Fashion recommender systems with focus on time and seasonability

Jaime Ferrando Huertas

September 22, 2021

Overview

Introduction

- Master Thesis collaboration with H&M.
- 40% increase in online revenue from 2019 to 2020, with a total of 5.1 Billion euros. Highly susceptible for improvements on recommender systems
- Fashion industry - different scenario as seen in state of the art recommender systems.

Introduction - Research question

- Does treating our user's history as an ordered sequenced of events improve our recommendation models?

Introduction - State of the art

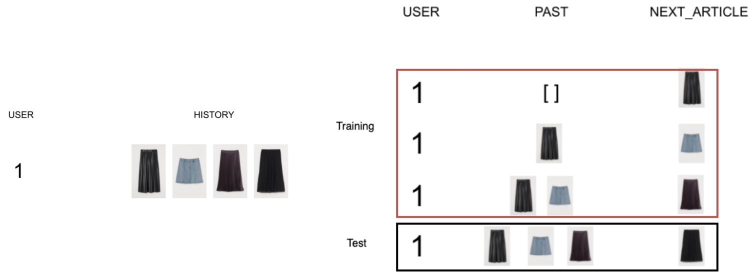
- Usage of Deep Learning methods across big tech companies. Models very closed to what is used in NLP.

Dataset

- User interactions in H&M website. Clicks, views.
- Swedish Market
- 1 month data, 2021-03-01 to 2021-04-01.
- Removed consecutive duplicates.
- Last user interactions as test set.

Dataset - Preprocessing

Given a user history H containing $h_1, h_2, ..h_n$ interactions, a training sample can be created from each interaction, h_t where the past interactions $h_1, .., h_t - 1$ are used as past history to predict interaction h_t



Given H the model will produce the conditional probability of the next item the user will interact across all different item classes. The ordered list of probabilities will represent the recommendation output from the model.

Evaluation

Classification Performance

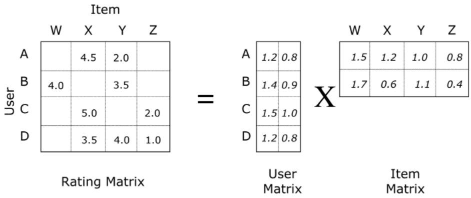
- Hit Ratio at K - Is the next interaction in the user recommendations set.

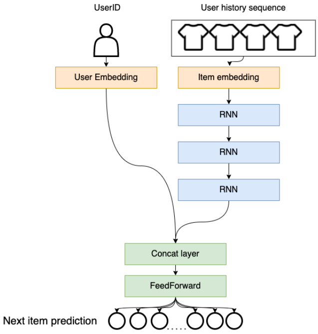
Diversity

- Coverage at K- % of total items included in the recommendation set
- Overlap at K- % of user recommendations sets that are equal
- Personalization at K - Measures how unique are each user recommendation set.
- Novelty at K - Measures novelty of recommended items.

At K: Measure within the top K recommendations. We will use 14.

Models





- AALS - Aproximate Alternative Least Squares
- Neural Sequencer - Tried RNN, LSTM, GRU, Attention, Transformers

Experiments - Models performance

Experiments	*HR @14*	*Coverage @14*	*Overlap @14*	*Personalization @14*	*Novelty @14*
AALS	4.42%	0.24%	**14.45%**	56.81%	7.6
N.S LSTM	**50.32%**	**8.75%**	21.54%	**90.57%**	**9.88**
N.S RNN	43.94%	7.68%	19.84%	89.87%	9.51
N.S GRU	49.00%	8.04%	25.94%	88.03%	9.45
N.S Attention	47.46%	7.61%	19.84%	89.55%	9.44
N.S Transformer	44.33%	7.67%	21.97%	86.98%	9.36

Table: Model results overview

Experiments - Embeddings

User Size	*Item Size*	*HR @14*	*Coverage @14*	*Overlap @14*	*Personalization @14*	*Novelty @14*
1	256	49.35%	8.67%	23.88%	89.84%	9.87
256	256	**50.32%**	**8.75%**	**21.54%**	**90.57%**	**9.88**

Table: User embedding size study on N.S LSTM

User Size	*Item Size*	*HR @14*	*Coverage @14*	*Overlap @14*	*Personalization @14*	*Novelty @14*
256	1	35.54%	6.20%	22.80%	89.40%	9.8
256	256	**50.32%**	**8.75%**	**21.54%**	**90.57%**	**9.88**

Table: Item embedding size study on N.S LSTM

Experiments - Ordering

User history Sequence order	*HR @14*	*Coverage @14*	*Overlap @14*	*Personalization @14*	*Novelty @14*
Correct order	**50.32%**	**8.75%**	**21.54%**	**90.57%**	**9.88**
Day level order	49.37%	8.04%	23.39%	90.46%	9.55
Random	43.37%	7.73%	22.89%	85.41%	8.89

Table: User history sequence order study on N.S LSTM

Experiments - History

User history Sequence order	*HR @14*	*Coverage @14*	*Overlap @14*	*Personalization @14*	*Novelty @14*
5	45.00%	**8.77%**	25.19%	92.05%	9.41
10	48.00%	8.61%	26.60%	89.50%	9.25
Default value - 20	50.32%	8.75%	21.54%	90.57%	**9.88**
50	**50.35%**	7.78%	**16.76%**	**94.60%**	9.69

Table: User history sequence length study on N.S LSTM

Conclusions

- Neural models beat our baseline matrix factorization methods.
- User and item embeddings both play different roles and affect specific metrics.
- It is not only the raw neural based models performance, but treating the user history as an ordered sequence that reports best results.
- The more history of the user we include, the better.
- Unable to follow NLP research path.

Future work

- Recommendations longevity - Study model's performance over time, how quick it degrades.
- Transfer learning - Reuse embeddings and previous models.
- Split representation and recommendation models - Train each one separately first and then join them.
- Usage of item/user features with embeddings - Will help reduce cold starts.

Questions