Transparent Text-Based Item Recommendation: Introducing TRecX

Pablo Pérez-Núñez*, Jorge Díez*, Antonio Bahamonde*, Oscar Luaces*
* *Artificial Intelligence Center*
University of Oviedo, Gijón, Asturias, Spain
{pabloperez, jdiez, abahamonde, oluaces}@uniovi.es

***Abstract*—Recommender systems have proven their usefulness both for companies and customers. The former increase their sales and the latter get a more satisfying shopping experience. These systems can benefit from the advent of explainable artificial intelligence since a well-explained recommendation will be more convincing and may broaden the customer's purchasing options. Many approaches offer justifications for their recommendations based on the similarity (in some sense) between users, past purchases, etc., which requires some knowledge of the users. In this paper, we present a recommender system with explanatory capabilities which can deal with the so-called cold-start problem since it does not require any previous knowledge of the user. Our method learns the relationship between the products and some relevant words appearing in the textual reviews from previous customers for those products. Then, starting from the textual query of a user's request for a recommendation, our approach elaborates a list of products and explains each recommendation based on the compatibility between the query's words and the relevant terms for each product.**

***Index Terms*—Recommender systems, Explainability, Cold-start, Text-based recommendations**

I. Introduction

It has been proven that the use of recommender systems increases the consumption of products offered by online platforms. However, a lack of confidence in the decisions taken by algorithms has been raised lately. Explainable Artificial Intelligence has been devised to tackle this issue. It is a recent branch of AI aimed at explaining the output of intelligent algorithms in a human-friendly manner, such that users can understand the reasons behind the algorithms' decisions [1]. In this paper, we present TRecX, a recommender system capable of explaining its recommendations to a user thanks to the textual reviews written by other users.

II. Proposed method

Our starting point is a set of users, $\mathcal{U}$, and a set of products, $\mathcal{P}$, collected in a dataset, $\mathcal{D}$, such that there is a record for every interaction between any $u \in \mathcal{U}$ with any $p \in \mathcal{P}$. Each interaction reflects the user satisfaction in form of a review r. We want to create a model capable of recommending products from textual user queries (asking for a recommendation). As we do not have queries in our dataset, we are going to use the textual reviews for that matter. The rationale behind this approach is to consider users' queries as a kind of *a priori* review. To make this possible, our model must learn the relationship between the words in the review and the product they refer to. Then, taking advantage of these relationships, we could explain the recommendations given.

A. Selection of relevant terms

Since we are going to use the words from the reviews to explain, we have to get rid of all the terms that may be useless for the explanation. Our proposal consists of applying Part-of-Speech tagging techniques [2] to get rid of all words except nouns and adjectives (more explanatory). Then, we will consider as relevant terms (i.e. our corpus) a 10% of the most frequent nouns and adjectives in all the reviews. Once we have the set of relevant terms we can encode the reviews using Bag of Words (BoW), to feed the model during training as well as during its operation, once trained.

B. Computing the recommendation

The proposed model is a classifier based on a multinomial logistic regression. It is trained with textual reviews, previously encoded using BoW, term frequency and $L1$ normalization. The resulting vector is fed as input to a fully connected layer and the transformed into a vector of probabilities thanks to a *softmax* activation function. Thus, the classifier outputs a vector whose dimension must coincide with the number of products, $|\mathcal{P}|$, that can be recommended. We pose a multiclass learning task, where the parameters of our layer (matrix $\boldsymbol{W}$) are learned, and where the loss function to be optimized is the usual categorical cross-entropy.

C. Explaining the recommendation

Once the model is trained, explanations can be extracted using the weights learned during the training stage, i.e., the matrix $\boldsymbol{W}$. Our explanations will show the user the weight that the relevant words of their query have in the recommended item. We will also display a word cloud with the terms that have the highest weight in the recommended item. The weight of every word in each product can be extracted by looking at the corresponding columns and rows in $\boldsymbol{W}$.

III. Experimental validation

To validate our approach, we have constructed a group of datasets with textual restaurant reviews downloaded from TripAdvisor.

Each dataset contains data from one of five cities selected based on several factors, such as population, number of restaurants, geographic location and language. The selected cities are *Gijón*, *Barcelona*, *Madrid*, *Paris* and *New York*. We will evaluate the recommendation performance of our approach on these datasets against two other systems.

A. Restaurant recommendation: implementation and results

In order to train our model, we preprocessed the textual information of each dataset as follows: case normalization (lower case), elimination of punctuation symbols, lemmatization (transforming all the variations and inflected forms of words into its common lemma), and elimination of accent marks and numbers. We also removed long reviews, with more than 2000 characters, and restaurants with less than 100 textual reviews. After preprocessing, we have divided each dataset into *training*, *validation* and *test* (80%, 10% and 10% respectively). The splits were made randomly but ensuring that all restaurants appear in the *training* set (unknown products cannot be recommended). Since our system operates in cold-start scenarios, we do not have to do the same for users.

We compared the performance of TRecX with a more complex classifier based on a *long-short term memory (LSTM)* network [3], much more powerful for NLP tasks. The input encoding used for LSTM2rst was also more sophisticated: we used a *word2vec* [4] approach instead of the BoW encoding used by TRecX. The reason for comparing TRecX with LSTM2rst is to test if it is worth the trade-off between the possible penalty in performance of TRecX and its ability to offer explanations for its recommendations. We also included, as a *baseline*, the precision obtained when always recommending the most popular restaurants.

TABLE I: Precision@{1,5,10} in percentage obtained by the systems. Best results for each metric and city are in bold.

Dataset	baseline			TRecX			LSTM2rst		
	P@1	P@5	P@10	P@1	P@5	P@10	P@1	P@5	P@10
GJN	6.8	17.6	25.7	**35.9**	**60.2**	**70.5**	34.8	58.4	69.6
BCN	0.7	3.0	4.8	28.1	46.5	54.5	**29.0**	**49.1**	**57.7**
MDR	1.3	4.4	6.7	**34.6**	53.8	61.0	34.4	**54.9**	**63.0**
PRS	0.8	2.1	3.1	23.7	38.5	44.9	**27.4**	**43.3**	**50.3**
NYC	1.9	5.9	8.8	38.8	57.6	64.8	**40.4**	**58.9**	**66.5**

As expected, the baseline is the worst performing. The difference in precision between the smallest city (Gijón) and the rest is noteworthy. The performance of LSTM2rst is, in general, better than that of TRecX, as expected. However, the scores obtained by TRecX are only slightly worse, despite using only 10% of the words in the textual reviews as input information. Moreover, it is straightforward to elaborate an explanation for the recommendations, as we already introduced in Section II-C.

B. Explanation of the recommendation

Our proposal learns the relevance of the most important linguistic terms which characterize the products, and this relevance, gathered in the matrix $\boldsymbol{W}$, is then used to elaborate explanations in the same linguistic terms appearing in the users' queries. Figure 1 depicts a mock-up of a restaurant recommender application based on TRecX trained on real New York City data. The query in the upper part of the figure triggers a suggestion of the top n (4 in the mock-up) restaurants with the highest probability to be related to the relevant terms of the query. Pointing to a given restaurant, the system can highlight some words of the query with an intensity proportional to their relevance for the recommendation, i.e., proportional to their weights in the corresponding row of $\boldsymbol{W}$. Additionally, the user can see a word cloud built up depending on all the weights of the row which corresponds to the selected restaurant.

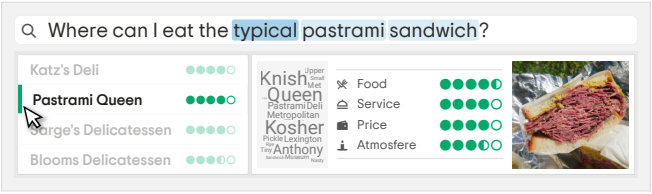

Fig. 1: Mock-up of a recommender application with explanations. The results shown were obtained from real recommendations and explanations from TRecX in New York City data.

IV. Conclusions

In this paper, we proposed TRecX, a text-based recommender system with explanatory capabilities. Our approach can learn the relationship between products and relevant words in customer reviews, allowing it to provide recommendations and user-friendly explanations for those recommendations. By highlighting the most relevant terms that define a product, our system helps users understand how the recommendation was generated and make informed decisions.

We conducted experiments on a dataset of restaurant reviews from five cities to validate our approach. The results showed that TRecX was able to provide accurate recommendations outperforming two baseline approaches and high-quality explanations. We have also demonstrated that our method adapts to different languages and dataset sizes and can also work in cold-start scenarios.

V. Acknowledgements

This work was funded by grant PID2019-109238GB-C21 (Ministry of Science and Innovation, Spain). The participation of Pablo Pérez-Núñez was funded by the Principality of Asturias through the predoctoral granting program *Severo Ochoa* (ref. BP19-012). We also thank NVIDIA Corporation for the donation of a Titan Xp GPU used in this research.

References

[1] D. Monroe, "Ai, explain yourself," *Communications of the ACM*, vol. 61, no. 11, pp. 11–13, 2018.
[2] H. Schmid, "Part-of-speech tagging with neural networks," 1994.
[3] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[4] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv:1301.3781*, 2013.