Improving Media Content Recommendation with Automatic Annotations

Ismail Harrando[1], Raphaël Troncy[1]

[1]EURECOM, France

Abstract

With the immense growth of media content production on the internet and increasing wariness about privacy, content-based recommendation systems offer the possibility of promoting media to users (e.g. posts, videos, podcasts) based solely on a representation of the content, i.e. without using any user-related data such as views and more generally interactions between users and items. In this work, we study the potential of using off-the-shelf automatic annotation tools from the Information Extraction literature to improve recommendation performance without any extra cost of training, data collection or annotation. We experiment with how these annotations can improve recommendations on two tasks: the traditional user history-based recommendation, as well as a purely content-based recommendation evaluation. We pair these automatic annotations with the manually created metadata and we show that Knowledge Graphs through their embeddings constitute a great modality to seamlessly integrate this extracted knowledge and provide better recommendations. The evaluation code, as well as the enrichment generation, is available at https://github.com/D2KLab/ka-recsys.

Keywords

Recommender Systems, Content-based Recommendation, Knowledge Graph, Automatic Annotation

1. Introduction

As user engagement with content online has become a crucial element in most if not all content-providing multimedia platforms – i.e. retaining a user's interest in the provided content and maximizing their time watching/reading/listening to the content, the role of recommender systems cannot be overstated in shaping and improving the user experience when it comes to consuming and interacting with said content, as it helps funneling the usually overwhelming amount of data into a condensed, targeted and interesting selection of items that the user is most likely to find enjoyable and interesting. Traditionally, recommendation systems either use **collaborative filtering**, i.e. leveraging user statistics and their implicit/explicit feedback (views, likes, watch time) to find items to recommend (the underlying assumption is that people who have similar interests interact with the same items), or provide **content-based** recommendations, which rely on the content of the item itself to find similar content without any input from the user. Content-based recommendations are particularly interesting in the case of the *cold start problem* where there is no feedback from users (no interactions to based the recommendations out of), and in cases where it is hard to collect such feedback (anonymity, privacy).

In this paper, we are interested in the second kind of recommendations which are based solely on the content of the media to recommend. The "content" in content-based can refer to a variety of potential formats: text, image, video, metadata (e.g. tags and keywords) and so on. Typically, a representation of such content is extracted or learned, and the task of recommendation is then cast as a content similarity/retrieval task: given the representation of an item of interest (e.g. the video the user is currently watching), and the representation of all items already existing in the catalog, we want to find the items which have the highest similarity to the item of interest. While many varieties of this approach exist (ones that target other metrics such as *serendipity* [1], *diversity* [2] and *explainability* [3]) which may formulate the problem differently, but at its core, the task can be framed as finding the best content representation that allows uncovering a meaningful measure of similarity.

We posit in this paper that the use of Knowledge Graphs (KGs), both created using item metadata and automatically generated from the given content, can improve the task of media recommendation. Instead of relying only on the content, we leverage several Information Extraction techniques to extract high level descriptors that allow the automatic creation of metadata, which can be then used to generate a KG connecting all content in the media catalog. Given the versatility of Knowledge Graphs, they allow us to combine these automatic annotations with already existing metadata seamlessly. To validate this approach, we focus on studying the TED dataset [4], an open-sourced multimedia dataset that of-

3rd Edition of Knowledge-aware and Conversational Recommender Systems (KaRS) & 5th Edition of Recommendation in Complex Environments (ComplexRec) Joint Workshop @ RecSys 2021, September 27–1 October 2021, Amsterdam, Netherlands

ismail.harrando@eurecom.fr (I. Harrando); raphael.troncy@eurecom.fr (R. Troncy)

0000-0002-3593-4490 (I. Harrando); 0000-0003-0457-1436 (R. Troncy)

fers the unique possibility of evaluating recommendations based on both the content only ("related videos", as curated by human editors) and the user preferences based on their interactions history. We demonstrate that our approach improves the recommendation performance on both tasks, and that KGs are a reliable framework to integrate external knowledge into the task of recommendation.

2. Related Work

The TED Dataset The TED dataset [4] is a multimodal dataset which contains the audiovisual recordings of the TED talks downloaded from the official website [1], which sums up to 1149 talks, alongside metadata fields and user profiles with rating and commenting interactions. The metadata fields are as follows: identifier, title, description, speaker name, TED event at which the talk is given, transcript, publication date, filming date, and number of views. For nearly every video, the dataset contains a list of user interactions (marked by the action of "Adding to favorites"), as well as up to three "related videos", which are picked by the editorial staff to be recommended to the user to watch next. What is unique for this dataset is that it provides two sorts of ground truths for the recommender system use-case, that we can formulate in these two tasks:

- **Task 1 - Personalized (user-specific) recommendations**: based on a user's list of *favorite* talks, the task is to predict what they would watch next. A evaluation dataset can thus be created using a "leave one out" protocol, i.e. removing one interaction from the user list of favorites, and measuring how successful a method is in predicting the omitted item. Most recommender system-type datasets contain a similar information, i.e. what items a user has actually interacted with in reality, based on their viewing/interaction history. This task is usually handled with collaborative filtering methods (e.g. [5]), but is still interesting for content-based recommendation in the case of the *cold start problem*: when a new talk is added to the platform, how can we recommend it to other users? The most common approach is to use its content to recommend it to users who previously liked a similar content.
- **Task 2 - General (content-based) recommendations**: to the best of our knowledge, this is the only dataset which offers ground truth for multimedia recommendations based on content only, which are referred to as "related videos", manually annotated by TED editorial staff. These are supposed to reflect subjective topical relatedness between talks in the corpus. Performance on this task reflects the model's ability to recommend content to either users without an interactions history (new users, visitors without accounts) or new videos (that have not yet received any interactions). We note that in the ground truth, some talks are associated with three related talks, some with two, and some with only one. We account for this in the evaluation metrics.

Previous works have studied specific aspects of this dataset such as sentiment analysis [6], estimating trust from comments polarity and ratings to improve recommendation [7], or studying hybrid recommender systems [8]. In this work, we focus our interest on this dataset as it offers a unique possibility of evaluating content-based recommendation using both real user feedback and hand-picked recommendations, as the later has not been considered in any of the published works on this dataset to the best of our knowledge.

We also note that, while the dataset is multimodal (TED Talks Videos are also available), our work does not tackle visual information extraction, mainly because TED Talks are not visually diverse (mostly speakers and audience wide shots). This is however a promising direction of work that has been tackled in previous works [9].

Graph-based Recommender Systems Given the recent growing interest in Knowledge Graphs and their applications, there is a growing literature on the techniques and models that can be leveraged to build "knowledge-aware" recommender systems. [10] present such an approach to bring external knowledge to the task of content-based Knowledge Graphs, identifying two main approaches to what they called "Semantics-aware Recommender Systems" to tackle traditional problems of content-based recommender systems, *Top-down Approaches* which incorporate knowledge from ontological resources such as WordNet [11], and encyclopedic knowledge sources such as Wikipedia[2], to enrich the item representations with external world and linguistic knowledge, and *Bottom-up Approaches* which uses linguistic resources such as what we commonly refer to as distributional word representations, e.g. using pretrained word embeddings to avoid the issue of exact matching in traditional content-based systems. They also raise the problem of the potential use of a graph structure to discover latent connections among items, which we study in our experiments. [12] offers an extensive survey of Knowledge Graph-based Recommender System approaches, proposing a high-level taxonomy of methods that either use graph embeddings, connectivity patterns

[1] https://www.ted.com

[2] https://en.wikipedia.org/wiki/Main_Page

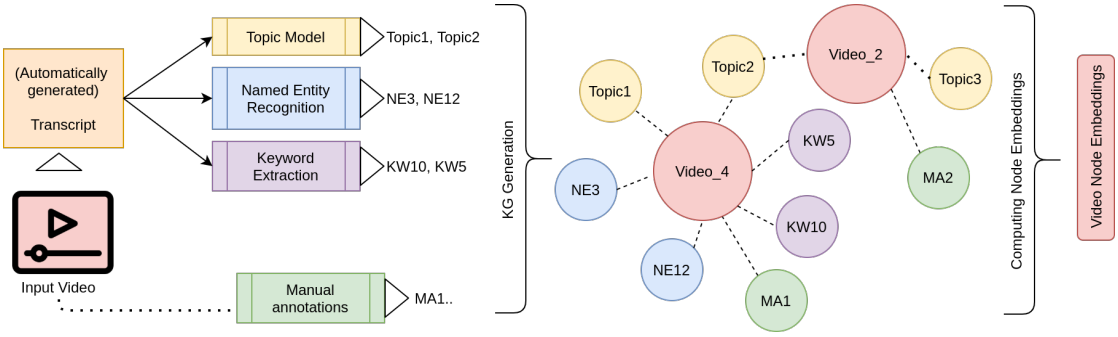

Figure 1: High level illustration of the approach: we start by extracting annotations from the video transcript using off-the-shelf Information Extraction tools, which we combine with manual annotations to create a Knowledge Graph, where the talks and the annotations are nodes, connected with the corresponding semantic relation. Using this graph structure, we can generate continuous fixed-dimensional representations using a Graph Embedding technique, which we can later use to measure content similarity for recommendation.

(common paths mining), or combining the two. In this paper, we only focus on embedding-based methods to study the use of automatic annotations on the performance of recommender systems. Additionally, unlike some previous works, our work does not tackle the two tasks jointly as a learning problem[13], but attempts to show how the same approach can at the same time improve the performance on both.

3. Approach

The proposed approach builds on using several Information Extraction techniques such as Topic Modeling (3.1), Named Entity Recognition (3.2), and Keyword Extraction (3.3), to generate high level descriptors – *annotations* – of the content of each video in the dataset. Once the annotations are generated for each video, we use them to build a Knowledge Graph connecting the talks by their annotations. This approach also allows us to integrate external metadata if such metadata is available (for our dataset, metadata such as "Tags" and "Themes" are available and will be used). Once the KG is generated, we can use a graph embedding method [14] to generate a fixed-dimensional embedding for each video in the dataset, such that videos having similar annotations would be represented in proximity in the embedding space. As a result, we can measure the (cosine) similarity between any two videos' embeddings as a proxy to their relatedness.

The approach is illustrated in Figure 1.

We present a selection of automatic annotations techniques and how they are used in our approach in the following subsections.

3.1. Topic Modeling

Topic modeling is a ubiquitously used Information Extraction technique, which attempts to find the latent topics in a text corpus. A topic can be roughly defined as a coherent set of vocabulary words that tend to co-appear with high probability in the same documents. When applied on documents of natural language, topic models have the ability to find the underlying "themes" in the document collection, such as sport, technology, etc.

The literature on topic modeling is rich and diverse, with approaches relying solely on word counts such as the commonly used LDA [15], to using state-of-the-art representations to represent documents in more meaningful representational spaces [16, 17]. Topics are usually represented with their "top N words" (the N words most likely to appear given a topic). In our dataset, we find topics such as:

- *Technology*: network,online,computers,digital,google
- *Environment*: waste,plants,electrical,plastic,battery
- *Gaming*: games,online,virtual,gamers,penalty
- *Health*: aids,malaria,drugs,mortality,vaccine

For our experiments, we use *LDA* as it is still commonly used and offers simple yet competitive performance[18]. We test two aspects of topic modeling that can influence the structure of the graph (the number of nodes and relations added) which are the number of topics (i.e. the number of topic nodes in the final KG), as well as the cutoff threshold reflecting the topic model's confidence is assigning a given topic to a given talk (which would affect the number of relations to topic nodes). We report the results in Section 4. For a better performance of the topic modeling task, we preprocess our dataset as follows:

1. Lowercase all words

2. Remove short words (less than 3 characters)
3. Remove punctuation
4. Remove the most frequent words (top 1%)

3.2. Named Entity Recognition

Named Entity Recognition is the task of extracting from unstructured text, terms or phrases that refer to named entities, i.e. real world objects that have proper names and can refer to one of several classes: persons, places, organizations, etc. Once extracted, these Named Entities can be used as high level descriptors for a text content. For example, if two talks mention "Einstein" and "Newton", they may have a similar topic. While this task used to rely on grammatical and hand-crafted features to designate what would constitute a Named Entity (e.g. starts with a capital letter), modern systems do without such hand crafted features [19, 20], but rely on combining the learning power of neural networks with annotated corpora of Named Entities.

In our experiments, we use SpaCy's [21] NER model which uses an architecture that combines a word embedding strategy using sub word features, and a deep convolution neural network with residual connections, which is "designed to give a good balance of efficiency, accuracy and adaptability"[3].

For our experiments, we keep the Named Entities belonging to the following classes: 'PERSON', 'LOC' (location), 'ORG' (organization), 'GPE' (geopolitical entity), 'FAC' (faculty), 'PRODUCT', and 'WORK_OF_ART'. We also experiment with the impact of keeping all extracted Named Entities or filtering some out based on frequency, thus altering the number of added nodes to the graph and their relations to the existing talks. We report the results in Section 4.

3.3. Keyword Extraction

Similarly to the two previous tasks, Keyword Extraction is the process of extracting terms of phrases that summarize on a high level the core themes of a textual document. Generally, the keywords (or sometimes called tags) are the terms or phrases that are explicitly mentioned in the text with a high frequency or are somehow relevant to a big portion of it.

For our experiments, we use KeyBERT [22], an off-the-shelf keyword extractor that is based on BERT [20], which extracts keywords by first finding the frequent n-grams, then measuring the similarity between their embedding and the embedding of the whole document. We experiment with keeping all keywords or filtering out rare ones and report the results in Section 4.

[3]urlhttps://spacy.io/universe/project/video-spacys-ner-model

4. Experiments and Results

In this section, we explain the experimental protocol and describe the results for the different experiments done to study the impact of using automatic annotations on recommendation performance. We first reintroduce the dataset and how it is going to be used in the rest of this section. Then, we define the metrics we use to measure this performance (Hit Rate, Mean Reciprocal Rate and Normalized Discounted Cumulative Gain), and the embedding method to use for the rest of the experiments. For each automatic annotation considered (i.e. Topics, Named Entities and Keywords), we consider several configurations, with and without the addition of the original metadata from the dataset. Finally, we observe the potential of combining the resulting automatically generated graph embeddings with the textual embeddings of the content, and show how the two complement each other to push the performance even higher.

4.1. Dataset

As mentioned previously, the TED Talks dataset has two versions of ground truths (or prediction tasks) for recommendation, namely:

- User-specific recommendations that are based on actual users interactions history (henceforth referred to as **T1**)
- Content-based recommendations, which are hand-picked by editors for each talk (henceforth referred to as **T2**)

For our evaluation purposes, to unify the evaluation for both tasks, we proceed as follows:

- For **T1**, we create a test split using the *leave-one-out protocol* that is commonly used in the literature [23], thus having a "training" set which contains all but one talk that the user interacted with (the user has to have at least two interactions otherwise they are dropped). We create a *user embedding* by averaging the computed embeddings of all talks in the training set. The top recommendations are then generated by taking the talks which have the highest similarity score (in the same KG embedding space) to the user embedding. We note that there is actually no actual training taking place, but this method allows us to leverage actual "historical" user behavior to evaluate purely content-based recommendation.
- For **T2**, we consider all "related videos" as a test set. In other words, for each talk, we compute its similarity to all other talks in the dataset, and we recommend the talks which score the highest.

4.2. Metrics

To evaluate the performance of our method, we use two commonly used metrics in the recommender systems literature. In the following paragraphs, T is the number of talks in the dataset, U is the number of users with at least 2 interactions in their history, K is the number of (ordered) model recommendations to considerate (we picked $K = 10$ in our results), t is a talk ID (which maps to its embedding), u is a user ID (which maps to its embedding, i.e. the average of the embeddings of all talks in the user's history), $rec_j(x)$ is the j^{th} recommendation by our model (x being a user ID for T1 and a talk ID for T2). $hit(x, j) = 1$ if the talk j is indeed in the ground truth for x, otherwise it is 0. $related(t)$ is the number of related talks in **T2** (which can be 1, 2 or 3). $rank(x, j)$ is the rank of talk j in the suggested recommendations for talk/user x by descending similarity score.

Hit Rate (HR@K): A simple metric to quantify the probability of an item in the ground truth to be among the top-K suggestions produced by the system. For **T1**, this means that the left-out item from the user history must be among the K most similar talks to the user embedding (as defined above). For **T2**, this means that the talk that was manually picked by editors is among the K-most similar talks in the embedding space.

For **T1** we get the formula:

$$HR@K = \frac{1}{U}\sum_{t=1}^{U}\sum_{i=1}^{K} hit(u, rec_i(u))$$

For **T2**, we normalize the counting of hits to account for the variance of number of talks in the ground truth so that the Hit Rate is 1 at best (i.e. when all related talks in the ground truth are included in the system's recommendations):

$$HR@K = \frac{1}{T}\sum_{t=1}^{T}\frac{1}{related(t)}\sum_{i=1}^{K} hit(t, rec_i(u))$$

Mean Reciprocal Rate (MRR@K): Similarly to $HR@K$, this metric also measures the probability of having ground truth recommendations among the system's predictions, but it also accounts for the rank (order) of the prediction: the closest it is to the top of the predictions, the better. For **T1** we get the formula:

$$MRR@K = \frac{1}{U}\sum_{t=1}^{U}\sum_{i=1}^{K}\frac{hit(u, rec_i(u))}{rank(u, rec_i(u))}$$

For **T2**, and again to account for varying number of talks in the ground truth, we slightly alter the previous formula so that it is equal to 1 if all related talks are occupying the top spots in the system predictions:

$$MRR@K = \frac{1}{T}\sum_{t=1}^{T}\frac{1}{\sum_{count=1}^{related(t)} 1/count}\sum_{i=1}^{K}\frac{hit(t, rec_i(t))}{rank(t, rec_i(t))}$$

4.3. Evaluation Protocol

The protocol is summarized in Figure 1. For each of the studied automatic annotations, we start by running our automatic annotation model (as described in 3). We then create a Knowledge Graph using on one hand the metadata provided in the dataset (each talk is labeled with a "tag" and a "theme"), and our automatically extracted descriptors on the other hand. Once we connect all the talks using these annotations, we run a Graph Embedding method (see Section 4.4) to generate an embedding for each talk in the dataset. These embeddings serve then as representations that we can use to measure similarities for both **T1** and **T2**.

4.4. Choice of embeddings

Throughout the experiments section, we generate a graph connecting the talks and their annotations. Next, we compute node embeddings for each talk in our dataset. While this choice is important for the overall performance of the final recommendation system, our focus in this paper is to demonstrate the utility of automatic annotations for improving content recommendation.

To bypass the need to select a proper graph embedding technique and the expensive hyperparameter finetuning that goes with it for each experiment, we simulate an ideal scenario where we start from the KG containing the talks and their manually annotated metadata from the original TED dataset, i.e. *tags* and *themes*. This would allow us to create a Knowledge Graph that does not contain any noisy or extraneous annotations. We compute the node embeddings for each talk using a selection of embedding algorithms contained in the `Pykg2vec` package [24][4], a Python library for learning representations of entities and relations in Knowledge Graphs using state-of-the-art models. We finetune each representation using a small grid-search optimization over learning rate, embedding size and number of training epochs. We also add the One-hot encoding of each talk (each talk is represented by a binary vector which represent the presence or absence of each tag and theme in the metadata) to see if there is an advantage for using graph embeddings over a simple flat representation of the nodes, i.e. whether the graph embeddings encode some semantics between the annotations that a simple binary representation cannot pick up on (e.g. the presence of one tag may be related to some

[4] https://github.com/Sujit-O/pykg2vec

other tag/theme, in other words that the annotations are not mutually orthogonal).

We report the results on tables 1 and 2, for **T1** and **T2**, respectively.

Embedding method	HIT@10	MRR@10
ConvE	0.0183	0.0062
DistMult	0.0088	0.0030
NTN	0.0533	0.0192
Rescal	0.0112	0.0031
TransD	**0.0765**	**0.0315**
TransE	0.0663	0.0258
TransH	0.0678	0.0251
TransM	0.0691	0.0268
TransR	0.0641	0.0234
One-hot	0.0661	0.0256

Table 1
The best performance of different embedding methods on T1

Embedding method	HIT@10	MRR@10
ConvE	0.0163	0.0094
DistMult	0.0176	0.0099
NTN	0.1244	0.0720
Rescal	0.0143	0.0083
TransD	**0.2403**	**0.1542**
TransE	0.2270	0.1352
TransH	0.2182	0.1309
TransM	0.2219	0.1316
TransR	0.1910	0.1123
One-hot	0.2215	0.1293

Table 2
The best performance of different embedding methods on T2

From these tables of results, we make the following observations:

- Over the studied configurations of hyperparameters, models generally have the same ranking in performance whether used on **T1** or **T2**, i.e. models which perform well on one task tend to perform well on the other task. This means that whatever properties an embedding method has, they seem to translate similarly on both tasks. The poor performance of some methods may be due to their high sensitivity to hyperparameter finetuning.
- Over the studied configurations of hyperparameters, translation-based methods perform the best empirically, with `TransD` [25] performing the best (by quite a margin) in both set of experiments. While further experiments may be needed to determine how much this performance is due to the nature of the dataset (size, sparsity, etc.) and the task itself, for our experiments, we will take this model as our embedding method of choice (with a learning rate of 0.001, embedding and hidden size of 300, all trained for 1000 epochs. The other hyperparameters are left at their default values).
- One-hot node embeddings perform well on both tasks, which shows that on clean, controlled, human-annotated metadata, a simple exact matching of metadata is good enough to produce good results. The fact that `TransD` outperforms One-hot embeddings even in this setting shows that the graph embeddings capture some semantics beyond exact matching, which means that it learns to find latent meaning between the tags and themes, which ultimately justifies the use of graph embeddings.

4.5. Automatic annotations

In this section, we observe the performance gain of the different automatic enrichment methods we have introduced in Section 3.

4.5.1. Topic Modeling

In Table 3, we report on the results of adding the output of the topic modeling annotations to the KG. We evaluate the results as we vary two parameters: the number of topics and the cutoff threshold (the confidence score above which we assign a talk to a given topic).

# topics	Threshold	HIT@10	MRR@10
	T1		
No topics added		0.0765	0.0315
10	0.03	0.0612	0.0246
10	0.3	0.0629	0.0262
40	0.03	0.0769	0.0317
40	**0.3**	**0.0782**	**0.0326**
100	0.03	0.0562	0.0220
100	0.3	0.0606	0.0230
	T2		
No topics added		0.2403	0.1542
10	0.03	0.2096	0.033
10	0.3	0.2135	0.1294
40	0.03	0.2365	0.1623
40	**0.3**	**0.2475**	**0.1716**
100	0.03	0.1921	0.1196
100	0.3	0.2074	0.1226

Table 3
The results of enriching the metadata KG with Topic nodes, varying the number of topics and the cutoff threshold

From this small sample of hyperparameters values, we see that both the number of topics and the cutoff thresh-

old impact the performance of the recommendation on both tasks. Performance improves when raising the cut-off threshold, which implies that when we only assign topics to talks, if the topic model is highly confident, it decreases the noisy relations in the graph and decrease the risk of accidentally connecting nodes that are not really topically similar. We also note that under the right configuration, we improve the performance on both metrics for both tasks, whereas in most other configurations the performance suffers. We note that with the number of topics one should find a value that is befitting the studied corpus, as the value 40 (inspired by the ground truth number of *themes* in the dataset) seems to give the best results.

Topic modeling is a task that is generally very sensitive to the initial hyper-parameters and subject to inherent stochasticity, which means that with enough experiments, it is likely to find a configuration of hyperpamaters (not only the number of topics and the cutoff threshold but also model-specific hyperparameters such as LDA's *alpha* and *beta*) that yields even better improvement over the reported results.

4.5.2. Named Entity Recognition

In Table 4, we report on the results of adding the output of the Named Entity Recognition annotations to the KG. We evaluate the results as we switch between keeping all entities we extracted in the KG and keeping only ones that appear with a high enough frequency: in our case, we only add nodes for entities that are mentioned more than 10 times in the corpus.

# mentions	HIT@10	MRR@10
T1		
No NEs added	0.0765	0.0315
All NEs added	0.0776	0.0304
More than 10 mentions	**0.0808**	**0.0314**
T2		
No NEs added	0.2403	0.1542
All NEs added	0.2435	0.1548
More than 10 mentions	**0.2575**	**0.1908**

Table 4
The results of enriching the metadata KG with Named Entity nodes, varying the number of filtered entities

From these results, we see that adding NEs improves the results of the recommender system, especially after removing rarely appearing Named Entities (either erroneous or superfluous mentions). We also notice that MRR increases significantly with this addition for **T2**, suggesting that the Named Entities are strong indicators of content relatedness.

4.5.3. Keywords Extraction

In Table 5, we report on the results of adding the output of the Keyword Extraction to the KG. We evaluate the results as we add either all extracted keywords or only the ones that the keyword extraction model assigned a high enough confidence score to. In our experiment, a confidence score above 0.3 has been chosen.

Confidence	HIT@10	MRR@10
T1		
No KWs added	0.0765	0.0315
All KWs added	0.0732	0.0295
Only with conf > 0.3	**0.0772**	**0.0322**
T2		
No KWs added	0.2403	0.1542
All KWs added	0.2398	0.1523
Only with conf > 0.3	**0.2494**	**0.1593**

Table 5
The results of enriching the metadata KG with Keywords nodes, varying the confidence threshold

4.5.4. Combining annotations

In Table 6, we summarize the results from previous experiments, and we see that the addition of the best configuration from each experimental setting into one KG further improves the results.

Annotation	HIT@10	MRR@10
T1		
No annotations added	0.0765	0.0315
Topics	0.0782	0.0326
Named Entities	0.0808	0.0314
Keywords	0.0772	0.0322
All	**0.0854**	**0.0355**
T2		
No annotations added	0.2403	0.1542
Topics	0.2475	0.1716
Named Entities	0.2575	**0.1908**
Keywords	0.2494	0.1593
All	**0.2613**	0.1584

Table 6
The results on both recommendation tasks with all the different annotations added to the KG

We observe that the automatic annotations overall improve the performance on the recommendation task on purely content-based recommendations (T2), but surprisingly, they do so even for user preference-based ones (T1), although the overall performance is still significantly

lower. One could argue that this is because users are usually interested in similar content to what they watched previously (in other words, all recommendation tasks are partially content-based). There is a possibility, however, that the user is likely to click on the suggested video in the "related" section, which creates a dependence between the two tasks that is impossible to untangle. This is beyond the scope of this paper, but it is interesting to study the feedback loop of recommendation in such setting. Finally, the results suggest that Named Entity Recognition contributes the most to the overall performance improvement of the system, as it is the closest to the overall performance and still gives a better absolute MRR score.

5. Conclusion and future work

In this work, we showed how combining the knowledge extracted automatically using Information Extraction techniques with the representational power of KG and their embeddings can improve the performance content-based media Recommender Systems without requiring any supervision or external data collection, as we demonstrated clear performance improvement as measured on two tasks: making recommendations based on manually curated recommendations, and based on actual users interaction history. Our results are reproducible using the code published at https://github.com/D2KLab/ka-recsys.

With these promising results showing actual improvement over relying only on human annotation, there are multiple paths for further exploration. First, other techniques from the information extraction literature can be investigated such as entity linking, aspect extraction, and concept mining, with more exploration to be done on the techniques already presented (i.e. experimenting with other approaches for Topic Modeling, Named Entity Extraction and Keyword Extraction). What's more, as shown experimentally, the way these automatic annotations are processed and filtered (thus changing the structure of the generated KG), the results can vary, which calls for further study of how to balance the quantity of automatic annotations and the cutback on the necessary noise that comes with it. Another direction of work is to further explore models that go beyond simple graph embeddings. We should also consider combining the results of such annotations with the original textual context, as our early experiments suggest that combining both the low-level features (text embeddings) and high level ones (graph embeddings) improve further upon the performance. Furthermore, as these extracted annotations live on a KG, multiple methods in the direction of *Explainable Recommendations* can be explored in tandem.

Finally, we would like to test this approach on other datasets to see if it can be as successful on other content-centric recommendation problems.

Acknowledgment

This work has been partially supported by the French National Research Agency (ANR) within the ANTRACT (ANR-17-CE38-0010) projects, and by the European Union's Horizon 2020 research and innovation program within the MeMAD (GA 780069) project.

References

[1] D. Kotkov, S. Wang, J. Veijalainen, A survey of serendipity in recommender systems, Knowledge-Based Systems 111 (2016) 180–192. URL: https://www.sciencedirect.com/science/article/pii/S0950705116302763.

[2] M. Kunaver, T. Požrl, Diversity in recommender systems – a survey, Knowledge-Based Systems 123 (2017) 154–162. URL: https://www.sciencedirect.com/science/article/pii/S0950705117300680.

[3] Y. Zhang, X. Chen, Explainable recommendation: A survey and new perspectives, Found. Trends Inf. Retr. 14 (2020) 1–101.

[4] N. Pappas, A. Popescu-Belis, Combining content with user preferences for ted lecture recommendation, in: 11th International Workshop on Content-Based Multimedia Indexing (CBMI), 2013, pp. 47–52.

[5] J. B. Schafer, D. Frankowski, J. Herlocker, S. Sen, Collaborative Filtering Recommender Systems, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, pp. 291–324.

[6] N. Pappas, A. Popescu-Belis, Sentiment analysis of user comments for one-class collaborative filtering over ted talks, in: 36th international ACM SIGIR conference on Research and development in information retrieval, 2013, pp. 773–776.

[7] A. Merchant, N. Singh, Hybrid trust-aware model for personalized top-n recommendation, in: Fourth ACM IKDD Conferences on Data Sciences, Association for Computing Machinery, 2017.

[8] N. Pappas, A. Popescu-Belis, Combining content with user preferences for non-fiction multimedia recommendation: a study on ted lectures, Multimedia Tools and Applications 74 (2013) 1175–1197.

[9] R. Sun, X. Cao, Y. Zhao, J. Wan, K. Zhou, F. Zhang, Z. Wang, K. Zheng, Multi-Modal Knowledge Graphs for Recommender Systems, Association for Computing Machinery, New York, NY, USA, 2020, p. 1405–1414. URL: https://doi.org/10.1145/3340531.3411947.

[10] M. de Gemmis, P. Lops, C. Musto, F. Narducci, G. Semeraro, Semantics-Aware Content-Based Recom-

mender Systems, Springer US, Boston, MA, 2015, pp. 119–159.

[11] G. A. Miller, Wordnet: A lexical database for english, Commun. ACM 38 (1995) 39–41. URL: https://doi.org/10.1145/219717.219748.

[12] Q. Guo, F. Zhuang, C. Qin, H. Zhu, X. Xie, H. Xiong, Q. He, A survey on knowledge graph-based recommender systems, 2020. URL: https://arxiv.org/abs/2003.00911.

[13] Y. Cao, X. Wang, X. He, Z. Hu, C. Tat-seng, Unifying knowledge graph learning and recommendation: Towards a better understanding of user preference, in: WWW, 2019. URL: https://arxiv.org/abs/1906.04239.

[14] H. Cai, V. Zheng, K. Chang, A comprehensive survey of graph embedding: Problems, techniques, and applications, IEEE Transactions on Knowledge and Data Engineering 30 (2018) 1616–1637.

[15] D. M. Blei, A. Y. Ng, M. I. Jordan, Latent dirichlet allocation 3 (2003) 993–1022.

[16] F. Bianchi, S. Terragni, D. Hovy, Pre-training is a hot topic: Contextualized document embeddings improve topic coherence, in: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), Association for Computational Linguistics, Online, 2021, pp. 759–766. URL: https://aclanthology.org/2021.acl-short.96.

[17] T. Tian, Z. F. Fang, Attention-based autoencoder topic model for short texts, Procedia Computer Science 151 (2019) 1134–1139. URL: https://www.sciencedirect.com/science/article/pii/S1877050919306283. doi:https://doi.org/10.1016/j.procs.2019.04.161, the 10th International Conference on Ambient Systems, Networks and Technologies (ANT 2019) / The 2nd International Conference on Emerging Data and Industry 4.0 (EDI40 2019) / Affiliated Workshops.

[18] I. Harrando, P. Lisena, R. Troncy, Apples to apples: A systematic evaluation of topic models, in: RANLP, volume 260, 2021, pp. 488–498.

[19] I. Yamada, A. Asai, H. Shindo, H. Takeda, Y. Matsumoto, LUKE: Deep contextualized entity representations with entity-aware self-attention, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Online, 2020, pp. 6442–6454. URL: https://aclanthology.org/2020.emnlp-main.523.

[20] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. URL: https://aclanthology.org/N19-1423.

[21] M. Honnibal, I. Montani, S. Van Landeghem, A. Boyd, spaCy: Industrial-strength Natural Language Processing in Python, 2020. URL: https://doi.org/10.5281/zenodo.1212303.

[22] M. Grootendorst, Keybert: Minimal keyword extraction with bert., 2020. URL: https://doi.org/10.5281/zenodo.4461265.

[23] S. Rendle, Factorization machines, in: IEEE International Conference on Data Mining, 2010, pp. 995–1000.

[24] S. Y. Yu, S. Rokka Chhetri, A. Canedo, P. Goyal, M. A. A. Faruque, Pykg2vec: A python library for knowledge graph embedding, 2019.

[25] G. Ji, S. He, L. Xu, K. Liu, J. Zhao, Knowledge graph embedding via dynamic mapping matrix, in: ACL, 2015.