Sets2Sets: Learning from Sequential Sets with Neural Networks

Haoji Hu
University of Minnesota, Twin Cities
Minneapolis, Minnesota
huxxx899@umn.edu

Xiangnan He*
University of Science and Technology of China
Hefei, Anhui
xiangnanhe@gmail.com

ABSTRACT

Given past sequential sets of elements, predicting the subsequent sets of elements is an important problem in different domains. With the past orders of customers given, predicting the items that are likely to be bought in their following orders can provide information about the future purchase intentions. With the past clinical records of patients at each visit to the hospitals given, predicting the future clinical records in the subsequent visits can provide information about the future disease progression. These useful information can help to make better decisions in different domains.

However, existing methods have not studied this problem well. In this paper, we formulate this problem as a sequential sets to sequential sets learning problem. We propose an end-to-end learning approach based on an encoder-decoder framework to solve the problem. In the encoder, our approach maps the set of elements at each past time step into a vector. In the decoder, our method decodes the set of elements at each subsequent time step from the vectors with a set-based attention mechanism. The repeated elements pattern is also considered in our method to further improve the performance. In addition, our objective function addresses the imbalance and correlation existing among the predicted elements. The experimental results on three real-world data sets show that our method outperforms the best performance of the compared methods with respect to recall and person-wise hit ratio by $2.7 - 20.6\%$ and $2.1 - 26.3\%$, respectively. Our analysis also shows that our decoder has good generalization to output sequential sets that are even longer than the output of training instances.

CCS CONCEPTS

• **Information systems → Information systems applications**.

KEYWORDS

Sequential sets, deep learning, temporal data forecasting

ACM Reference Format:
Haoji Hu and Xiangnan He. 2019. Sets2Sets: Learning from Sequential Sets with Neural Networks. In *The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3292500.3330979

*Xiangnan He is the corresponding author.

KDD '19, August 4–8, 2019, Anchorage, AK, USA

ACM ISBN 978-1-4503-6201-6/19/08…$15.00
https://doi.org/10.1145/3292500.3330979

1 INTRODUCTION

Given the history records, forecasting the future events is widely studied in different domains. It provides useful information for decision-making and planing. Forecasting the NYSE (The New York Stock Exchange) composite index can help the traders to evaluate the risk in stock market [19]. Predicting the grade of a course that the students will obtain in the next term can support them to make decisions on choosing the suitable courses [21]. Predicting the items that the customers will buy next time can help to make decisions on choosing items recommended to the customers [31]. In these applications, the methods predict an aggregated element/value or a set/sequence of elements/values based on the past records.

However, in some applications, we need to deal with more complex data — sequential sets in which each sequence consists of sets instead of elements. In retail area, given the past orders of the customers, predicting the items that will appear in their subsequent orders can provide information about the change of the purchase intentions in the future. In medical area, given the services and diagnoses in patients' past visits to the hospitals, predicting the services and diagnoses in their subsequent visits can provide information about the possible future disease progression. If we denote a visit/order as a set, the task becomes a new type of forecasting problem — sequential sets forecasting in which the input and output are both sequential sets. This problem is a generalization to time series forecasting [6] and sequence prediction [4] that only has a single value/element at each time step.

Another line of research related to our problem can be categorized as next set prediction. Next basket recommendation [31][29][22], next clinical events prediction [3][9] and next repeat set prediction [5] belong to this category. All of these methods predict the next set with the past sequential sets given. Even though we can tweak this type of methods to recurrently or directly predict the subsequent sets, our empirical analysis shows that it causes the sequential structure information loss in the output and the error accumulation in the input.

In this paper, we aim to use the past sequential sets to predict the subsequent sequential sets. We formulate this sequential sets forecasting to *sequential sets to sequential sets learning* problem whose input and output both are a sequence of sets (sequential sets). It can be seen as a generalization to *sequence to sequence learning* (seq2seq) [25] whose input and output both are a sequence of elements. We introduce a recurrent neural networks based method **Sets2Sets** to solve this problem. Our contributions are as follows:

- We formulate a new type of problem — sequential sets to sequential sets learning problem. An encoder-decoder framework called Sets2Sets is proposed to solve this problem. Sets2Sets can learn the complex relations among the elements within and across different sets.

- Our encoder-decoder framework uses an element-embedding-based set embedding to represent each set and uses an attention mechanism to leverage the information in different input sets for different output sets accordingly.
- A repeated elements based element-element component is proposed to capture the element-element interactions across different sets. It utilizes the observation that the elements frequently appearing in the past can have high probability to appear in the future.
- Our objective function addresses the label imbalance and exploits the correlation among labels.
- We conduct experiments on three real-world data sets , one about diabetes medical claims data and two about grocery data, to demonstrate the effectiveness of our method. Our analysis also shows the superiority of our decoding method, compared to other options.

The rest of the paper is organized as follows: Section 2 surveys the related work. Section 3 describes the limitations of related methods. Section 4 defines the problem. Section 5 introduces our proposed method. Section 6 evaluates our method. Finally, section 7 provides some concluding remarks.

2 RELATED WORK

Time Series Forecasting and Sequence Prediction Time series forecasting [6] and sequence prediction [4] have been widely studied. Our problem is also a sequential data forecasting problem. However, the study towards forecasting sequence of sets is lacked.

Next Set Prediction Next set prediction has been studied in different areas. In e-commerce, next basket recommendation [31][29][22] is an important component for many e-commerce website. FPMC [22] is a classical next basket recommendation method proposed by Rendle et al. It learns both sequential behaviors and users' personal tastes. Yu et. al. [31] propose a state-of-the-art next basket recommendation based on RNNs to capture the temporal dynamic of the orders that is ignored by Wang et al. [29]. In medical area, predicting the clinical events [9] in the next visit can allow clinicians to anticipate the patient status at the time of visit. The method proposed by Choi et al. [9] is similar to the next basket recommendation method [31] except they use linear dimension reduction instead of embedding for the input vector. Tian et.al. [3] also propose a similar RNN based next visit prediction method with extra attention mechanism based on the elapsed time between consecutive visits. In our setting, we only focus on the temporal order. Benson et al. [5] study sequential repetition behaviors existing in different kinds of sequences of sets data and propose a stochastic model to capture this behavior. However, there are two limitations of their method. First, their method has the assumption that the next set only consists of elements that have appeared in the past sets. In our applications, the new elements dominate the subsequent sets. Second, their training steps need to generate and store all the ordered set partitions[1] of a set, which is only applicable when the size of the set is very small[2].

[1]https://en.wikipedia.org/wiki/Ordered_Bell_number.

[2] The number of ordered set partitions reaches 2,677,687,796,244,384,203,115 when the set size is 20. The set size in our applications is larger than 20.

Encoder-decoder for Sequence to Sequence Learning Learning a mapping from a sequence to another sequence is called sequence to sequence learning, which is widely studied in machine translation. Existing methods achieve a great success with different encoder-decoder structures [2][25][27]. Our problem is different from the sequence to sequence learning problem in two ways. First, we focus on modelling sequential sets instead of sequence of elements, which indicates our problem has more complex patterns. Second, the sequence to sequence learning is to learn a mapping between two different sequences while the sequential sets in the inputs and outputs of our problem belong to the same sequence.

Recurrent Neural Networks and Attention Mechanism Recurrent neural networks (RNNs) have been successfully applied in many sequence related tasks including speech recognition [24], image captioning [28], and demand forecasting [13]. Different attention mechanisms [2][20] are introduced to improve the performance of RNNs by guiding the networks to focus on particular parts of the input. Recently, attention mechanism is also introduced into recommend system and improves the performance at different recommendation systems [17][7]. We use the attention mechanism to adaptively utilize the information in different past sets.

Multilabel Classification Multilabel classification is to classify instances with a set of labels simultaneously [33]. The correlation among labels can usually improve the performance. Ghamrawi et al. [14] exploit the pair-wised label co-occurrences to improve the performance. Zhang et al. [32] propose a loss function to simultaneously maximize the margin between positive and negative the labels. Since the prediction at each step in our problem is a multilabel classification, we can leverage the characteristics of multilabel classification to improve the performance.

3 LIMITATIONS OF RELATED METHODS

Limitation of time series forecasting As more structure information contained within and across the sets, forecasting sequential sets not only requires us to consider the temporal dynamic but also consider the relations among different elements within and across sets. Current time series forecasting and sequence prediction methods only have one value/element at each step and miss to consider the complex relations in sequential sets data.

Limitation of next set prediction Our problem can be seen as an extension to the next set prediction problem. We can borrow the ideas from time series forecasting [15] to recurrently predict next set or directly predict all the subsequent sets together with next set prediction methods. However, it has limitations. Recurrently predicting next set introduces errors into the input as there are always errors at each next set prediction. And it also loses part of temporal correlation in the subsequent sets. Directly predicting the subsequent sets together seems to avoid this problem, but there is a max number of sets that can be predicted at one shot as the output dimension is fixed. If we need to predict sets longer than this, we still need to use the similar recurrent strategy and suffer from the same problems. We will discuss this in section 6.3.

4 PROBLEM DEFINITION

Denote $\mathbf{D}$ as the set that consists of all the possible elements. Each set (e.g. visit/basket) of a person is represented as a $|\mathbf{D}|$-dimensional

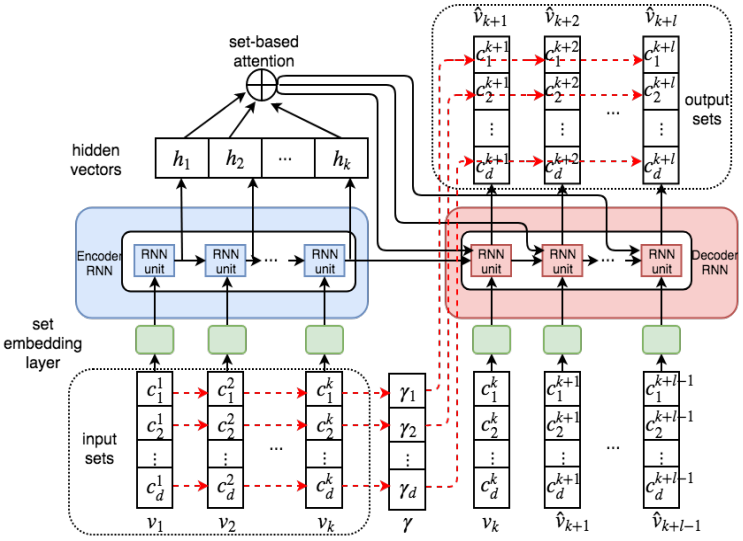


Figure 1: The Encoder-Decoder framework. The input of the first RNN unit in the decoder RNN is the last input in the encoder RNN. The inputs of other RNN units in the decoder RNN are the outputs of their last units.

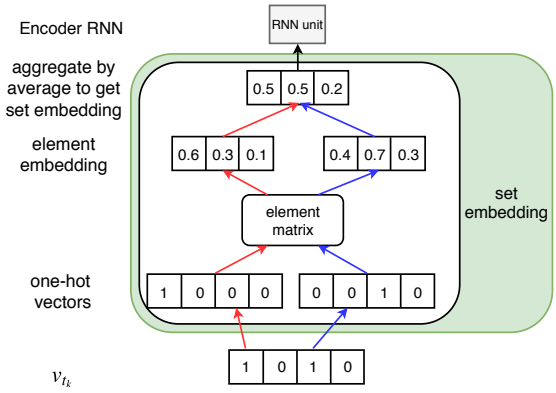


Figure 2: Input a visit/basket vector to set embedding and forward the generated representation into a RNN unit.

vector $\mathbf{v}$ in which each dimension c_m is set to 1 if the corresponding element (e.g. code/item) appears in the set, and 0 otherwise. Then we can formulate our *sequential sets to sequential sets learning problem* as follows:

Given a sequence of sets $\{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_k\}$ of a person, where $\mathbf{v}_i$ is the vector containing the elements appearing in the i-th set, predict the subsequent l sets $\{\hat{\mathbf{v}}_{k+1}, \hat{\mathbf{v}}_{k+2}, ..., \hat{\mathbf{v}}_{k+l}\}$. l is a given parameter.

Existing next set prediction methods treat the predicted set size as a given parameter. We argue that manually setting this parameter can let the users look for results based on their needs. So we still insist on this setting.

5 PROPOSED METHOD

Our encoder-decoder framework is shown in Figure 1. In the encoder, each past sequential set $\mathbf{v}_i$ is first embedded into low dimensional representation x_i by set embedding which will be introduced latter. Then, x_i is forwarded into corresponding RNN unit that

$$h_i = f(x_i, h_{i-1}),$$

to generate the corresponding hidden state vector h_i. The f is some nonlinear function chosen by different variants of RNNs. All the hidden state vectors will be used in the set-based attention. The last hidden state h_k is passed as the initial hidden vector of the decoder.

In the decoder, we have the RNN unit that

$$s_i = g(\hat{x}_i, s_{i-1}, z_i), \tag{1}$$

where $\hat{x}_i$ is the low dimensional representation of embedding $\hat{\mathbf{v}}_{i-1}$ and s_i is the hidden state. The z_i is a context vector obtained from the set-based attention mechanism that will be introduced latter. g is some nonlinear function chosen by different variants of RNNs. The output $\mathbf{o}(\hat{\mathbf{v}}_i)$ of the unit is calculated by

$$\mathbf{o}(\hat{\mathbf{v}}_i) = \text{softmax}(W_s s_i), \tag{2}$$

where W_s is a matrix to project the hidden state back to the space of elements. The softmax function is used to normalize all the entries into the range of $[0, 1]$. The repeat pattern is recorded in vector γ to further improve the prediction.

5.1 Set Embedding

For the original set vectors, we can observe two properties. First, the vector $\mathbf{v}$ has high dimensionality and extreme level of sparsity. Directly using these vectors as the input of RNN will result in a lot of parameters in RNN and each instance only updates a few parameters, which increases the complexity of the RNN model and requires a large number of training data. Second, there are task-specific correlations among different elements within each set. For example, the ICD codes in our data classify all the codes into many categories based on their clinical semantics so that different codes can belong to the same categories. Associate rule analysis shows that some items co-occur in the same baskets [26]. Based on this, we propose to use word embedding that learns a dense vector with small dimensionality for each element in the set. Word embedding is known to generate similar representations for elements with similar task-specific similarity to improve the performance [12]. The word embeddings of all the elements in the same set are aggregated as the embedding for the set by the average pooling. Figure 2 shows an example of the set embedding. The element matrix which transfers the original one-hot vector to the corresponding embedding will be jointly learned within the encoder-decoder framework.

5.2 Set-based Attention

The output sequential sets have temporal correlation relation with input sequential sets. Specifically, we argue that different input sets may have different effects on different output sets. For example, as diabetes causes many complications [23], patients need to visit hospitals regularly. Once the patients are diagnosed with some complications, the treatments and tests will be changed due to this diagnosis. In the grocery shopping case, customers usually consume different items with different time, which means the next time to repeatedly purchase the same type of items are different. So when we try to predict the future elements, the past related elements are expected to have higher effect than other unrelated

elements. We introduce an attention mechanism to focus on the sets containing these past related elements to leverage the effects from the past accordingly. The set-based attention mechanism is used in Equation 1, in which the context vector z_i is a weighted sum of all the hidden states in the encoder and is calculated by

$$z_i = \sum_j \alpha_{ij} h_j.$$

The weight α_{ij} is calculated by

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_k \exp(e_{ik})}$$

where $e_{ij} = a(s_{i-1}, h_j)$. The $a(x, y)$ is implemented as a multi-layer perception that takes the concatenation of x and y as input.

5.3 Modeling Repeated Elements

The set-based attention mechanism only models the set-element interactions (from past set to future element). The finer grain element-element (from past element to future element) is not explicitly modeled. In this section, we introduce a mechanism to model repeated-element-specified element-element relation.

There are usually some common regular tests and procedures repeatedly applied to diabetic patients (i.e. checking blood pressure). People usually have repeat purchase in shopping [1]. Our analysis on all data sets shows that 15%-60% of the elements in the future sets have appeared in the past sets. We also calculate the probability of each item appearing in the past to appear in the subsequent sets and observe that the elements frequently appearing in the past have high probability to appear in the future. So we propose to integrate a component to model this repeated pattern. We revise the equation 2 as follows:

$$\mathbf{o}(\hat{\mathbf{v}}_i) = \text{softmax}(W_s s_i \circ (\mathbf{1} - \beta \circ \alpha) + \gamma \circ \alpha), \tag{3}$$

where $\circ$ is the element-wise product. Each entry of the vector γ is the probability of the corresponding element appearing in the past sets of a given person. Vector β is a k-hot vector that records all the non-zero entries of the γ. It is used to control that the elements which have appeared in the past can get the contributions from our encoder-decoder and repeated pattern adaptively while the elements not appearing in the past are only predicted by our encoder-decoder. Vector α is the coefficient that balances the contributions coming from our proposed encoder-decoder and the probability from the repeated element interaction. The α is calculated by:

$$\alpha = \text{sigmoid}(W_\gamma \gamma + \mathbf{b}),$$

where matrix W_γ and vector $\mathbf{b}$ are learned from the data.

5.4 Training and Inference

According to the problem definition, the prediction at each subsequent set can be seen as a multilabel classification. So we need to consider the characteristics of multilabel classification. First, multilabel classification can suffer from label imbalance problem [11], which means some labels appear in much more instances than other labels. Figure 3 shows the frequency distribution of the top 30 most frequent codes in our diabetic data set. It shows that the frequency across different labels are imbalance. Similar phenomenons

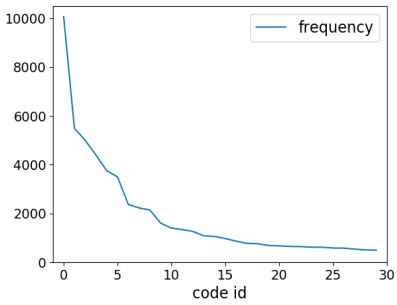


Figure 3: Frequency distribution of the top 30 most frequent codes.

can be observed in other data sets. Thus, we need to resolve label imbalance.

Second, the labels in multilabel classification has some patterns that can be exploited to improve the performance. Zhang et. al [32] show that the correlation among labels can be captured by forcing the predicted probabilities for the labels belonging to an instance to be higher than the ones not belonging to the instance.

Here, we use an objective function shown in the Equation 4 to consider both characteristics,

$$\arg\min_\theta \sum_Y \sum_{i=1}^{l} \text{WMSE}(\mathbf{v}_i, \mathbf{o}(\hat{\mathbf{v}}_i)) + \lambda \, \text{PSM}(\mathbf{v}_i, \mathbf{o}(\hat{\mathbf{v}}_i)), \tag{4}$$

where the accumulated sum outside goes through all the output sequential sets of different persons in the training data. The $\mathbf{v}_i$ is the ground truth and $\mathbf{o}(\hat{\mathbf{v}}_i)$ is the output from decoder for the i-th set. WMSE is *weighted mean square loss* and PSM is *partitioned set margin* constraint. The λ is a hyperparameter. The $\text{WMSE}(\mathbf{v}_i, \mathbf{o}(\hat{\mathbf{v}}_i))$ is calculated by:

$$\sum_m w(m)(c_m - \mathbf{o}(\hat{c}_m))^2, \tag{5}$$

where c_m is the m-th entry of $\mathbf{v}_i$ and $\mathbf{o}(\hat{c}_m)$ is the m-th entry of $\mathbf{o}(\hat{v}_i)$. The $w(m)$ is a weighted function for element m that is calculated by

$$w(m) = \frac{\max_n freq(n)}{freq(m)},$$

in which $freq(m)$ is the frequency of the element m in the training set. $w(m)$ is used to balance the contributions of different elements to the loss. Other loss functions like cross entropy, binary cross entropy and so on are also considered but cannot show better performance. The $\text{PSM}(\mathbf{v}_i, \mathbf{o}(\hat{\mathbf{v}}_i))$ is calculated by:

$$\frac{1}{|\mathbf{P}_{\mathbf{v}_i}||\overline{\mathbf{P}}_{\mathbf{v}_i}|} \sum_{(k,l) \in \mathbf{P}_{\mathbf{v}_i} \times \overline{\mathbf{P}}_{\mathbf{v}_i}} \text{marg}(\mathbf{o}(\hat{c}_k^i), \mathbf{o}(\hat{c}_l^i)), \tag{6}$$

where $\mathbf{P}_{v_i}$ is the positive set that contains all the elements appearing in v_i, $\overline{\mathbf{P}}_{v_i}$ is the negative set that contains all the elements not appearing in v_i and the pair-wise margin $\text{marg}(\mathbf{o}(\hat{c}_k^i), \mathbf{o}(\hat{c}_l^i))$ is calculated by $\exp(-(\mathbf{o}(\hat{c}_k^i) - \mathbf{o}(\hat{c}_l^i)))$. This constraint is to maximize the pair-wise margin between the predicted positive and negative sets.

Inference: We use a greedy algorithm to predict the elements with the top k highest probabilities at each subsequent set. We set the output vector's k entries that correspond to the top k highest

values in $\mathbf{o}(\hat{\mathbf{v}}_i)$ to 1 and the other entries to 0 as the prediction for each set. k and l are parameters given by users.

6 EXPERIMENTS

In this section, we conduct extensive experiments to answer the following research questions:

RQ1 How is the effectiveness of our designed framework? Can it provide better performance compared to the designed baselines and the tweaked state-of-the-art next set prediction methods?

RQ2 What kind of benefit the decoder RNN can bring? What are the problems in recurrently predicting the next set and directly predicting the subsequent sets?

RQ3 How does the repeated-element-specified element-element interaction contributes to the performance?

6.1 Experimental Settings

6.1.1 Data Sets. We experimente with three real-world data sets.

OPTUM. This data set from OPTUM consists of 4-year longitudinal medical claims data of diabetic patients. Medical claims data contains facility claims and physician claims that record all the services, the physicians' checking and diagnoses given to the patients. It also records all the procedures that the patients receive in the hospitals. All the events are encoded by standard codes including ICD code, Diagnosis Related Group code, and CPT/HCPCS Procedure code. We treat all the codes appearing in one visit to the medical facilities as a set.

Dunnhumby. This data set is opened sourced by dunnhumby, a global customer data science company. It contains the transactions about which items are bought by each customer in each order with the time stamp. We treat all the items bought in the same order as a set.

Ta-Feng. It is a public data set. It also contains the transactions about which items are bought by each customer in each basket with the time stamp. For simplicity, we only consider the top 5000 most frequent items that cover 83% items appearing in all the baskets. We treat all the items bought in the same order as a set.

The statistic information of all the data sets after pre-processing is shown in the Table 1.

Table 1: Statistic information after pre-processing.

Data	#elements	#sets	#persons	ave. set size	ave. #sets /person
OPTUM	4,226	100,682	7,280	2.67	13.87
Dunnhumby	4,997	289,918	36,241	7.33	7.99
Ta-Feng	5,000	66,714	7,384	5.35	9.03

6.1.2 Evaluation Measurement. We use **recall**, **NDCG** and **person-wise hit ratio (PHR)** to evaluate our methods. The prediction for each set can be seen as a multilabel classification. Recall is a wildly-used measurement for multi-label classification [33]. We use the average recall of all the predicted sets as the measurement. NDCG is a ranking based measure which takes into account the order of elements in a list [16]. We calculate the NDCG for each set based on the top k sorted elements list. The average NDCG of all the predicted sets is used as the measurement. Original hit ratio is used to measure the ratio of the users whose recommended items appear in the ground truth in the recommend system [10]. Since we have multiple sets in the predictions for each person, we define **person-wise hit ratio** (PHR) as *the ratio of the persons whose predicted sets all contain the elements appearing in the corresponding ground truth sets.* Unlike the recall focuses on the accuracy of the methods at set level, PHR focuses on the accuracy of the methods at person level.

6.1.3 Evaluation Method. In OPTUM data set, we have the records of patients across 4 years after they are diagnosed with diabetes. We partition the visits of each patient into to two parts. The visits in the first two years is used as history records to forecast the visits in the second two years. In Dunnhumby and Ta-Feng data sets, we partition the baskets of each user into two equal parts based on the time stamps. The first part with earlier time stamps is used as the history records to forecast the baskets in the second part. All the data sets are partitioned across persons. We reserve the data of 10% persons as the validation set for hyperparameters searching in all the methods. The left data is applied 5-fold cross-validation across persons to evaluate the methods.

6.1.4 Compared Methods. We compare our method with following methods.

Top-k frequent (TopKFreq): It uses the most frequent k elements that appear in all the sets of the training data as the prediction for all the subsequent sets.

Personalized Top-k frequent (PersonTopKFreq): It uses the most frequent k elements that appear in the past sets of a given person as the prediction for all the subsequent sets.

PersonKNN: It is a k-nearest neighbors (k-NN) based baseline proposed by us. The basic idea is to predict each subsequent set by selecting the top n most frequent elements appearing in the corresponding subsequent sets of the k nearest persons. To calculate the similarity between persons, we propose a way to represent the past sets of each person as a feature vector. First, the past sets are represented as m-hot vectors. Then all the m-hot vectors are partitioned into 3 groups (long time ago group, short time ago group, and recent group) of equal size according to the temporal order. If the number of the sets cannot be equally partitioned, we have the priority, the recent group > short time ago group > long time ago group, to enlarge the size of the group. The vectors within each group are summed up to one vector as the group vector. After applying a time decay weight to each group vector, we sum up all the weighted group vectors as the feature vector for similarity calculation.

FPMC: A classical hybrid model for next basket recommendation based on markov chain and factorization method [22]. Both sequential behaviors and users' personal tastes are taken into account for prediction. Since our problem can be seen as an extension to next basket recommendation as mentioned in the related work, we tweak this next basket recommendation method as our compared method. As the FPMC is a first-order markov model, we recurrently applying this method to predict the next set to get all the subsequent sets.

Table 2: Comparison with different methods on OPTUM data set. The k is the number of elements predicted for each set.

methods			k = 20			k = 40			k = 60			k = 80
predict next 2 visits	**Recall**	**NDCG**	**PHR**	**Recall**	**NDCG**	**PHR**	**Recall**	**NDCG**	**PHR**	**Recall**	**NDCG**	**PHR**
TopKFreq	0.3307	0.1786	0.5578	0.3870	0.1369	0.6688	0.4141	0.1431	0.7009	0.4313	0.1432	0.7277
PersonTopKFreq	0.2813	0.2346	0.3809	0.2836	0.2354	0.3857	0.2846	0.2356	0.3864	0.2855	0.2358	0.3877
PersonKNN	**0.3778**	**0.3097**	0.6094	0.4147	**0.3198**	0.6561	**0.4472**	**0.3273**	0.7035	**0.4765**	**0.3336**	0.7426
FPMC	0.3646	0.2709	**0.6266**	**0.4232**	0.2872	**0.7114**	0.4428	0.2944	**0.7740**	0.4601	0.2984	**0.8180**
DREAM	0.3472	0.2105	0.5552	0.4191	0.2303	0.6774	0.4329	0.2401	0.7323	0.4575	0.2479	0.7680
Sets2Sets	0.4050	0.3224	0.6616	0.4425	0.3318	0.7378	0.4682	0.3385	0.7906	0.4908	0.3439	0.8496
Improvement	7.2%	4.0%	5.9%	4.6%	3.7%	3.7%	4.7%	3.4%	2.1%	3.0%	3.1%	3.9%
predict next 3 visits	**Recall**	**NDCG**	**PHR**	**Recall**	**NDCG**	**PHR**	**Recall**	**NDCG**	**PHR**	**Recall**	**NDCG**	**PHR**
TopKFreq	0.2904	0.1447	0.4402	0.3338	0.1320	0.5489	0.3644	0.1343	0.5651	0.4083	0.1357	0.6122
PersonTopKFreq	0.2820	0.2400	0.2953	0.2858	0.2412	0.3046	0.2869	0.2414	0.3046	0.2880	0.2417	0.3046
PersonKNN	**0.3501**	**0.2816**	0.4241	**0.3917**	**0.2929**	0.4778	0.4341	**0.3028**	0.5583	**0.4686**	**0.3100**	0.6080
FPMC	0.3292	0.2263	**0.4268**	0.3778	0.2453	**0.6067**	**0.4360**	0.2546	**0.6714**	0.4627	0.2604	**0.7382**
DREAM	0.2891	0.1593	0.3181	0.3564	0.1835	0.4724	0.423 7	0.1975	0.5879	0.4605	0.2073	0.6617
Sets2Sets	0.3842	0.2934	0.5288	0.4299	0.3007	0.6335	0.4599	0.3062	0.7060	0.4822	0.3121	0.7704
Improvement	9.7%	4.1%	23.9%	9.7%	2.7%	4.4%	5.5%	1.1%	5.1%	2.7%	0.6%	3.2%

Table 3: Comparison with different methods on Dunnhumby data set. The k is the number of elements predicted for each set.

methods			k = 20			k = 40			k = 60			k = 80
predict next 2 baskets	**Recall**	**NDCG**	**PHR**	**Recall**	**NDCG**	**PHR**	**Recall**	**NDCG**	**PHR**	**Recall**	**NDCG**	**PHR**
TopKFreq	0.1158	0.0653	0.2602	0.1520	0.0676	0.3174	0.1791	0.0727	0.3623	0.2010	0.0734	0.3914
PersonTopKFreq	**0.2343**	**0.2182**	**0.4074**	**0.2940**	**0.2205**	**0.4673**	**0.3057**	**0.2312**	**0.4757**	**0.3100**	**0.2387**	**0.4791**
PersonKNN	0.1151	0.1171	0.2442	0.1291	0.1203	0.2687	0.1413	0.1240	0.2918	0.1533	0.1272	0.3131
FPMC	0.1106	0.0968	0.2479	0.1454	0.1082	0.3028	0.1753	0.1174	0.3491	0.2023	0.1248	0.3901
DREAM	0.1129	0.0955	0.2491	0.1478	0.1050	0.3279	0.1828	0.1274	0.3727	0.2173	0.1430	0.4130
Sets2Sets	0.2435	0.1701	0.4151	0.3167	0.1975	0.5062	0.3488	0.2091	0.5371	0.3645	0.2144	0.5538
Improvement	3.9%	-22.0%	1.9%	7.7%	-10.4%	8.3%	14.1%	-9.5%	12.9%	17.6%	-10.1%	15.6%
predict next 3 baskets	**Recall**	**NDCG**	**PHR**	**Recall**	**NDCG**	**PHR**	**Recall**	**NDCG**	**PHR**	**Recall**	**NDCG**	**PHR**
TopKFreq	0.1167	0.0655	0.1741	0.1517	0.0674	0.2218	0.1793	0.0727	0.2591	0.2007	0.0733	0.2861
PersonTopKFreq	**0.2323**	**0.2156**	**0.3201**	**0.2922**	**0.2300**	**0.3643**	**0.3042**	**0.2347**	**0.3725**	**0.3087**	**0.2364**	**0.3761**
PersonKNN	0.1153	0.1181	0.1593	0.1284	0.1210	0.1792	0.1409	0.1246	0.1957	0.1528	0.1279	0.2119
FPMC	0.1095	0.0964	0.1628	0.1440	0.1075	0.2083	0.1814	0.1186	0.2584	0.2034	0.1241	0.2864
DREAM	0.1102	0.0951	0.1598	0.1565	0.1113	0.2392	0.1855	0.1231	0.2757	0.2072	0.1325	0.3025
Sets2Sets	0.2560	0.1939	0.3315	0.3025	0.2112	0.3747	0.3195	0.2183	0.3912	0.3309	0.2227	0.4025
Improvement	10.2%	-10.0%	3.6%	3.5%	-8.1%	2.9%	5.0%	-6.9%	5.0%	7.2%	-5.8%	7.0%

DREAM: A state-of-the-art deep model based on RNN for next basket recommendation [31]. It considers personal dynamic interests at different time and the global interactions of all baskets of the user over time. To predict the subsequent sets, we consider both recurrently predicting next set and predicting a concatenation of all the subsequent sets that are widely used in time series forecasting [15]. We only report the results of the latter way as it achieves better performance.

We tune the hyper parameters in all the compared methods with grid search.

6.1.5 Configuration of Our Method. In our method, the RNNs are implemented by gated recurrent unit [8]. Adam [18] is used as the optimizer. The number of units in RNN and the dimension of embedding are both set to 32. The λ is set to 10.

6.2 Performance Comparison (RQ1)

The comparisons with different methods are shown in Table 2, Table 3, and Table 4. Several observations can be made. First, the simple TOPKFreq can achieve comparable recall and PHR compared to other model-based methods. It indicates that people have some common elements. But TOPKFreq achieves the worst NDCG as people still have distinct elements. TopKFreq is better than PersonTopKFreq in the OPTUM data set but is worse than PersonTopKFreq in other two shopping transactions data sets. The reason is that many regular tests and diagnoses are common across different patients as diabetes results in some shared pattern for different patients. But customers have distinct preferences when they purchase items.

Second, the personalized top-k frequent method outperforms other baselines in Dunnhumby and Ta-Fang data sets. It even achieves better NDCG than our proposed method in Dunnhumby

Table 4: Comparison with different methods on Ta-Feng data set. The k is the number of elements predicted for each set.

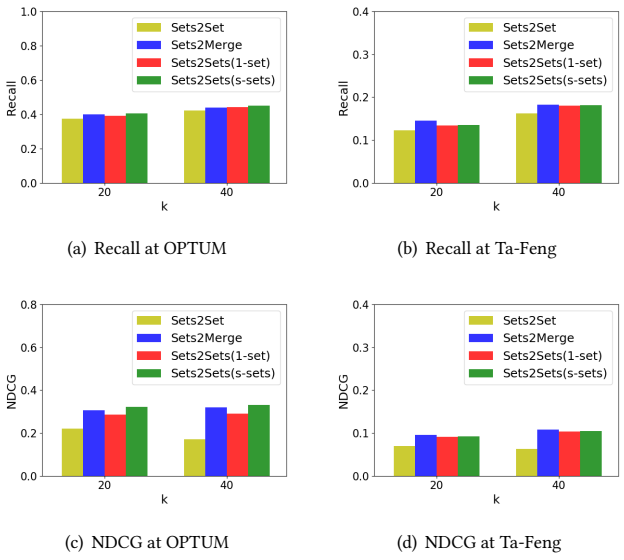
methods			k = 20			k = 40			k = 60			k = 80
predict next 2 baskets	**Recall**	**NDCG**	**PHR**	**Recall**	**NDCG**	**PHR**	**Recall**	**NDCG**	**PHR**	**Recall**	**NDCG**	**PHR**
TopKFreq	0.0905	0.0541	0.1111	0.1287	0.0505	0.1779	0.1606	0.0563	0.2433	**0.1876**	0.0642	0.2911
PersonTopKFreq	**0.1187**	**0.0859**	**0.1870**	**0.1464**	**0.0945**	**0.2419**	**0.1662**	**0.0998**	**0.2798**	0.1816	**0.1038**	**0.3122**
PersonKNN	0.0923	0.0675	0.0752	0.1095	0.0716	0.0970	0.1236	0.0763	0.1315	0.1384	0.0799	0.1582
FPMC	0.0866	0.0488	0.0893	0.1236	0.0606	0.1701	0.1505	0.0682	0.2271	0.1692	0.0732	0.2693
DREAM	0.0947	0.0538	0.0904	0.1245	0.0620	0.1743	0.1526	0.0706	0.2416	0.1758	0.0744	0.2780
Sets2Sets	0.1271	0.0890	0.2067	0.1739	0.1027	0.2876	0.2029	0.1107	0.3488	0.2263	0.1160	0.3909
Improvement	7.1%	3.6%	10.5%	18.8%	8.7%	18.9%	22.1%	10.9%	24.7%	20.6%	11.7%	25.2%
predict next 3 baskets	**Recall**	**NDCG**	**PHR**	**Recall**	**NDCG**	**PHR**	**Recall**	**NDCG**	**PHR**	**Recall**	**NDCG**	**PHR**
TopKFreq	0.0904	0.0428	0.0347	0.1259	0.0501	0.0765	0.1544	0.0507	0.1232	**0.1846**	0.0602	0.1600
PersonTopKFreq	**0.1163**	**0.0806**	**0.1073**	**0.1522**	**0.0989**	**0.1510**	**0.1669**	**0.1032**	**0.1660**	0.1796	**0.1066**	**0.1938**
PersonKNN	0.0812	0.0624	0.0268	0.1000	0.0668	0.0417	0.1146	0.0719	0.0516	0.1279	0.0754	0.0715
FPMC	0.0875	0.0536	0.0278	0.1211	0.0640	0.0725	0.1438	0.0709	0.1123	0.1627	0.0758	0.1332
DREAM	0.0926	0.0553	0.0284	0.1232	0.0648	0.0748	0.1470	0.0720	0.1263	0.1686	0.0773	0.1530
Sets2Sets	0.1216	0.0856	0.1123	0.1731	0.1005	0.1779	0.1946	0.1061	0.2097	0.2116	0.1097	0.2395
Improvement	4.5%	6.2%	4.6%	13.7%	1.6%	17.8%	16.6%	2.8%	26.3%	17.7%	2.9%	23.6%

data set. Our analysis shows that, in both customers' shopping orders data sets, different customers have some repeated purchases on different items. Especially, different customers in the Dunnhumby data set have their must-buy items in many of their orders. Personalized top-k frequent method can exactly capture this property and put the frequently bought items at high ranks. Thus, this baseline achieves high NDCG. It also indicates existing methods fail to effectively utilize this important property in the data. However, the OPTUM data set does not have this property as the disease progression results in the change of the codes over the time. So the personalized top-k frequent method is worse than other model-based baselines in OPTUM data set.

Third, our proposed PeronKNN is generally better than other baselines in OPTUM data with respect to recall and NDCG. We believe the reason is that the patients with similar historical records will have some common disease progression in the future which results in some shared codes appearing in the subsequent sets. But PeronKNN is still worse than PersonTopKFreq in two shopping transactions data sets as the items appearing in the future shopping transactions are distinct across different people.

Fourth, FPMC and DREAM can not outperform personalized top-k frequent method in Dunnhumby and Ta-Fang data sets. We believe that too many interactions between different elements in FPMC and DREAM make the effect of the important interactions between repeated items vague. FPMC achieves higher PHR than PersonKNN, which indicates the transition from the elements in the last set affecting the elements in the next set is personalized and FPMC can capture it accordingly while the elements from the nearest neighbors are sometimes different from the given person's elements. FPMC outperforms DREAM in OPTUM data set while DREAM outperforms FPMC in Dunnhumby and Ta-Fang data sets. We believe the reason is that the next set is largely affected by the previous set in OPTUM data. FPMC is first-order markov model so that it achieves better performance in this data.

Fifth, our Sets2Sets method consistently outperform the best performance of all the baselines with respect to recall and PHR

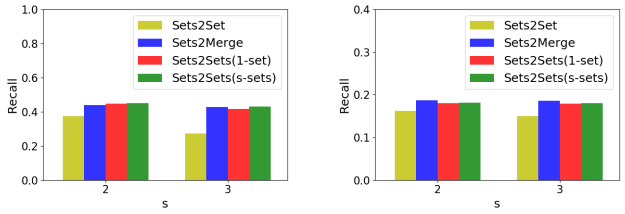


(a) Recall at OPTUM (b) Recall at Ta-Feng

(c) NDCG at OPTUM (d) NDCG at Ta-Feng

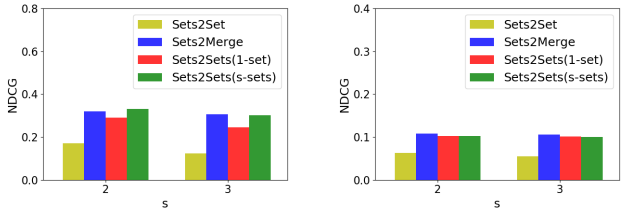
Figure 4: Comparison with different variants by predicting next 2 sets. The k is the number of elements predicted at each set.

by 2.7 − 20.6% and 2.1 − 26.3%, respectively. Our method also achieves better NDCG in OPTUM and Ta-Feng data sets. It indicates our method can address the issues mentioned in the baselines. Our encoder with embedding can model the personal tastes and the sequential behaviors. Our set attention mechanism can focus on the most related sets, which can adaptively handle the different properties of different data. Our repeated-element-specified element-element interaction component enhances the signal of the repeated elements.



(a) Recall at OPTUM

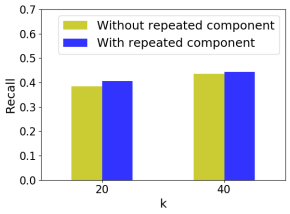
(b) Recall at Ta-Feng



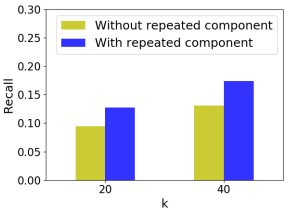
(c) NDCG at OPTUM

(d) NDCG at Ta-Feng

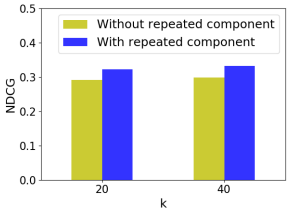
Figure 5: Comparison with different variants by predicting each set with 40 elements. The s is the number of sets predicted.



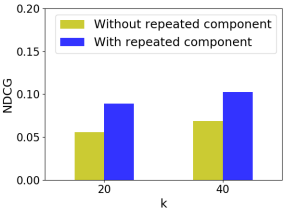
(a) Recall at OPTUM



(b) Recall at Ta-Feng



(c) NDCG at OPTUM



(d) NDCG at Ta-Feng

Figure 6: The effect of the repeated element component. We compare the methods by predicting next 2 sets. k is the number of elements predicted at each set.

6.3 Effect of the Decoder (RQ2)

In this section, we investigate if it is necessary to use our decoder to predict sequential sets. We compare our method with three variants of our method. The first two variants are based on the widely-used ideas in time series forecasting [15]. The first variant is to predict subsequent sequential sets by recurrently feeding both past sets and previous predicted set into the encoder to predict the next set. Denote it as Sets2Set. The second variant is to predict subsequent sequential sets by concatenating these sets into one large vector, which means we only have one RNN unit in the decoder to predict all the sets together. Denote it as Sets2Merge. The third variant is to limit the supervision with only the first subsequent set. No matter how many subsequent sets we predict in the test step, we ignore all the supervision information after the first subsequent set in the training. Denote it as Sets2Sets(1-set). Our method is denoted as Sets2Sets(s-sets), which means we use subsequent s sets as the supervision in the training step.

The experimental results are shown in the Figure 4 and 5 . We can observe that Sets2Sets(s-sets) outperforms Sets2Set. We believe there are two reasons. First, Sets2Sets can be seen as a multi-task learning that jointly predicts different sets in the outputs, which leverages the correlation among different sets in the outputs. Specifically, each set in the outputs is also associated with other sets after it during the training. But in the training of Sets2Set, each set is predicted without the information of the sets after them. Second, Sets2Set requires us to forward the predicted set as part of the inputs, which means the errors in the predictions enter into the inputs. Note that the average set size is less than 10 while the predicted set size are larger than this. Many fake elements is introduced into the inputs. It is supported by the observation that the NDCG decreases with the set size k increased. Also, we can observe that the performance decrease of Sets2Set is larger than other methods. It implies that the error is accumulated when we increase the number of subsequent sets s. We believe that tweaking the existing next basket recommendation methods in the similar recurrent way also suffers from the same problems.

Another observation is that Sets2Merge is competitive with Sets2Sets. By predicting the subsequent sets together, we also avoid to introduce errors into the input. The correlation existing in the subsequent sets can also be captured by our constraint in the loss function. It indicates our method can properly utilize the supervised information from the subsequent sets.

The superiority of our way to predict subsequent sets exists in predicting subsequent sets beyond the supervision. As the number of the subsequent sets is given by users, we may predict subsequent sets that are longer than any training instances. As Sets2Merge has a max number of sets to predict at one shot, we can use the similar idea in Sets2Set to recurrently predict the part beyond the max number. Actually, when $s > 1$, Sets2Set can be seen as using an enhanced version of Sets2Merge(1-set) to predict subsequent s sets with recurrent strategy. Sets2Set has more data than Sets2Merge(1-set) as Sets2Set contains the data with the supervision after the first subsequent set in the training. However, Sets2Sets(1-set) still achieves better performance than Sets2Set. Sets2Sets(1-set) even achieves comparable performance compared to Sets2Merge and Sets2Sets(s-set) with respect to the recall. Even though the NDCG of Sets2Sets(1-set) decreases with the increase of s, we believe it is expected as Sets2Sets(1-set) has less supervision information than other methods. The results support that our decoding way not only avoids introducing error into the input but also generalizes fairly to subsequent sets beyond the supervised part.

6.4 Effect of the Repeated Elements (RQ3)

In this section, we investigate how effective the repeated element component is. We compare our method with a variant without this component. The results are shown in the Figure 6. The component improves the performance in both data sets. In Ta-Feng data, our method gets improvement by significant margin. We believe that this component can properly capture the important property that many customers repeatedly buy some items and improve the performance largely. However, similar repeat pattern is not strong in OPTUM data as the codes change with the disease progression.

7 CONCLUSION AND FUTURE WORK

In this paper, we propose a sequential sets to sequential sets problem to model subsequent sets forecasting. Existing methods that only predict the sets in the next steps have limitations to utilize the information of elements existing within and across different sets. To address these issues, we develop an encoder-decoder framework. In future, we will consider to learn from sequential graph, which may bring benefit by introducing pre-existing relations between elements within each set [30].

With the emergence of many applications, the study towards mining the pattern in the new type of data — sequences of sets — starts to get attention recently [5]. Our work makes a step towards studying how to predict the new type of output — sequential sets. To our best knowledge, this paper is the first attempt to predict this new structured output. We envision that the proposed method will serve as a starting point for the prediction of the sequential sets. Code accompanying with this paper is available at Github.

Acknowledgement: This research was supported in part by NSF (1447788, 1704074, 1757916, 1834251), the Thousand Youth Talents Program 2018, Army Research Office (W911NF1810344), Intel Corp, and the Digital Technology Center (DTC) at the University of Minnesota. Access to research and computing facilities was provided by the DTC and the Minnesota Supercomputing Institute.

REFERENCES

[1] Alhassan G Abdul-Muhmin. 2010. Repeat purchase intentions in online shopping: The role of satisfaction, attitude, and online retailers' performance. *Journal of International Consumer Marketing* 23, 1 (2010), 5–20.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[3] Tian Bai, Shanshan Zhang, Brian L Egleston, and Slobodan Vucetic. 2018. Interpretable Representation Learning for Healthcare via Capturing Disease Progression through Time. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 43–51.

[4] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*. 1171–1179.

[5] Austin R. Benson, Ravi Kumar, and Andrew Tomkins. 2018. Sequences of Sets. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1148–1157.

[6] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. 2015. *Time series analysis: forecasting and control*. John Wiley & Sons.

[7] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 335–344.

[8] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259* (2014).

[9] Edward Choi, Mohammad Taha Bahadori, Andy Schuetz, Walter F Stewart, and Jimeng Sun. 2016. Doctor ai: Predicting clinical events via recurrent neural networks. In *Machine Learning for Healthcare Conference*. 301–318.

[10] Evangelia Christakopoulou and George Karypis. 2018. Local latent space models for top-n recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1235–1243.

[11] Zachary Alan Daniels and Dimitris N Metaxas. 2017. Addressing Imbalance in Multi-Label Classification Using Structured Hellinger Forests.. In *AAAI*. 1826–1832.

[12] Manaal Faruqui, Yulia Tsvetkov, Pushpendre Rastogi, and Chris Dyer. 2016. Problems with evaluation of word embeddings using word similarity tasks. *arXiv preprint arXiv:1605.02276* (2016).

[13] Valentin Flunkert, David Salinas, and Jan Gasthaus. 2017. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *arXiv preprint arXiv:1704.04110* (2017).

[14] Nadia Ghamrawi and Andrew McCallum. 2005. Collective multi-label classification. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. ACM, 195–200.

[15] Coşkun Hamzaçebi, Diyar Akay, and Fevzi Kutay. 2009. Comparison of direct and iterative artificial neural network forecast approaches in multi-periodic time series forecasting. *Expert Systems with Applications* 36, 2 (2009), 3839–3844.

[16] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. 2015. Trirank: Review-aware explainable recommendation by modeling aspects. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 1661–1670.

[17] Xiangnan He, Zhenkui He, Jingkuan Song, Zhenguang Liu, Yu-Gang Jiang, and Tat-Seng Chua. 2018. NAIS: Neural Attentive Item Similarity Model for Recommendation. *IEEE Transactions on Knowledge and Data Engineering* (2018).

[18] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[19] William Leigh, Russell Purvis, and James M Ragusa. 2002. Forecasting the NYSE composite index with technical analysis, pattern recognizer, neural network, and genetic algorithm: a case study in romantic decision support. *Decision support systems* 32, 4 (2002), 361–377.

[20] Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. 2014. Recurrent models of visual attention. In *Advances in neural information processing systems*. 2204–2212.

[21] Sara Morsy and George Karypis. 2017. Cumulative knowledge-based regression models for next-term grade prediction. In *Proceedings of the 2017 SIAM International Conference on Data Mining*. SIAM, 552–560.

[22] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*. ACM, 811–820.

[23] CM Ryan, TM Williams, DN Finegold, and TJ Orchard. 1993. Cognitive dysfunction in adults with type 1 (insulin-dependent) diabetes mellitus of long duration: effects of recurrent hypoglycaemia and other chronic complications. *Diabetologia* 36, 4 (1993), 329–334.

[24] Hagen Soltau, Hank Liao, and Hasim Sak. 2016. Neural speech recognizer: Acoustic-to-word LSTM model for large vocabulary speech recognition. *arXiv preprint arXiv:1610.09975* (2016).

[25] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.

[26] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. 2005. *Introduction to Data Mining*.

[27] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[28] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*. IEEE, 3156–3164.

[29] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2015. Learning hierarchical representation model for nextbasket recommendation. In *Proceedings of the 38th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 403–412.

[30] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *Proceedings of the 42th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM.

[31] Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. A dynamic recurrent model for next basket recommendation. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 729–732.

[32] Min-Ling Zhang and Zhi-Hua Zhou. 2006. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE transactions on Knowledge and Data Engineering* 18, 10 (2006), 1338–1351.

[33] Min-Ling Zhang and Zhi-Hua Zhou. 2014. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering* 26, 8 (2014), 1819–1837.