Lit@EVE: Explainable Recommendation based on Wikipedia Concept Vectors

M. Atif Qureshi (✉) and Derek Greene

Insight Center for Data Analytics, University College Dublin,
{muhammad.qureshi@ucd.ie, and derek.greene@ucd.ie}

Abstract. We present an explainable recommendation system for novels and authors, called *Lit@EVE*, which is based on Wikipedia concept vectors. In this system, each novel or author is treated as a concept whose definition is extracted as a concept vector through the application of an explainable word embedding technique called *EVE*. Each dimension of the concept vector is labelled as either a Wikipedia article or a Wikipedia category name, making the vector representation readily interpretable. In order to recommend items, the *Lit@EVE* system uses these vectors to compute similarity scores between a target novel or author and all other candidate items. Finally, the system generates an ordered list of suggested items by showing the most informative features as human-readable labels, thereby making the recommendation explainable.

1 Introduction

Recently, considerable attention has been paid to providing meaningful explanations for decisions made by algorithms [**?**]. On the legislative side, the European Union has approved regulations that requires a "right to explanation" in relation to any user-facing algorithm [**?**]. This increased emphasis on the need for explainable decision-making algorithms is the first motivation for our work. As further motivation, increasingly recommender systems attempt to offer serendipitous suggestions, where the items being recommended are relevant but also potentially different from those items which they users seen previously [**?**]. To address both of these motivations, we propose the *Lit@EVE* system, which makes use of Wikipedia articles and categories as a rich source of structured features. Furthermore, to explain the similarity between items, the system makes use of our previously-proposed word embedding algorithm called *EVE* [**?**]. Word embedding algorithms generate real-valued representation for words or concepts in a vector space, allowing simple comparisons to be made between them by operating over their corresponding vectors. In the case of *EVE*, the dimensions of this space are human-readable, as each dimension represents a single Wikipedia article or category. We demonstrate this approach in the context of recommending books and authors, where *EVE* concept vectors are used to represent both authors and their literary works.

Recently, Chang et al. [**?**] described a crowdsourcing-based framework for generating natural language explanations which relies on specific human-generated annotations, whereas our system harnesses the ongoing work of Wikipedia editors, and automatically assigns labels to explain a given recommendation. Moreover, the use of a rich set of Wikipedia articles and categories as features helps to highlight serendipitous aspects of recommended items which are otherwise difficult to discover.

2 System Overview

We now present an overview of the *Lit@EVE* system. First, we discuss the dataset used to build our recommender, then we discuss the corresponding *EVE* word embeddings, and finally we show how recommendations are generated using the system.

2.1 Dataset

Our dataset is based on the curated "Wikiproject novels"[1] list which contains 49,999 Wikipedia entries (as of 20 April 2017) relating to literature. Many of these entries correspond to novels, although some denote other literary concepts, such as genres, publishers, and tropes. In order to exclusively extract novels, we include only those with a Wikipedia info box that contains an "author" attribute. This filtered set has 18,572 entries corresponding to novels. From the author attribute of each entry, we discovered 2,512 unique authors. Our combined dataset contains both the novel and author entries.

2.2 Concept Embeddings

The *EVE* algorithm generates a vector embedding of each word or concept by mapping it to a Wikipedia article[2] [**?**]. For example, the concept "Harry Potter" is mapped to the Wikipedia article of the novel "Harry Potter". After identifying the concept article, *EVE* generates a vector with dimensions quantifying the association of the mapped article with other Wikipedia articles and categories. In the case of articles, *EVE* exploits the hyperlink structure of Wikipedia. Specifically, associations are calculated as a normalised sum of the number of incoming and outgoing links between the concept article and other Wikipedia articles. Furthermore, a self-citation is also added for the concept article. To quantify associations with Wikipedia categories, *EVE* propagates scores from the concept article to other related Wikipedia categories – e.g., "Harry Potter" has related categories "Fantasy novel series", "Witchcraft in written fiction", etc. Each of the related categories receives a uniform score which is propagated to neighbouring categories (i.e., super and sub categories) by means of a factor called *jump probability*. The propagation continues until a maximum hop count is reached, which prevents topical drift. The final embedding vector for the concept is constructed from the associations for all articles and categories. For further details on the construction of embedding vectors refer to our paper on *EVE* [**?**]. We apply this process for all novels and authors in our dataset. The resulting vectors form the input for *Lit@EVE* to generate explainable recommendations.

2.3 *Lit@EVE* Recommendations

Lit@EVE generates recommendations via a two step process. Firstly, it embeds domain-specific knowledge in the *EVE* vectors, and then it applies a similarity function to these vectors to rank candidate recommendations.

[1] `https://en.wikipedia.org/wiki/Wikipedia:WikiProject_Novels`

[2] Either an exact match or a best match.

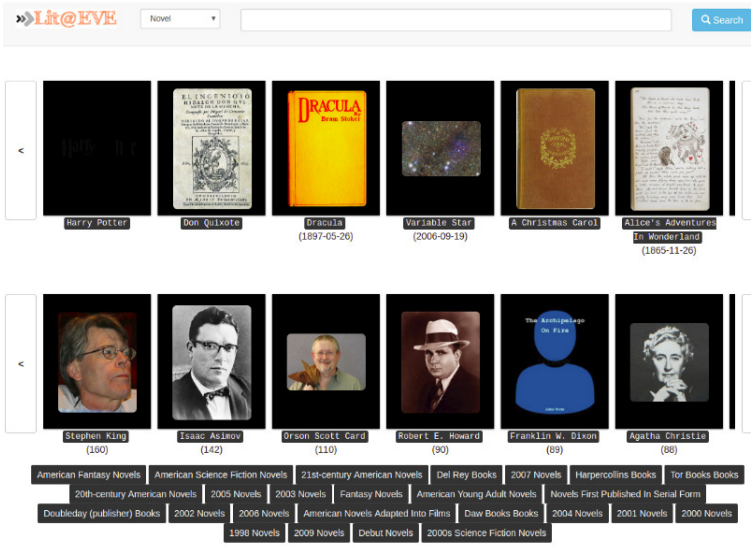

Fig. 1. The *Lit@EVE* interface supports three selection levels – novels, authors, and features.

Domain-specific vector rescaling: To generate recommendations, we eliminate rare dimensions from the *EVE* vector embeddings for novels and authors and incorporate domain-specific knowledge in the vector embeddings. This is done as follows. First, we calculate the item frequency of each dimension (i.e., the number of novels or authors with a non-zero association for this value). Dimensions with a frequency < 3 are eliminated from the model. This limits the dimensionality to 156,553 unique features for novels and authors. Next, we scale the dimensions by the inverse item frequency of each dimension. Furthermore, each association of the Wikipedia hyperlink in the vector representation is scaled by the importance of the Wikipedia hyperlink which is calculated by PageRank score [**?**]. Finally, the vectors are normalised to unit length.
Generating recommendations: The rescaled vectors representing novels and authors are used to generate recommendations. For a target novel or author, we calculate cosine similarities between that item and the rest of the items in the dataset. The candidate list is then sorted by similarity to identify the top recommended items. Each recommended item is explained by the most informative features i.e., the embedding dimensions which maximise the similarity score between the target and recommend item; we select top-n informative features where n equals 10 for this demonstration. The explanation corresponding to the informative feature is the label of that dimension (e.g. "American Horror Novelist").

3 User Interface

Figure 1 shows the query-based exploratory interface of *Lit@EVE*. Users may query or select an item (a novel or author) which allows for further exploration through ex-

plainable recommendations. Each novel suggested to a user is explained through features such as "Novels Set In Kansas", while each suggested author is also explained with features such as "British Writers". Alternatively, users may opt to browse items strongly associated with features, such as "Fantasy Novels" or "Victorian Novelists". The following use-cases illustrate the various aspects of recommendations generated by *Lit@EVE*:

- Selecting the novel "Harry Potter and the Order of the Phoenix" suggests "The Lord of the Rings" as the recommended novel, with common features such as both being "BILBY Award-winning works", both being "Sequel Novels", and both involving a plot having "Fictional Prisons".
- Selecting the author "Terry Pratchett" offers a list of similar author recommendations e.g. "John Fowles". Both are explained with common features such as "English Humanists", "English Atheists ", "20th-century English Novelists".
- Selecting the feature "Nautical Fiction" offers a list of novel recommendations from genres such as "Adventure novel", "Historical Fiction", and "Children's fantasy novel". This may be interesting to a user who is interested in "Nautical Fiction" who would like to browse novels from different genres which incorporate aspects of nautical fiction.

An interesting aspect of the explanations associated with our recommendations is the granularity at which they help users to discover serendipitous aspects around a given novel or author. For instance, in the first use case above, the feature "BILBY Award-winning works" connects diverse works that have won this children's book award, potentially allowing users to make serendipitous discoveries of novels of this type. For further details on the unique aspects of recommendations generated by *Lit@EVE*, we refer the reader to an online video demonstration of the system[3].

Acknowledgments: This publication has emanated from research conducted with the support of Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/2289.

References

1. S. Chang, F. M. Harper, and L. Terveen. Crowd-based personalized natural language explanations for recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*, pages 175–182. ACM, 2016.
2. B. Goodman and S. Flaxman. European union regulations on algorithmic decision-making and a "right to explanation". *arXiv preprint arXiv:1606.08813*, 2016.
3. D. Kotkov, S. Wang, and J. Veijalainen. A survey of serendipity in recommender systems. *Knowledge-Based Systems*, 111:180–192, 2016.
4. L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.
5. L. Qiu, S. Gao, W. Cheng, and J. Guo. Aspect-based latent factor model by integrating ratings and reviews for recommender system. *Knowledge-Based Systems*, 110:233–243, 2016.
6. M. A. Qureshi and D. Greene. EVE: Explainable vector based embedding technique using Wikipedia. *arXiv preprint arXiv:1702.06891*, 2017.

[3] `http://mlg.ucd.ie/liteve/`