Natural language processing in context of recommender systems

Andre Soblechero Salvado

[1] Hamburg University of Applied Sciences, Berliner Tor 5, 20099 Hamburg, Germany
[2] Department of Computer Science

Abstract. In this paper fundamental techniques for recommender systems are explained and several techniques to handle natural language with the objective to combine both in future projects. Starting with content based and collaborative based filtering [6] to LASER [7] and Transformers [9].

Keywords: recommender · system · content based filtering · collaborative filtering · LASER · nlp · Transformers

1 Introduction

In this paper fundamental techniques for recommender systems are explained and several techniques to handle natural language with the objective to combine both in future projects. Starting with content based and collaborative based filtering [6] to LASER [7] and Transformers [9].

2 Recommender system

Nowadays people are confronted with an constant flood of information. Many people are working on a solutiuon to provide user-specific filters against the flood of information. These filters can also be called recommendations thus users get more personalized information shown. Especially plattforms based on advertising benefit from recommender systems because users are more likely to click on personalized advertising what could imply that these companys earn more money.

The definition of recommender systems "In a typical recommender system people provide recommendations as inputs, which the system then aggregates and directs to appropriate recipients. In some cases the primary transformation is in the aggregation; in others the system's value lies in its ability to **make good matches between the recommenders and those seeking recommendations.**" [5] seems to consider nearly every aspect.

There are plenty of different techniques to archive a good recommender system but the most common ones are content based and collaborative filtering.

2.1 Content Based Filtering

Content based filtering [6] describes the approach to build a recommender system based solely on features of users and items which we want to recommend. User-features like residence, age, user behaviour and item-features like language, topics are eg. feed into an model which predicts the next recommendation.
There are several approaches to this technique. One could build a model which takes the user-, item-features and predicts wheter the item fit the needs of an user or not. Furthermore a model could be build which only takes the user-features and predicts the best recommendation out of every item directly.

The advantage of content based filtering is that it does not suffer that much from the cold start problem [4] like other techniques. The cold start problem is the name of the problem that some recommender systems are not able to recommend items to users because of an lack on information about the user eg. reading history or up or down votes of movies. These informations are not avaiable for new users but the residence and thus the language as well as the age. Additionaly the varianz of the recommendations are not that high.

On the other hand not always all features of a user are avaiable eg. many people do not like to give informations about their age. A big problem is too that these kind of filters are very biased because the user-feature can be superficial. This means eg. that users from germany, which are more likely to speak german, would only get recommendations for german movies although they prefer to view movies from hollywood.

2.2 Collaborative Filtering

Collaborative filtering [6] is a technique which recommend items to a user based on all interactions between all users und items.
All interactions betweens all users und items are stored in a so called user-item interaction matrix M^{uxi} where u is the amount of users, i the amount of items and every entry of this matrix M_{ij} is a interaction between user i and item j. A interaction could be eg. a rating between 0 and 10 of the movie j by user i.
This matrix is than used for memory based or model based collaborative filtering.

$$\begin{array}{cc} & \text{product rated by user} \\ \text{user id} & \begin{pmatrix} 1.0 & 0 & 5.0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3.0 & 0 & 0 & 0 & 0 & 11.0 & 0 \\ 0 & 0 & 0 & 0 & 9.0 & 0 & 0 & 0 \\ 0 & 0 & 6.0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 7.0 & 0 & 0 & 0 & 0 \\ 2.0 & 0 & 0 & 0 & 0 & 10.0 & 0 & 0 \\ 0 & 0 & 0 & 8.0 & 0 & 0 & 0 & 0 \\ 0 & 4.0 & 0 & 0 & 0 & 0 & 0 & 12.0 \end{pmatrix} \end{array}$$

Fig. 1. Example of a user-item interaction matrix [13]

Memory Based The memory based approach applies algorithms to the user-item interaction matrix to generate recommendations.
There are two subcategories in memory based filtering.
The first one focuses on finding users which had similar interactions with each item. The second one focuses on finding items which had similar interactions per user.
The algorithms are eg. k-nearest neighbors or Approximate nearest neighbor. Nearly all algorithms are based on similarity of vectors, in particular the cosine similarity. [3]

$$sim(\overrightarrow{a}, \overrightarrow{b}) = \frac{\overrightarrow{a}\overrightarrow{b}}{\|\overrightarrow{a}\|\|\overrightarrow{b}\|} \tag{1}$$

Since every row of the matrix M represents the interactions of a user these rows can be seen as vectors.

The user-user technique tries to find similar rows of the matrix M, thus similar interacting users. This can be interpreted as a way of finding user with similar taste and behavior. When a similar user $\overrightarrow{b}$ is found then the actual user $\overrightarrow{a}$ gets a recommendation for an item M_{ij} which the user $\overrightarrow{a}$ has not seen yet but the user $\overrightarrow{b}$ interacted positive with.

The advantages of user-user are very personalized recommendations. On the other hand this technique scales poor considering that todays plattforms can have millions of users and items. Futhermore the varianz is very high eg. after some new interactions M_{ij} the algorithms have to recompute similar users. Also the cold start problem will appear because users with less to no interactions would not have meaningful similar users.

The item-item technique tries to find similar columns of the matrix M with the same algorithms. This can been seen as a approach of eg. finding similar movies which have been rated by most users the same way.

A clear advantage is that the varianz is not that high because eg. bad movies are generally rated as bad by most people and later nothing would change. In contrast this technique scales poor too. Also is this technique not that personilized like user-user and could suffer from the could start problem. A new problem appearing with item-item is the rich-get-richer effect [2]. This effect discribes the problem of users only getting recommendations for eg. already good rated movies what leads to less recommendations of niche movies.

Model Based The model based approach of collaborative filtering tries to explain the user-item interaction matrix M by learning user and item representations which can be used to reconstruct the user-item interaction matrix M. This objective can be accomplished using matrix factorization.

Having a user-factor matrix U^{uxd} where each row represents a user and a factor-item matrix I^{dxi} where each colomn represents a item we get a reconstruction matrix R^{uxd} by multiplying U and I. This shows us that the reconstruction matrix has the same dimensions as the user-item interaction matrix.
The objective is to minimize the error between reconstruction matrix and user-item interaction matrix what implies minimizing the error between user-item interaction matrix and the dotproduct of U and I.

Using the mean squared error it is possible to write down this objective formally:

$$min\frac{1}{n}\sum_{(ij)\epsilon M}((U_i)(I_j)^T - M_{ij})^2 \tag{2}$$

To minimize the error several procedure can be used. The gradient method is a common procedure to minimize this error.

$$R_{ij} = R_{ij} - \eta\Delta(\frac{1}{n}\sum_{(ij)\epsilon M}((U_i)(I_j)^T - M_{ij})^2) \tag{3}$$

After modelling the user and item representations it would be possible to feed the model with a user representation to predict all ratings of a user. This rating could be compared to the actual ratings to get a possible recommendation.

A advantage of this technique is that all predicted ratings are very personilzed. Furthermore model based filters can be scaled more easily than memory based filters.

Disadvantages are cold start problems, no explainability why interactions are predicted as they are and thus a possible bias.

Hybrid Hybrids of content and collaborative filtering techniques are in common because they supplement each other very well. Especially because of deep-learning various features can be used at ones. [10]

3 Universal, language-agnostic sentence embeddings

Because we are focusing on natural language processing and especially on news recommendations it is beneficial to get a vector representation for natural language. For this purpuse Universal, language-agnostic sentence embeddings (LASER) will be used. [7]

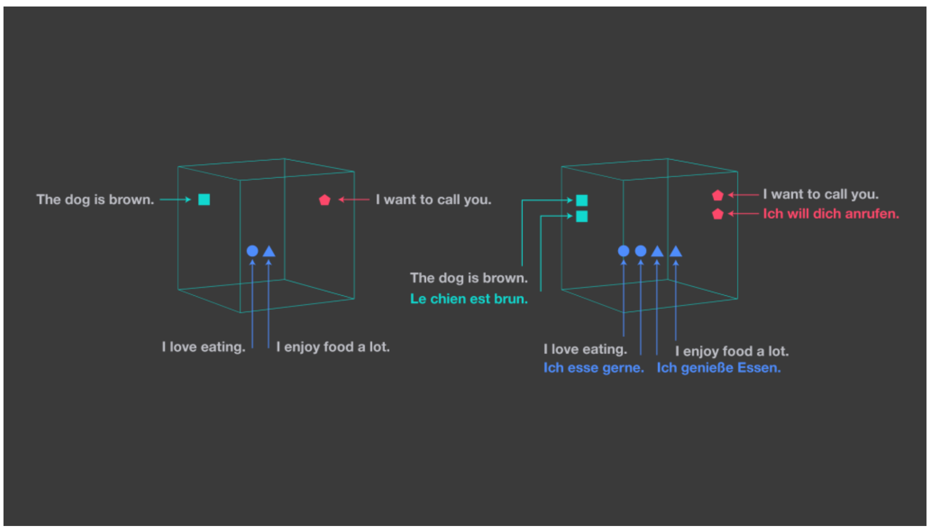

Fig. 2. Visualization of sentence to vector mappings by LASER in a 3-Dimensional space [8]

"LASER's vector representations of sentences are generic with respect to both the input language and the NLP task. The tool maps a sentence in any language to a point in a high-dimensional space with the goal that the same statement in any language will end up in the same neighborhood. " [8]

The advantage is that it is no longer needed to train models in different languages instead every sentence language can be mapped language independent to a vector where the vector represents the topic, intent etc..

This objective is fullfilled by training a autoencoder which translates sentences in 93 different languages into sentences of two target languages. The target languages are spanish and english. 223 million parallel sentences have been used to accomplish the objective.

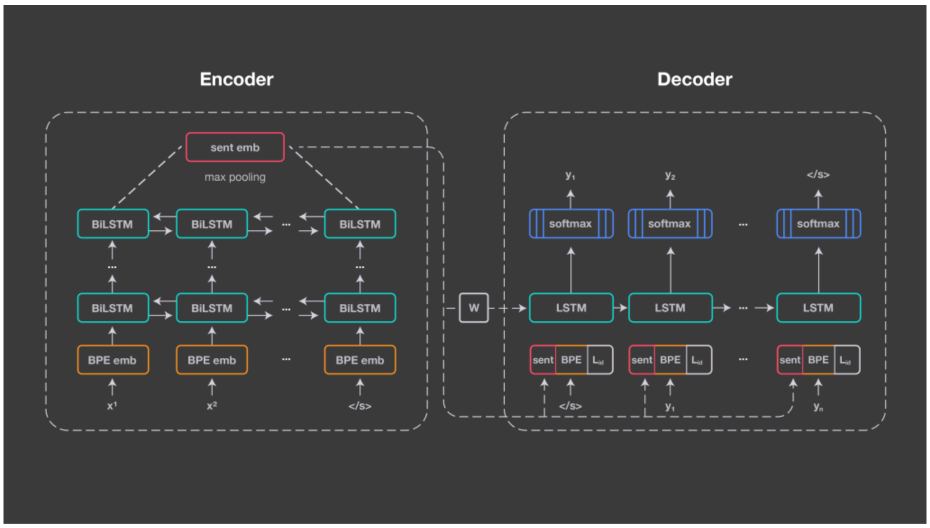

Fig. 3. LASER Architecture [8]

The architecture is based on bidirectional LSTMs on the encoder and a simple LSTM as a decoder. After training, the encoder outputs a context vector which then can be used as a sentence embedding. For tokenization byte pair encoding (BPE) is used. [1]

4 Transformer

If we want a precomputed user representation based on the read history a autoencoder suggests itself. Considering that the read history of a user can have a variable length we also would need a autoencoder which can handle series. Recurrent neuronal networks can perform very well on many tasks but scale poorly with more than one GPU. Nowadays Transformers can be used to work against this issue. Transformers have already been used effectively for recommender systems. [9] Transformers make effective usage of multiple GPUs and are not sequential. It introduces several new techniques like the positional embeddings or multi-head attention. In recurrent networks the order of words are implicite processed by the model. Transformers instead use positional embeddings to give the model information about the position of an input in a series. The multi-head attention is a layer which allows the model to train relations between entries of a series. The output of the softmax function in the attention function can be evaluated by humans too. [9]

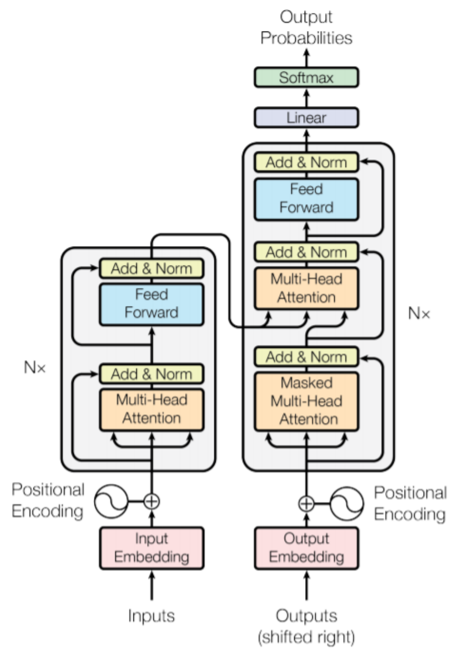

Fig. 4. Transformer architecture [9]

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{4}$$

$$\begin{aligned} &MultiHead(Q, K, V) = Concat(head_1, ..., head_h)H^O \\ &where \\ &head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \end{aligned} \tag{5}$$

5 Outlook

I have several possible projects in mind.

- I will try to combine content based, model based collaborative filtering, LASER and the Transformer architecture.

  At first LASER will generate a vector for each text to provide a accurate representation of them. These vectors will then be feed as read history for a user into a transformer autoencoder which will then be trained to generate user embeddings. Furthermore user features will be used to train the transformer.

  The pregenerated user embeddings and text embeddings will then be used as pretrained user and item representations for matrix factorization. The user-item interaction matrix, where a interaction is the indication that a user has read the text, will then be approximated by a model with the representations as inputs.

  In addition a approach against the cold start problem would be tackeled. This appoach focuses on generating a user history only by taking the user features. This user history can then be feed into the previosly trained transformer to get a user representation.

- Another possible project would be to simply train a transformer encoder to predict the next click of a user, taking the reading history and some user features as inputs.[10]
  The text embeddings would also be precomputed by LASER.

6 Possible Problems

At first the native transformer encoder-decoder architecture does not support the training of an autoencoder which generates one contenxt vetor because the decoder does not only take one context vector as input but all output vectors of the encoder. A transformer encoder outputs a matrix with the dimensions ixd where i is the input length and d a fixed dimension thus every input vector produces an output vector which will then be used by the decoder. This architectual problem would be tackeled in the project.

A problem for the second idea would be that for every new text the model would need to be retrained to consider the new text for recommendations too. This problem would not appear that often in the first idea because all big networks are trained to perform with various texts and user histories.

References

1. Rico Sennrich, Barry Haddow, Alexandra Birch: Neural Machine Translation of Rare Words with Subword Units (2015)
2. Fabrizio Germano, Vicenç Gómez, Gaël Le Mens : The Few-get-richer: A Surprising Consequence of Popularity-based Rankings (2019)
3. Faisal Rahutomo, Teruaki Kitasuka, and Masayoshi Aritsugi :Semantic Cosine Similarity (2012)
4. Yu Zhu and Jinhao Lin and Shibi He and Beidou Wang and Ziyu Guan and Haifeng Liu and Deng Cai: Addressing the Item Cold-start Problem by Attribute-driven Active Learning (2018)
5. Resnick and Varian: Recommender systems (1997)
6. Lalita Sharma, Anju Gera: A Survey of Recommendation System: Research Challenges
7. Mikel Artetxe, Holger Schwenk Massively Multilingual Sentence Embeddings for Zero-Shot Cross-Lingual Transfer and Beyond (2019)
8. https://engineering.fb.com/ai-research/laser-multilingual-sentence-embeddings/ (2019)
9. Ashish Vaswani and Noam Shazeer and Niki Parmar and Jakob Uszkoreit and Llion Jones and Aidan N. Gomez and Lukasz Kaiser and Illia Polosukhin: Attention Is All You Need (2017)
10. Qiwei Chen, Huan Zhao, Wei Li, Pipei Huang, Wenwu Ou: Behavior Sequence Transformer for E-commerce Recommendation in Alibaba (2019)
11. Shuai Zhang, Lina Yao, Aixin Sun, Yi Tay: "Deep Learning Based Recommender System." ACM Computing Surveys 52.1 (2019): 1–38. Crossref. Web.
12. Yehuda Koren, Robert Bell and Chris Volinsky: MATRIX FACTORIZATION TECHNIQUES FOR RECOMMENDER SYSTEMS (2009)
13. https://towardsdatascience.com/recommendation-system-part-1-use-of-collaborative-filtering-and-hybrid-collaborative-content-in-6137ba64ad58