMS 2018-2018 IEEE/IFIP network operations and management symposium, pp 1–4
15. Xu J, Mu J, Chen G (2020) A multi-view similarity measure framework for trouble ticket mining. Data Knowl Eng 127:101800
16. Molino P, Zheng H, Wang Y (2018) COTA: improving the speed and accuracy of customer support through ranking and deep networks. In: Guo Y, Farooq F (eds) Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, KDD 2018, London, UK, 19–23 Aug, 2018. ACM, pp 586–595. https://doi.org/10.1145/3219819.3219851
17. Fuchs S, Drieschner C, Wittges H (2022) Improving support ticket systems using machine learning: a literature review. In: 55th Hawaii international conference on system sciences, HICSS 2022, Virtual Event/Maui, Hawaii, USA,4–7 Jan, 2022. ScholarSpace, pp 1–10. http://hdl.handle.net/10125/79570
18. Zangari A, Marcuzzo M, Schiavinato M et al (2023) Ticket automation: an insight into current research with applications to multi-level classification scenarios. Expert Syst Appl 225:119984. https://doi.org/10.1016/j.eswa.2023.119984
19. Zhou W, Tang L, Zeng C et al (2016) Resolution recommendation for event tickets in service management. IEEE Trans Netw Serv Manag 13(4):954–967
20. Zhou W, Li T, Shwartz L et al (2015) Recommending ticket resolution using feature adaptation. In: Paper presented at 2015 11th international conference on network and service management (CNSM), pp 15–21
21. Xu J, He R (2018) Expert recommendation for trouble ticket routing. Data Knowl Eng 116:205–218
22. Han J, Li J, Sun A (2020) UFTR: a unified framework for ticket routing. Preprint at https://arxiv.org/abs/2003.00703
23. Raheja V, Tetreault J (2019) Dialogue act classification with context-aware self-attention. In: Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (Long and Short Papers). Association for Computational Linguistics, Minneapolis, Minnesota, pp 3727–3733. https://doi.org/10.18653/v1/N19-1373. https://aclanthology.org/N19-1373
24. Deng Y, Zhang W, Lam W (2020) Opinion-aware answer generation for review-driven question answering in e-commerce. In: d'Aquin M, Dietze S, Hauff C et al (eds) CIKM '20: The 29th ACM international conference on information and knowledge management, virtual event, Ireland, 19–23 Oct, 2020. ACM, pp 255–264. https://doi.org/10.1145/3340531.3411904
25. Sutskever I, Vinyals O, Le QV (2014) Sequence to sequence learning with neural networks. Adv Neural Inf Process Syst 27
26. Bahdanau D, Cho K, Bengio Y (2014) Neural machine translation by jointly learning to align and translate. Preprint at https://arxiv.org/abs/1409.0473
27. Keneshloo Y, Shi T, Ramakrishnan N et al (2019) Deep reinforcement learning for sequence-to-sequence models. IEEE Trans Neural Netw Learn Syst 31:2469–2489
28. Peng X, Song L, Gildea D et al (2019) Sequence-to-sequence models for cache transition systems. In: Paper presented at the 56th annual meeting of the association for computational linguistics (Volume 1: Long Papers), pp 1842–1852
29. Ma D, Li S, Wu F et al (2019) Exploring sequence-to-sequence learning in aspect term extraction. In: Paper presented at the 57th annual meeting of the association for computational linguistics, pp 3538–3547
30. Huang J, Pang G, Kovvuri R et al (2019) A multiplexed network for end-to-end, multilingual OCR. In: Paper presented at the IEEE/CVF conference on computer vision and pattern recognition, pp 4547–4557
31. Lewis M, Liu Y, Goyal N et al (2020) BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In: Jurafsky D, Chai J, Schluter N et al (eds) Proceedings of the 58th annual meeting of the association for computational linguistics, ACL 2020, 5–10 July, 2020. Association for Computational Linguistics, pp 7871–7880. https://doi.org/10.18653/v1/2020.acl-main.703
32. Mao Y, He P, Liu X et al (2021) Generation-augmented retrieval for open-domain question answering. In: Zong C, Xia F, Li W et al (eds) Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), virtual event, 1–6 Aug, 2021. Association for Computational Linguistics, pp 4089–4100. https://doi.org/10.18653/v1/2021.acl-long.316
33. Goyal A, Lamb A, Zhang Y et al (2016) Professor forcing: a new algorithm for training recurrent networks. In: Lee DD, Sugiyama M, von Luxburg U et al (eds) Advances in neural information processing systems 29. Annual conference on neural information processing systems 2016, 5–10 Dec, 2016, Barcelona, Spain, pp 4601–4609. https://proceedings.neurips.cc/paper/2016/hash/16026d60ff9b54410b3435b403afd226-Abstract.html
34. Sun P, Tao S, Yan X et al (2010) Content-aware resolution sequence mining for ticket routing. In: Hull R, Mendling J, Tai S (eds) Business process management—8th international conference, BPM 2010, Hoboken, NJ, USA, 13–16 Sept, 2010. Proceedings, lecture notes in computer science, vol 6336. Springer, pp 243–259. https://doi.org/10.1007/978-3-642-15618-2_18

35. Shao Q, Chen Y, Tao S et al (2008) Efficient ticket routing by resolution sequence mining. In: Proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining, pp 605–613
36. Chang C, Lin C (2011) LIBSVM: a library for support vector machines. ACM Trans Intell Syst Technol 2(3):1–27. https://doi.org/10.1145/1961189.1961199
37. Chen T, Guestrin C (2016) Xgboost: a scalable tree boosting system. In: Krishnapuram B, Shah M, Smola AJ et al (eds) Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, San Francisco, CA, USA, 13–17 Aug, 2016. ACM, pp 785–794. https://doi.org/10.1145/2939672.2939785
38. Liu P, Qiu X, Huang X (2016) Recurrent neural network for text classification with multi-task learning. In: Kambhampati S (ed) Proceedings of the twenty-fifth international joint conference on artificial intelligence, IJCAI 2016, New York, NY, USA, 9–15 July 2016. IJCAI/AAAI Press, pp 2873–2879. http://www.ijcai.org/Abstract/16/408
39. Kim Y (2014) Convolutional neural networks for sentence classification. In: Moschitti A, Pang B, Daelemans W (eds) Proceedings of the 2014 conference on empirical methods in natural language processing, EMNLP 2014, 25–29 Oct, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL. ACL, pp 1746–1751. https://doi.org/10.3115/v1/d14-1181