A Multi-Modal Approach to Movie Recommendation: Text and Image Synergy

Maureen Cooper, Lucía De Pineda, Anna Ramon

Pontificia Universidad Católica de Chile
Departamento de Ciencias de la Computación
IIC3633-Sistemas Recomendadores

In the contemporary era of an unprecedented access to a wide range of movies thanks to streaming platforms and online databases, users face the challenge of navigating an overwhelming number of options, making the selection of the right movie a huge challenge. This project aims to enhance user experience by developing a robust recommendation system, utilizing all the information available of movie characteristics (genre, ratings, poster images) and user opinions. The primary objective is to assess various recommendation models, with Bayesian Personalized Ranking and Alternating Least Squares serving as baseline models, using a user-item matrix based on their interactions. The study explores text-based models using BERT embeddings, image-based models with Convolutional Neural Networks, and multimodal models combining textual and visual information. The research provides valuable insights into the performance of recommendation systems, highlighting the strengths of different models across diverse data types. The central focus is on evaluating how user and item embeddings impact model performance.

Keywords: Recommendation System, embeddings, BERT, CNN, CLIP, user-item matrix

1 Introduction

Nowadays, we enjoy unprecedented access to a wide range of movies thanks to streaming platforms and online databases. Users are confronted with the overwhelming number of movie options available, making the selection of the right movie a huge challenge. To address this issue and enhance the user experience, the goal is to develop a highly effective recommendation system combining all the available data. Several factors have accentuated this issue. The proliferation of streaming platforms like Netflix, Disney+, Amazon Prime, and HBO has greatly expanded the pool of accessible films. Consequently, users are now confronted with such a volume of choices that making a decision can be quite unnerving. Furthermore, the explosion of data has incremented the importance of addressing this problem. As we collect vast amounts of data from users and develop user profiles, the potential for providing accurate recommendations becomes evident. This data includes users' preferences and reviews, films' characteristics and their posters, enabling the development of robust recommendation systems. It's crucial to recognize that users' behavior evolves over time, and they expect recommendation systems to adapt and capture these evolving preferences and trends. In today's landscape is very important to face this problem. The development of an accurate recommendation system can improve users' experience and let them discover films that adapt to their preferences, increasing their satisfaction degree and engaging them to the streaming platform. Due to the competitive market, it is important for the streaming plataforms to get as much users as possible. The tendency to personalized recommendations is becoming very relevant, adapting them to the users' preferences. The challenge of developing a movie recommendation system is highly relevant in the current context due to the shift in consumer behavior, the availability of a lot of data, and the need to enhance the user experience in a highly competitive entertainment environment

2 Dataset

In this section, we will perform an exploratory analysis of the IMDB Vision and NLP dataset, which consists of textual reviews on movies and their corresponding poster JPG images. A deep understanding of the dataset is crucial to build a successful movie recommendation system that truly meets the needs and preferences of users. We will start by providing an overview of the dataset's structure and key components.

2.1 Dataset Overview

The IMDB Vision and NLP dataset is an extensive collection of data, combining both textual and visual information, making it an ideal resource for our project. It is divided into two main files: *reviews.csv* and *movies.csv*. The specifics of each file are:

Reviews Dataset:

The *reviews.csv* file contains the main source of information, including:

- **Reviews**: It includes 541,811 reviews contributed by 12,088 reviewers for a total of 4,067 movies. Each review is associated with various attributes, making it a valuable source of data for our analysis.
- **Reviewers**: There are 12,088 reviewers, with a mean of 44.88 reviews per user and a standard deviation of 118.34. The user with most reviews reaches 3301 interactions, while the one with less has only one.
- **Movies**: The reviews are about 4067 different movies. Each movie has a mean of 133.38 interactions with a standard deviation of 95.17. The movie with more interactions has 889, while the one with less has 51, which is a relatively high number.
- **Rating**: Reviews are rated on a scale from 1 to 10, allowing for a diverse range of user preferences and feedback. However, there are 76,800 missing values, that is 76,800 reviews with no rating, these will be handled in the next section. Moreover, the average rating is 6.96, with a standard deviation of 2.31, which means users tend to rate high the movies.
- **Review Summary**: The review summary attribute contains brief but impactful feedback from reviewers. It reflects common themes and emotional tones, providing valuable insights for building movie recommendations.
- **Review Date**: Each review includes a timestamp, providing a temporal dimension to the dataset. The review date statistics reveal that the dataset consists of a total of 542,461 review dates, with 8,195 unique dates. The most frequently occurring date is '2020-03-30', appearing 362 times. The dataset spans from its earliest review date on '1998-07-28' to its most recent review date on '2021-01-08'.
- **Spoiler Tag**: A binary spoiler tag (0 or 1) indicates whether the review contains spoilers, which may have an impact on movie recommendations. We can also use this information to warn users or provide options for spoiler-free content. In the dataset, there are 411575 reviews without spoilers and 130886 that do contain spoilers.
- **Review Detail**: The content of the review itself, which holds valuable textual information. This allows us to gain a deep understanding of the reasons behind a reviewer's rating and opinion about a movie. The content in Review Detail plays a significant role in improving our recommendation system, as we can use the textual feedback to provide more personalized and context-aware recommendations.
- **Helpfulness Tag**: A helpfulness tag provides insights into how other users perceive the review's value and it is a measure of the quality and usefulness of reviews. The values, represented as ['x', 'y'], show the helpfulness of reviews, with 'x' denoting the number of users who found a review helpful out of 'y' total users who provided helpfulness ratings. The dataset reveals a diverse distribution of these tags, ranging from ['0', '0'] for reviews that received no helpful ratings (81,993 reviews) to ['1', '2'] (46,194 reviews), ['1', '1'] (32,690

reviews), and ['1', '3'] (26,254 reviews), among others.

Movies Dataset:

The *movies.csv* file complements the reviews dataset and consists of 4168 rows, representing 4071 unique movies. The duplicates processing will be detailed in the next section. This dataset includes the following attributes:

- **IMDB ID**: A unique identifier for each movie review on IMDB. There are 4071 different IMDB IDs.
- **IMDB Link**: A link to the movie's IMDB page, which can serve as a reference for additional information.
- **Title**: The movie's title, also serving as a unique identifier. There are four titles that appear twice, with two different IMDB IDs. This is because of two reasons: either they have two different IMDB reviews linked to that movie, with different scores and image posters, or there are two different movies with the same title.
- **IMDB Score**: The movie's IMDB score, which reflects its overall popularity and reception. The average score is 6.71, with a standard deviation of 1.00. Also, it is important to note that the maximum score given is 9.00, while the minimum is 1.8.
- **Genre**: Categorization of the movie's genre, allowing for genre-based analysis and categorizing it into relevant themes and styles. The most common genres are drama (1929 movies), comedy (1283 movies) and action (976 movies).
- **Image Path**: A path to the movie's poster JPG image, linking the visual modality to the dataset.

2.2 Dataset Preprocessing

Following our first exploratory data analysis, we identified certain aspects of the data that require preprocessing to improve the dataset's quality and prepare it for further analyses. The steps followed are:

- **Removing Duplicate Entries:** The first preprocessing step involves removing the duplicate entries in both datasets. Regarding the movies dataset, *movies.csv*, it has been found to contain 4168 rows representing 4071 unique movies, indicating the presence of duplicate movie records. In the reviews dataset, *reviews.csv*, there are 645 duplicate rows. In this step, we will identify and remove these duplicate entries, leaving us with a clean dataset containing unique movies and reviews only.
- **Handling Missing Values in Ratings:** The next step involves handling missing values in the "Rating" attribute of the reviews dataset. We identified 76,800 missing values, indicating 76,800 reviews with no assigned ratings. To address this, we will impute these missing ratings with appropriate values to ensure that all reviews are included in our analysis, and no valuable information is lost. The strategy followed is to use the average rating for each movie by the reviewers to fill in missing ratings.
- **Text and Image Preprocessing:** Our dataset contains both textual reviews and movie poster images. To extract meaningful insights from these modalities, we will carry out specialized processing for each:
  - **Textual Data:** Textual reviews will undergo a series of cleaning and feature engineering processes. We'll also create numerical representations of text using embedding techniques to allow a deeper text-based analysis. These preprocessing steps will be further detailed in the "Text-Based Modeling" section, Section **??**.
  - **Image Data:** Movie poster images will be loaded, resized, and normalized for consistency. Data augmentation techniques will be applied to diversify the dataset. All of this will prepare the images for our image-based modeling. The image preprocessing details will be provided in the "Image-Based Modeling" section, Section **??**.

The successful execution of these preprocessing steps will be essential for the modeling phase, which is carried out in the next sections of our report.

3 Methodology

To assess the effectiveness of our recommendation models, we will employ a range of evaluation metrics:

- **NDCG (Normalized Discounted Cumulative Gain)**: NDCG will evaluate the quality of the recommendations, particularly in the case of multimodal recommendations, which aim to provide users with more relevant content.
- **Precision and Recall**: These metrics will help us understand how well the recommendations align with user interactions and preferences.
- **MAP (Mean Average Precision)**: Average precision of all the recommendations.
- **AUC (Area Under the Curve"**: Area under the curve ROC (Receiver Operating Characteristic).

3.1 Baseline Models

In order to be able to properly evaluate the performance of a recommendation system, it is essential to establish baseline models. In this section, we will evaluate the performance of traditional and well-known models, such as BPR and ALS. This evaluation will serve as a valuable reference for comparing the results obtained from the more complex models that will be developed in the following sections.

To evaluate the model's performance it is important to split the data into train and test sets. The training set is used in the training model phase where the model tries to learn as much as possible from the given dataset with the goal to generalize the train data behaviour. Once the model has gathered all the possible knowledge from training data, it is important to evaluate the performance in the test set (unseen data). It is very relevant to split data in a way that both datasets are balanced, otherwise, the knowledge that the model gets from the training set will not be useful for the test set and the resulting recommendations will not be accurate. The *traintestsplit* function from *sklearn* library allows us to split the dataset in a balanced way taking into account the rating value. We want to have similar number of relevant movies and unrelevant movies in each set, for training we use the 80% of the instances and for testing the other 20%.

3.1.1 BPR

Using BPR as a baseline model allows us to compare the performance of our future advanced models against a simple but strong reference point. This helps in assessing whether the complexity of a new model is justified by a significant improvement in recommendations.

Here we will be using BPR matrix factorization as a collaborative filtering recommendation system by leveraging user-item interactions to make recommendations. The users had interacted with some movies rating them. For this section, we will only consider those positive interactions (movies rated with a 7 or higher). Our goal is to take into account the movies that the user really liked. Eventhough BPR can be used as an implicit feedback recommendation system, we will not be using it in that way because we have knowledge of which interactions are positive and those ones that are negative.

BPR begins with matrix factorization, defining the user-item interaction matrix, where rows represent users, columns represent items, and the values represent user-item positive interactions. BPR aims to factorize this matrix into two lower-dimensional matrices: one for users and one for items. The idea is to learn latent factors that capture user and item characteristics. In order to determine the best number of latent factors we fit some models, each with a different number of latent factors. The goal is to find the best model in terms of error metrics. The metrics that we will consider in this section are: AUC, ndcg10 and the MAP. The number of latent factors evaluated are: 50,100,200,500 and 1000. The ndcg10 and MAP values were almost the same for each model, all around **map: 0.0173** and **ndcg@10: 0.0436**, so we used the accuracy value to determine the best model. The library *implicit* contains a function for training a BPR model which also calculates the train accuracy.

Nº Latent Factors	AUC Value
50	92.36%
100	92.51%
200	92.03%
500	91.22%
10000	90.50%

Table 1: Training Set's AUC

The optimal number of latent factors is 100. The next step is evaluate the model's capacity of generalization, so we train a BPR model with 100 latent factors and evaluate the performance in the test set. The results obtained are:

MAP	**ndcg10**	**AUC**
0.0103	0.0241	86.61%

Table 2: Test Set performance

The results do not show a model capable of capturing the data set behaviour, actually, the metrics are very low in precision terms.

3.1.2 ALS

Alternating Least Squares (ALS) is another matrix factorization technique used in collaborative filtering. Unlike BPR, which optimizes a pairwise ranking loss function, ALS minimizes the squared error between the observed ratings and the product of the latent factor matrices for users and items, under a regularization term that prevents overfitting. ALS alternates between fixing the user matrix to solve for the item matrix and vice versa, hence the name.

In our implementation, we utilize the ALS algorithm to model user preferences and item characteristics from the IMDB Vision and NLP dataset. We filtered the dataset to focus on positive interactions, similar to our approach with BPR. For ALS, the ratings are treated as confidence scores, with higher ratings indicating a higher confidence in user preferences. We set the model to factorize the user-item interaction matrix into latent factors, iterating over user and item matrices sequentially to minimize the regularized least squares error.

To find the optimal number of latent factors, we trained multiple ALS models with varying dimensions. The number of factors explored were 50, 100, 200, 500, and 1000. We selected the model with the best performance on the train set based on the Mean Average Precision (MAP) and Normalized Discounted Cumulative Gain (NDCG@10), which corresponds to the one with 200 latent factors.

The results obtained from the best ALS model are as follows:

MAP	**ndcg10**
0.0724	0.1157%

Table 3: Test Set performance

The results indicate that the ALS model has been effective in capturing user preferences and providing meaningful recommendations. While the MAP suggests there is room for improvement, the NDCG@10 score shows that the top-ranked recommendations are of high relevance. Compared to BPR, the ALS model shows an increase in NDCG@10, highlighting its strength in identifying top-ranked items which are critical in a recommendation system.

To further improve the ALS model, future work could explore hybrid models that incorporate both textual and visual features of the movies to enrich the recommendations. Additionally, parameter tuning and incorporating more sophisticated forms of user feedback into the model could enhance its predictive performance.

3.2 Text-based Models

In this section, we will explore the incorporation of state-of-the-art text-based models, such as BERT, into our recommendation system. BERT, introduced in Devlin et al. (2018), has demonstrated remarkable performance in various text-related tasks, and we will investigate how its integration enhances our current recommendation methods. Here, we will focus on a comprehensive analysis of movie reviews, using the power of language understanding.

The main goal here is representing all the reviews as a vector and using this representations for the users' and movies' embeddings. Firstly, the BertForMaskedLM is downloaded. This pretrained model is able to capture the syntactic and semantic word's relationships, so it is useful for representing the review context and finding similar reviews. All the reviews are tokenized and then modeled as a vector. Secondly, for determining an embedding for each user and movie, the vector representations of each review and the given rating are associated to its user and movie. The embedding representation is the mean of all the reviews' representations and its scaled rating, associated to each user and movie. For retrieving recommendations for a given user it is used the cosine similarity. The methodology used is computing the cosine similarity between the user embedding and all the movie embeddings, the movies with higher score are the ones recommended.

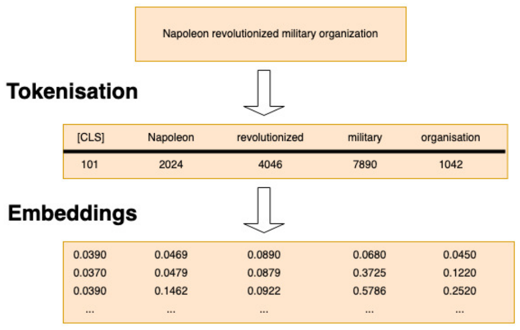

Figure 1: Example of how BERT tokenises and transforms to an embedding a sentence'

The described approach incorporates elements of collaborative filtering, specifically in the way it associates vector representations of reviews and ratings with users and movies. While collaborative filtering is a central component, the reliance on vector representations and the utilization of the pre-trained model BertForMaskedLM for review analysis, also introduce elements of content-based filtering. This hybrid approach is often referred to as content-based collaborative filtering. It combines collaborative and content-based methods to enhance recommendation performance. The metrics obtained using BERT embeddings are:

MAP	**PR**	**RC**
0.1187	0.2785	0.2617

Table 4: Performance of BERT embeddings

3.3 Image-based Models

This section analyses image-based modeling using Convolutional Neural Networks (CNNs). CNNs have long been the standard choice for image processing tasks. Taking advantage on the visual data of our dataset, the movie posters, we will assess how integrating CNNs can be a valuable addition to our recommendation system.

The chosen architecture, ResNet50, has proven to be highly effective in image-related tasks due to its deep and sophisticated design. ResNet50 (He et al. (2015)), short for Residual Network with 50 layers, is a renowned CNN architecture known for its deep structure and the introduction of residual connections. These residual connections allow for the training of very deep networks without encountering issues like vanishing gradients. ResNet50 has demonstrated state-of-the-art performance on various image classification tasks, making it a natural choice for extracting intricate visual features from movie posters.

The methodology followed for movie recommendation using the extracted visual features is the following:

1. Feature Extraction from Movie Posters: The process begins by loading and preprocessing movie posters using the ResNet50 model. This involves resizing the images to a standard size and normalizing the pixel values. The pre-trained ResNet50 model is then applied to these posters, extracting high-level features that capture intricate patterns and visual details.

2. Movie Poster Embeddings: The extracted features serve as the foundation for creating embeddings that contain the visual characteristics of each movie. These embeddings are representations of the posters in a feature space, capturing the unique visual elements that distinguish one movie from another.

3. Collaborative Filtering with Visual Embeddings: Similar to the collaborative filtering approach in the text-based models section, the visual embeddings are associated with users and movies. The embeddings for users are computed by aggregating the visual features of the movies they have rated higher than 7, weighted by their ratings. This collaborative filtering approach ensures that the visual preferences of users are reflected in their embeddings.

4. Personalized Recommendations using Cosine Similarity: To provide personalized recommendations, cosine similarity is employed. For a given user, the system calculates the cosine similarity between the user's embedding and the embeddings of all movies. Movies with higher similarity scores are recommended, aligning with the user's visual preferences.

The metrics obtained using ResNet-50 embeddings are:

MAP	PR	RC
0.0132	0.0130	0.0006

Table 5: Performance of ResNet-50 embeddings

3.4 Multimodal Models

In recent developments, multi-modal models like CLIP (Contrastive Language–Image Pretraining) have gained prominence, offering a robust way to fuse textual and visual information. Unlike traditional models that separately handle text (like BERT) and images (like ResNet-50), CLIP is designed to understand and relate both modalities, making it ideal for tasks that require a comprehensive understanding of both text and images.

The power of CLIP lies in its ability to learn visual concepts from natural language supervision, thus creating a shared representation space for both images and texts. In the context of a movie recommender system, this capability is leveraged to process and relate movie posters (visual content) and user reviews (textual content).

Here's how CLIP is utilized in our recommender system:

Data Processing: Movie posters are preprocessed using CLIP's image processing pipeline, while user reviews are tokenized and truncated to a manageable size using CLIP's text processing capabilities. Embedding Generation: CLIP generates embeddings for both images (posters) and texts (reviews). These embeddings capture the nuanced features and semantics of both modalities. Embedding Aggregation and Similarity Calculation: The system calculates the average of image and text embeddings to create a unified representation of movies. It then measures the similarity between a user's review embeddings and these unified movie embeddings using cosine similarity. Recommendation Generation: Based on similarity scores, the system recommends movies that align with the user's preferences as indicated by their reviews and perceived interests from the visual content of the movies they have engaged with. This approach signifies a significant shift from traditional methods, combining the strengths of text-based and image-based models, and leveraging the synergy between textual and visual information to provide more accurate, personalized recommendations.

By integrating CLIP, the recommender system should not only understands textual content but also gains insights from visual elements, leading to a richer, more nuanced recommendation process. This multi-modal strategy is especially effective in domains like movie recommendations, where both visual aesthetics and narrative content play crucial roles in user preferences. The metrics obtained using CLIP are:

MAP	PR	RC
0.0134	0.0145	0.0237

Table 6: Performance of Clip embeddings

Radford et al. (2021)

4 Results

Overall, the summary of the results obtained are:

	MAP	PR	RC
BERT	0.1187	0.2785	0.2617
ResNet-50	0.0132	0.0130	0.0006
CLIP	0.0134	0.0145	0.0237

Table 7: Summary of models' performance

From the results in the table, it's evident that the text-based model (BERT) significantly outperforms the models utilizing image embeddings (ResNet-50 and CLIP) in generating recommendations. This superior performance of BERT can be attributed to the nature of the images used: movie posters, which may not always accurately reflect the movie's content or themes. While it was anticipated that the multimodal CLIP model would surpass the image-only ResNet-50 model due to its integration of more comprehensive information in the embeddings, CLIP's performance fell short of expectations. This underperformance might be linked to its pre-training focus on object classification, which may not align well with the nuances required for effective movie recommendation based on posters.

5 Conclusions

A key insight from our experiments is the realization that more complex or sophisticated models, like CLIP, do not automatically guarantee superior results. This underscores the importance of a deep understanding of the dataset at hand when selecting the most appropriate model for a given scenario. However, an intriguing direction for future exploration involves diversifying the application of multi-modal models. For instance, we could experiment with user-provided images or leverage the CLIP model for more nuanced tasks such as poster classification, moving beyond mere feature extraction.

Looking ahead, a promising avenue for further research would be to integrate the most effective elements from each model tested. This hybrid approach aims to tailor a model specifically optimized for this dataset, potentially enhancing its performance significantly. Additionally, incorporating user feedback and interaction data could offer a more dynamic and personalized recommendation experience. Understanding the context in which recommendations are made, and considering factors such as temporal dynamics and user preferences, could also play a pivotal role in refining the model's accuracy and relevance.

In conclusion, while the sophistication of a model is a valuable asset, its alignment with the dataset's unique characteristics and the specific needs of the recommendation scenario is paramount. Our journey into the realm of recommender systems is just beginning, and these findings pave the way for more nuanced and effective approaches in the future.

References

Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2018). BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, *abs/1810.04805*. Retrieved from http://arxiv.org/abs/1810.04805

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep residual learning for image recognition. *CoRR*, *abs/1512.03385*. Retrieved from http://arxiv.org/abs/1512.03385

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., . . . Sutskever, I. (2021). Learning transferable visual models from natural language supervision. *CoRR*, *abs/2103.00020*. Retrieved from https://arxiv.org/abs/2103.00020