Beyond Metadata for BBC iPlayer: an autoencoder-driven approach for embeddings generation in content similarity recommendation

Simone Spaccarotella

September 2024

Contents

1 Introduction and background

I am a Software Engineer at the BBC, Team Lead for the Sounds web team, and I have been training as a Data Scientist, working in attachment with the iPlayer Recommendation team.

I built a machine learning model pipeline that produces content-to-content (C2C) similarity recommendations of video on-demand (VOD), for the "More Like This" section on BBC iPlayer [**?**]. This project is relevant to me because it is about recommendations, and I have been crossing paths with this world multiple times during my career at the BBC. This sparked an interest in it. I had a tangent encounter back in 2015, while working for a team that was building an initial recommender for BBC News and an API to provide recommendations using 3rd party engines. During a Hack Day some time later, I produced and presented a talk called "Recommendation Assumptions" [**?**], which was about type of recommendations and external factors affecting them, which are contextual to the consumption of the content itself. Until today, where I can finally put my knowledge into practice with an actual project on real data.

The BBC is a well-known British broadcaster, and it is always evolving to remain relevant to its audience. Its mission is to inform, educate and entertain, and it operates within the boundaries set by the Royal Charter [**?**]. The current media landscape requires the BBC to deliver digital-first content that is relevant to the audience, and this involves investements in data and personalised services, not to mention a certain revolution in machine learning that is keeping everyone busy.

2 Outline of the issue or opportunity and the business problem to be solved

The BBC produces and stores a vast amount of metadata for its content, and this metadata is surfaced by countless services and APIs. One of the priority for the BBC is to increae the adoption across the business of "Passport" [**?**], an internal service that generates, stores, manages and provides access to a richer dataset of metadata annotations for multi modal content (audio, video and text). The usage in production is very low, and its adoption would make the access to metadata consistent BBC-wide, removing duplications and reducing effort and costs.

The similarity score of the current C2C recommender is directly proportional to the number of values in common between any pairs of items on a per-feature basis. But the commonality is calculated with an exact string equality, hence it ignores any relationship between different categorical values expressing a similar concept (e.g. "comedy", "stand-up comedy").

The number and types of tags are not enough to sufficiently describe the content.

The data distribution is severely skewed towards the most popular category and no pre-processing is applied.

Lastly, each similarity score is multiplied by a hardcoded weight that modulates the importance of a feature, but it doesn't solve the polarising effect of a skewed distribution. Unfortunately, because they are hyperparameters and not learned weights, the model can't improve its performances by minimising them against a cost function.

To address these issues, the aim of this project was:

- **To improve the quality of the C2C similarity recommendations**. The hypothesis was that by using in input a richer set of metadata that better describes the content, and by reducing the high-dimensional data to a lower-dimensional latent manifold, the model is able to generate embeddings that can improve the performances of the model by mapping the item similarity problem to a geometric distance calculation between vectors in a multi-dimensional Euclidean space.

- **To reduce the costs to generate C2C similarity recommendations**. I sourced the input data from Passport, to build a BBC-wide general solution that could be used by any product and applied to any type of content, because they all share the same set of annotations.

- **To build a foundational item-embeddings generator**. Content-based recommenders use item metadata. This project provided an immediate solution for C2C unpersonalised recommendations that solely relies on them and it also provided a foundational approach for personalised recommender that combine content metadata with other information like user interactions and contextual data.

3 Methods used & justification

Methods used:

One-Hot Encoding to transform the categorical features into multiple numerical values. It's a simple yet effective method and It's perfect for the transformation of nominal categoricals, because it doesn't introduce any ranking and/or additive meaning to the encoded values. The downside is that it generates high-dimensional sparse arrays, introducing the so called curse of dimensionality problem into the modelling. This is an accepted drawback which is managed by the next step of the pipeline.

Undercomplete Autoencoder to compress the high-dimensional sparse array into a lower-dimensional dense space, to mitigate the curse of dimensionality. This space is represented by embeddings, which are multi-dimensional vectors in a dense format. The autoencoder is capable of capturing non-linearity in the data, learning latent key aspects of the input that improve the model performances during training. The learned embeddings, also help with the cosine similarity calculation, because one-hot encoded vector pairs have either a zero or a right angle, making the cosine respectively 1 or 0. For multi-hot encoded vector pairs (having multiple ones), angles can also assume a value in between 0 and 90, which is still limiting in terms of similarity, which we would ideally want on a continuous scale, bound between 0 and 1.

Cosine of angle

$$\theta$$

between vector embeddings, to calculate the item similarity.

Tools: Python as a programming language Pandas for dataframe manipulation. Each row of the table corresponds to an item, and each column to an annotation. Multi-hot, because more than one element of the array can have 1. Dask for out-of-core and out-of-memory operations. The iPlayer catalogue considered has roughly 1 year worht of data. Scikit-learn for dataset splitting and cosine similarity calculation TensorFlow and Keras for modelling the Neural Network matplotlib and seaborn for visualisation rdflib to fetch the RDF entity from the Ontology for visualisation purposes

Justification

The data in input is a list of JSON files containing the Passport tags for each item in the iPlayer catalogue. The annotations are categorical, so they must be transformed This set of tags is extrapolated into a dictionary data structure which is then transformed in a numerical table during a preprocessing phase where the rows represent the programmes and the columns the Passport tags. The numerical vectors are subsequently passed in input to

an Autoencoder model, that is trained to reconstruct the input with minimal loss. The encoder part of the learned weights can now be used to generate the embeddings given a set of Passport tags. These embeddings are finally used to calculate the content-to-content similarity between each pair of items, using the cosine as a scoring measure. The result is a matrix of scores that can be used to recommend the top K similar programmes, by filtering the seed programme, sorting the related scores in descending order and returning the top K elements.

A passport tag consists of a predicate/value pair, where the predicate is a string that describes the meaning of the tag (e.g. "about", "narrativeTheme", etc.) and the value is an RDF entity within the BBC Things Ontology. Each Passport tag has a zero-to-many relationships with the item, so I had to consider the dynamic nature of the input size. Also, the same entity value can appear in conjunction with different predicates for the same item. For example: "contributor:Jeremy Clarkson" and "about:Jeremy Clarkson" are two tags with the same value but with different meaning. So, I had to consider this case in the pre-processing phase, to keep the information intact.

For the pre-processing phase I used One-Hot Encoding. Each programme can be annotated with multiple tags, so it is more like a Multi-Hot encoding, where more than one element can be set to 1. This type of encoding may sound counterintuitive given that for high-cardinality data it generates a high-dimensional vector with all the "curse" that follows. But this is not an issue in this case, because the following step would take care of this.

For the embedding generation, I used an Encoder-Decoder ANN architecture, specifically, an Undercomplete Autoencoder model, built using TensorFlow and Keras. This model can learn latent features and compress the information in a smaller multidimensional vector of condensed numerical values for a more efficient processing. To do so, I extrapolate the Encoder portion of the trained Autoencoder. The input layer of this new model is the same as the Autoencoder, so it is fed with encoded Passport tags and the output layer correspond to the bottleneck layer of the Autoencoder which is how it is able to generate embeddings.

4 Scope of the project and Key Performance Indicators

5 Data selection, collection & pre-processing

6 Survey of potential alternatives

7 Implementation - performance metrics

8 Results

9 Discussion & conclusions/recommendations

10 Summary of findings

11 Implications

12 Caveats & limitations

13 Appendices

References

[1] BBC. Bluey - "more like this" tab. https://www.bbc.co.uk/iplayer/episodes/m000vbrk/bluey?seriesId=more-like-this.

[2] BBC. The Royal Charter. https://www.bbc.com/aboutthebbc/governance/charter, 2017. Valid until 31 December 2027.

[3] BBC. How metadata will drive content discovery for the BBC online. https://www.bbc.co.uk/webarchive/https%3A%2F%2Fwww.bbc.co.uk%2Fblogs%2Finternet%2Fentries%2Feacbb071-d471-4d85-ba9d-938c0c800d0b, 2020. This page was archived on 1st August 2023 and is no longer updated.

[4] Simone Spaccarotella. Recommendation assumptions. https://www.slideshare.net/slideshow/recommendations-assumptions/236291920, 2015. Presented at Prototyping Day @ Mozilla London on

3 September 2015, at Engineering Summit @ BBC on 7 March 2018, uploaded on SlideShare on 27 June 2020.