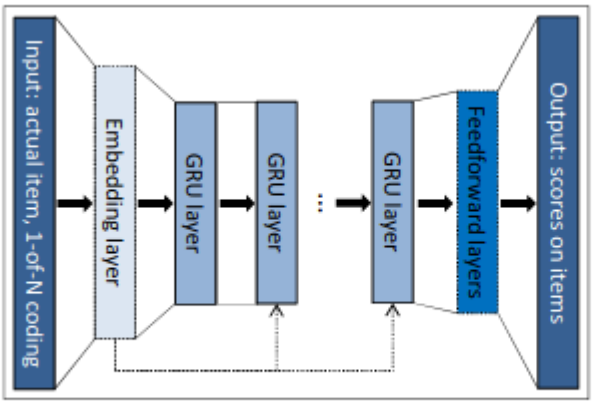
I presented the top 10 recommendations for the task given to me, with two different recommendation methods, session-based and content-based. **I used two systems to increase the variety in the recommendations I returned.**

Session-based approach:

This approach is a GRU-based neural network model. It tries to predict the next product to be purchased according to the products purchased earlier in the session. Therefore,it brings a point of view that; Users who bought these products also bought these products.

It takes the 1-to-N encodings of the products as input and returns the probability of each product being the next product to be bought.



General architecture of the network. Processing of one event of the event stream at once.

While evaluating this model, the recall@10 metric is used. In this metric, the number of instances where the product actually received at the next time, among the 10 products that the model finds with the highest probability, is divided by the total number of instances.

While creating train and test data for this model, a certain time point was selected, distances with event time before this time point were determined as train data, and instances after this time point were determined as test data. Another operation performed outside of this split is the cleaning of test instances with products that are not in the train data. Therefore, this model can only produce predictions for products in the train data. This is a negative feature of this model. In real life, there may be products that are very similar to those that users bought, but users may not buy them. Unfortunately, this model cannot handle these situations.

The recall@10 score value obtained from the test data created as mentioned above is **0.22**. It may seem a little low, but the model was trained only 10 epochs due to time and memory constraints.

I think this score can increase even more as a result of a good parameter optimization. Metadata is not used in this approach. If a way is found to include metadata in this model, recall results may increase a little more.

Content-based approach:

In this approach, the features of each product are extracted using meta data. Using these features, the cosine similarity matrix of the products is calculated. This similarity matrix is used for recommendation.

The features of the products are extracted as follows; **1)** Using the NLTK library, the key words of the name column in the metadata are extracted. **2)** By combining the keyword column with the brand, category and subcategory columns, unique bag of words are extracted for each product. **3)** The average vector of the word2vec vectors of the bag of words for each product is determined as product's feature. **(Experiments were also made with count vector and tf-idf vectors, but the features created with these methods were extremely sparse and long. For this reason, word2vec embeddings are used.)**

When making a recommendation for a session, the 10 products with the highest similarity scores in the similarity vectors of the products received so far in that session are recommended. (of course I excluded the products themselves)

This method is very different from the session-based method. Test data of the session-based model was used to test how successful this method would be. This evaluation looked at whether the latest products taken in each session were recommended. For this, as described above, for each session, 10 products with the highest similarity scores in the similarity vectors of the received products are recommended. It is checked whether the last product received in the session is among the recommendations. The number of products both recommended and received is divided by the length of the data.

The accuracy value I got at the end of this calculation is **0.0194**. Although this accuray seems very low; **1)** The evaluation method I use is entirely my own and needs to be improved. (Unfortunately, I could not find any examples of how content-based recommenders were evaluated in my literature search.) **2)** I think that more generalizable and robust representations can be made by using doc2vec instead of word2vec.

Preprocessing:

1) Instances containing NaN in both metadata and event data have been cleaned.
2) Categories with very few unique products have been removed from the metadata. (Like Fruit and Vegetable category)
3) If the event data contains product IDs that are not included in the metadata, these instances are cleared from the event data.
4) Sessions that bought only one product are cleared from the event data.
5) Instances with products with less than 5 purchases cleaned from the event data.

An Example for API: