"Don't forget the milk again!"; Predicting temporal shopping sets using GNNs

Sara Bizjak
sb6054@student.uni-lj.si

Maruša Oražem
mo3757@student.uni-lj.si

Vid Stropnik
vs2658@student.uni-lj.si

1 Introduction

Predicting future dynamics of real world processes has an inherent value, which is arguably clearer than that of many other machine learning tasks. Time series prediction has seen substantial application in domains such as finance, meteorology, transportation, HR allocation, quality control, etc. While most proposed solutions consider temporal data with corresponding numeric values, there is great value to be had when also considering nominal events with respect to the time of their sampling. These nominal *temporal sets* are very pervasive in real world scenarios and require models, attempting their prediction, to also capture the semantic relationships of individual sets' items to reach their full expressive potential. As part of our course project, we aim to reproduce the findings of a recent approach by Yu et. al.[1] for temporal set prediction using graph neural networks, and explain the model's inner workings on the example of shopping cart prediction.

2 Application domains

To showcase the performance of our temporal set prediction model, we will use *Dunnhumby's* The Complete Journey dataset [2], containing $\sim$ 1M household level transactions over two years from a group of 2,500 households who are frequent shoppers at a retailer. The dataset will be preprocessed in such a way that items bought by the same household at the same time unit will be grouped into (temporal) sets. Should the dataset prove as not rich enough for our intents, the authors of the resource also offer a synthetic dataset with $\sim$ 300M transactions, which captures real patterns and correlations, so it (or a subset thereof) could be used as well. We aim to solve tasks following the procedure described in [1], and will thus attempt to predict the shopping basket contents (purchased items by a household) at a given timestamp, S_i^t by considering all of it's previous transactions; more formally $S_i^t = f(S_i^1, S_i^2, \ldots S_i^{t-1}, \mathbf{W})$, where $\mathbf{W}$ represents the trainable parameters.

Our main performance measure will be the predictions recall, as it is a widely known and interpretable metric, often considered for these sorts of tasks. For better understanding, we also plan to consider other measures, proposed in [1], such as the Normalized Discounted Cumulative Gain and Personal Hit Ratio. For all considered metrics, we will weigh the ground truths against a list of *top K* elements, output from our model, for different values of K. Our intuition for these choices is the wish to compare our models performance with those, reported in [1]. However, we also plan to attempt to complement the original done work with our own experiments. Our current ideas include the evaluation of a rolling train/test split, as well as predicting with ensambles of the GNN approach and top performing naïve benchmarks.

The reasons for the choice of dataset are manyfold. Among the datasets used in the original work, it has the best ratio of size and performance, which will enable us to work efficiently and produce results which will be interpretable. Furthermore, the dataset also contains a very rich collection of metadata (eg. household demography information including age, gender, # of children and income level) will allow us to perform interesting analyses and create engaging examples for our final blog post.

3 Methodology and techniques

We aim to achieve our prediction with a GNN, consisting of three main modules. Their roles briefly boil down to learning the mutual similarities of observed goods, learning the sets temporal dependencies and then updating the representations through gated fusion. The overall proposed framework is visualized in Figure 1 and is explained further below.

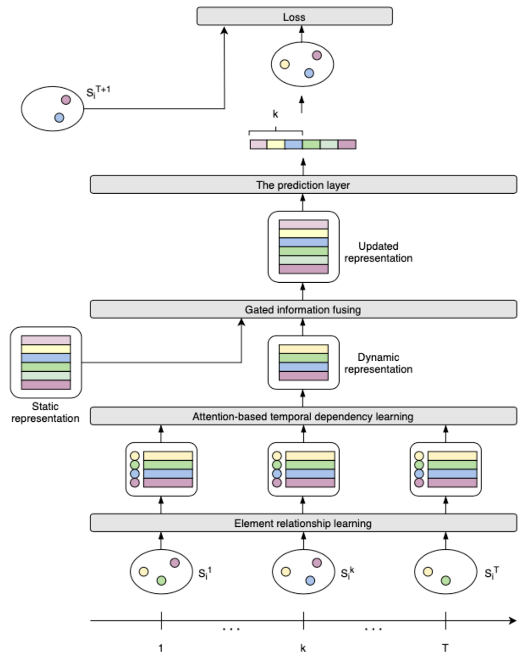

Figure 1: Proposed framework.

Element relationship learning. First we construct a weighted graph in a way, that will try to keep the useful information we can get in the set-level relationships. We denote the a household individual temporal shopping cart sets of household u_i, as $S_i^t \in \mathbb{S}_i$, where $t \in T$ represents the time index. For each such set, we extract unique pairs of its elements. We then assign a normalized frequency f to each such pair, according to its occurence accross $\mathbb{S}_i$. From these *(pair, frequency)* tuples, we construct belonging graphs, with f as the edge weights. For each household, we thus construct $|T|$ weighted, undirected graphs $\mathcal{G}_i^t = (\mathcal{V}_i, \mathcal{E}_i^t) \in \mathcal{G}_i$, where $\mathcal{V}_i$ is the set of all vertices or elements of $\mathbb{S}_i$, and $\mathcal{E}_i^t$ the set of edges between them in time unit t. By inputting $\mathcal{G}_i$ into a convolutional module, we gain a new sequence of element representations $\mathbb{C}_{i,j}$, with one representation $c_{i,j}^t$ per time unit. A parameter sharing strategy is adopted for the sharing of parameters between different timesteps in the convolutional layer.

Attention-based temporal dependency learning. The second component is proposed to learn the mutual dynamics of these sequences and their development over time, creating the temporal-information laden shared representation $\mathbb{Z}_i$, from the input sequence $\mathbb{C}_{i,j}$, as shown on Figure 1. More formally, the new F-dimensional representations considering temporal dependency are computed as follows,

$$Z_{i,j} = softmax\left(\frac{(C_{i,j}\mathbf{W}_q) \cdot (C_{i,j}\mathbf{W}_k)^T}{\sqrt{F}} + \mathbf{M}_i\right) \cdot (C_{i,j}\mathbf{W}_v), \tag{1}$$

where $\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v$ are trainable matrices, and $\mathbf{M}_i$ is a matrix that takes into account only knowledge from previous states. Further, $Z_{i,j}$ is aggregated by the following weighted aggregation,

$$z_{i,j} = \left((Z_{i,j} \cdot \mathbf{w}_{agg})^T \cdot Z_{i,j}\right)^T, \tag{2}$$

where $\mathbf{w}_{agg}$ is a trainable parameter.

Gated information fusing. Finally, we combine the dynamic embedding matrix given as the output of the previous two components with the static-information embedding matrix, shared between all households u_i. The representations of items appearing in $\mathcal{V}_i$ are updated using both static and dynamic information as follows,

$$\mathbf{E}_{i,I(j)}^{update} = \left(1 - \beta_{i,I(j)} \cdot \gamma_{I(j)}\right) \cdot \mathbf{E}_{i,I(j)} + \left(\beta_{i,I(j)} \cdot \gamma_{I(j)}\right) \cdot z_{i,j}, \tag{3}$$

while for the other elements, their original static representations are preserved. In Equation 3, the function $I(\cdot)$ maps item $v_{i,j}$ to its corresponding index in $\mathbf{E}_i$. $\beta_{i,j}$ and γ_j represent the j-th dimension of β_i and γ, where β_i is a vector which indicates the existence of corresponding element in $\mathcal{V}_i$ and γa vector that controls the importance of the static and dynamic information.

The prediction layer and learning process. At the final layer, we use E^{update} and trainable parameters b_0, w_0 for final predictions of probabilities that each element might appear in the next set.

$$\hat{\mathbf{y}}_\mathbf{i} = sigmoid(E_i^{update} \cdot w_0 + b_0). \tag{4}$$

From there, we simply select the top K probabilities of $\hat{y}_i$ as the final predicted set.

References

[1] Yu, L. et al. (2020) 'Predicting Temporal Sets with Deep Neural Networks', in Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery Data Mining. KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event CA USA: ACM, pp. 1083–1091. doi:10.1145/3394486.3403152.

[2] Source Files - dunnhumby. Available at: www.dunnhumby.com/source-files/ (Accessed: 20 October 2021).