Collaborative Filtering versus Multi-modal Network for Movie Genres Classification

Nguyen Dieu Nhat
21020656
dieunhat@gmail.com

Le Hoang Minh Ha
21020621
21020621@vnu.edu.vn

Vu Bao Chau
21020460
21020460@vnu.edu.vn

Abstract

Movie genre classification is a crucial task that has found applications in various fields. Can be treated as an *Extreme Multi-label Classification* (XMC) task with long-tailed data, this problem is not only valuable in real-world domains, such as for recommendation systems, but also leave space for researches on XMC. In this experiment, we implement two different approach for this problem: Rating-based Collaborative Filtering and Label Correlation-based Multi-modal Network. We conduct our methods on the given *MovieLens* dataset, achieving the best macro F1-score of 0.56.

Keywords: movie genres classification, long-tail data, multi-modal, multi-label learning, class imbalanced, label correlation, rating-based

1 Introduction

Movie genre classification is a fundamental task in machine learning that has widespread applications in various domains, including recommendation systems and personalized user experiences. Accurate classification of movie genres enables effective organization and targeted recommendations for users, enhancing their movie-watching experiences. This study aims to classify movie genres by the movie's title, poster, and user's input rating.

In our experiment, we propose two distinct methods for movie genre classification on the MovieLens (Harper and Konstan, 2015) dataset: Collaborative Filtering with k-Nearest Neighbors (k-NN) using movie ratings features, and a Multi-modal model utilizing features such as movie titles and posters. By exploring these two approaches, we aim to compare their effectiveness and identify the most suitable method for genre prediction. Both approaches use only the provided dataset and no additional data.

Traditionally, the Collaborative Filtering (Herlocker et al., 2000) approach leverages the power of user ratings data to make personalized recommendations. By calculating the similarity or weight between users or items based on their ratings, collaborative filtering techniques can predict user preferences and suggest movies that align with their tastes. In our method, we utilize k-NN, a popular algorithm in collaborative filtering, to identify movies with similar ratings patterns and make genre predictions based on this similarity.

In contrast, the Multi-modal model takes advantage of both textual and visual information to classify movie genres. Our approach combines the power of the XLNet language model for text analysis and the VGG-16 convolutional neural network architecture for image analysis, aiming to improve the accuracy and effectiveness of genre classification.

The XLNet language model is a state-of-the-art model known for its ability to learn contextual representations from text. By analyzing the textual content of movie titles, we can capture important semantic features that contribute to genre classification. XLNet's contextual understanding allows it to capture subtle nuances and relationships within the movie titles, enhancing the classification accuracy.

In addition to textual information, we incorporate visual cues by employing the VGG-16 convolutional neural network architecture. VGG-16 is a powerful model that has demonstrated exceptional performance in image classification tasks. By extracting features from movie posters using VGG-16, we can capture visual patterns and characteristics that are indicative of specific genres.

Our presentation video can be accessed by the following url: https://drive.google.com/drive/folders/1mc7szTQo_xN-C4SFYjLXTG1l1bK-akmz?usp=sharing

2 Exploration & Data Analysis

The dataset used for this task was extracted from the MovieLens 1M dataset[1], including a poster image folder and 3 tables: users, ratings and movies with the movies splitted into training and testing sets.

In *users* table, each entry represents one user, and has the following format: *(userId, gender, age, occupation, zip)*. This table contains 6040 users with 7 age groups: under 18, 18-24, 25-34, 35-44, 45-49, 50-56 and older.

In *ratings* table, each entry represents one rating of one movie by one user, and has the following format: *(userId, movieId, rating, timestamp)*. This table contains 1,000,209 ratings for 3706 movies by 6040 users. Ratings are made in a 5-star scale with 1 star increments.

In *movie*s table, each entry represents one movie, and has the following format: *(movieId, title, genres)*. This table comprises 3106 movies. Movie titles include the year of release. Genres are a pipe-separated list, and are selected from the list of genres shown in Table 1.

Action	Adventure	Animation
Children's	Comedy	Crime
Documentary	Drama	Fantasy
Film-Noir	Horror	Musical
Mystery	Romance	Sci-fi
Thriller	War	Western

Table 1: List of genres

After reviewing the dataset, we noticed that there are a total of 177 movies that do not have any ratings, which is 143 and 34 movies in the training set and test set, respectively. Movies without ratings as well as ratings for movies that are not in the dataset will be eliminated from training, there are only 817,424 ratings for 2963 movies left to train in the dataset.

Figure 1 shows the distribution of genres in the training set. With the dataset having all other categories with relatively fewer occurrences compared to Drama and Comedy making this a long-tailed label set. Figure 2 shows the correlation matrix of genres. Since the dataset have an unbalance number of movies for each genre, the matrix appears to be uneven.

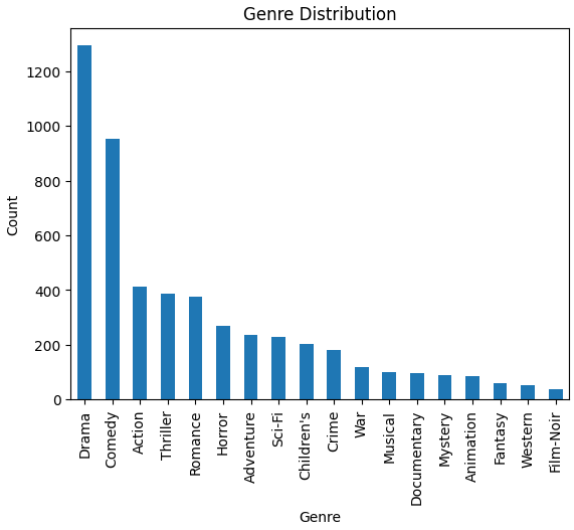

Figure 1: Illustration of genres distribution by movies.

Ultimately, we conclude that there are three critical challenges of the dataset that needed to be handled:

- **Long tail data distribution**
- **Missing modalities**
- **Data scarcity**

In our experiment, the first problem with imbalanced long-tailed data will be resolved by grouping approach based on label correlations, incomplete data types are ramified to be predicted separately and transfer learning is implemented to handle data hunger.

3 Methods

3.1 User-based Collaborative Filtering Approach

Collaborative Filtering

Collaborative Filtering (CF) is a popular technique in recommendation systems that uses algorithms to filter data from user reviews to make personalized recommendations, such as movies to watch, books to read, or products to buy. Memory-based CF methods (Yu et al., 2004) use the saved user's rating data from database to calculate the similarity or weight between users or items, and make predictions or recommendations according to those calculated similarity values (Su and Khoshgoftaar, 2009). These methods are widely and notably deployed into large commercial systems such as Amazon[2]

[1] https://grouplens.org/datasets/movielens/1m/

[2] www.amazon.com/

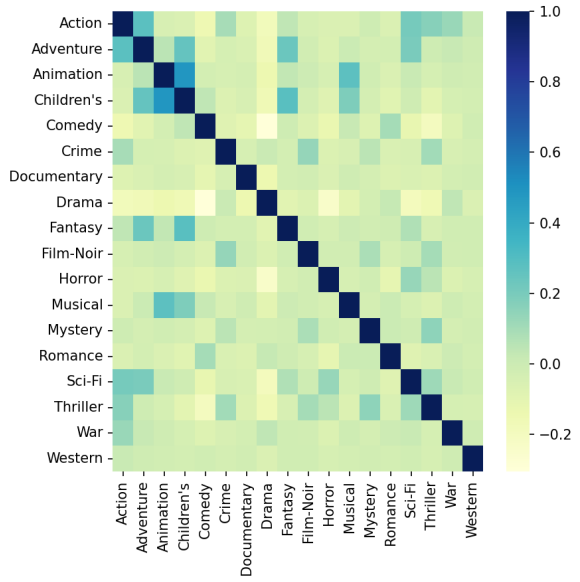

Figure 2: Illustration of the correlation matrix between movie genres based on their co-occurrence.

(Linden et al., 2003) and GroupLens[3] (Konstan et al., 1997), because they are highly effective and easy to implement. (Hwang et al., 2016) employed CF to predict the unknown rating of a given user for a movie using only the information of similar users ratings on the same movie.

Instead of using movie-based information like textual (such as movie title, description), or audio-visual (such as movie trailer, poster) information, PFEFFER (2020) uses a completely different type of data - the user ratings of the movies using CF. This approach is based on the assumption that a user may have favorite genres, and they will more likely to watch and give movies of those certain genres high ratings. Which means, movies of the same genres are more likely to be rated, and to be rated with similar rating scores, by a set of users who prefer that genres.

For each pair of movies *in the training set* that have ratings, we calculate our movies similarity measure using the product of the Pearson correlation coefficient of the ratings given to two movies, and the Jaccard index of common users. We then construct a network with the movies as nodes, and weighted edges representing the similarity between the movies. The movies that don't have any ratings will be excluded from our network.

Pearson correlation (Sedgwick, 2012), also known as the Pearson Product Moment Correlation (PPMC), is a statistical measure that quantifies the strength and direction of the linear relationship between two continuous variables. The Pearson correlation coefficient is a number between -1 and 1. A value of -1 indicates a total negative linear correlation, 0 indicates no correlation, and $+1$ indicates a total positive correlation. The Pearson correlation coefficient ρ of the ratings given to two movies i and j by the same users is defined as

$$\rho_{ij} = \frac{\sum_{u \in U_{ij}} (r_{ui} - \bar{r}_i)(r_{uj} - \bar{r}_j)}{\sqrt{\sum_{u \in U_{ij}} (r_{ui} - \bar{r}_i)^2} \sqrt{\sum_{u \in U_{ij}} (r_{uj} - \bar{r}_j)^2}},$$

where U_{ij} is the set of users who have rated both movies i and j, and $\bar{r}_i = \frac{1}{|U_m|} \sum_{u \in Ui} r_{ui}$ is the average rating of movie i.

Jaccard index, also known as the Jaccard similarity coefficient, is a measure of similarity between two sets. It calculates the ratio of the size of the intersection of the sets to the size of their union. The Jaccard index of common users, which is the fraction of users who rated both i and j to all users who rated i and/or j, is defined as:

$$\phi_{ij} = \frac{|U_{ij}|}{|U_i \cup U_j|}.$$

Let $S_{i,j}$ be the product measure which denote the similarity between the nodes i and j in our movies network, which defined as the product of the Pearson correlation coefficient and the Jaccard index of common users:

$$S_{i,j} = \rho_{ij}.\phi_{ij}$$

k-Nearest Neighbors

In order to classify a movie's genres in the test set, we use k-Nearest Neighbors (k-NN) to leverages the similarity between movies based mainly on users ratings, by considering the genres of the nearest neighbors (Figure 3). k-NN works by finding the k closest data points in the training set to a new input point and making predictions based on their labels or values. We will cluster a neighborhood of movies for each rated movie in the test, using not the common distance formulas like Cosine or Euclidean, but a distance based on our similarity measure $S_{i,j}$ instead. Let $D_{i,j}$ be the distance between node i and j:

$$D_{ij} = 1 - S_{ij}$$

[3] https://grouplens.org/

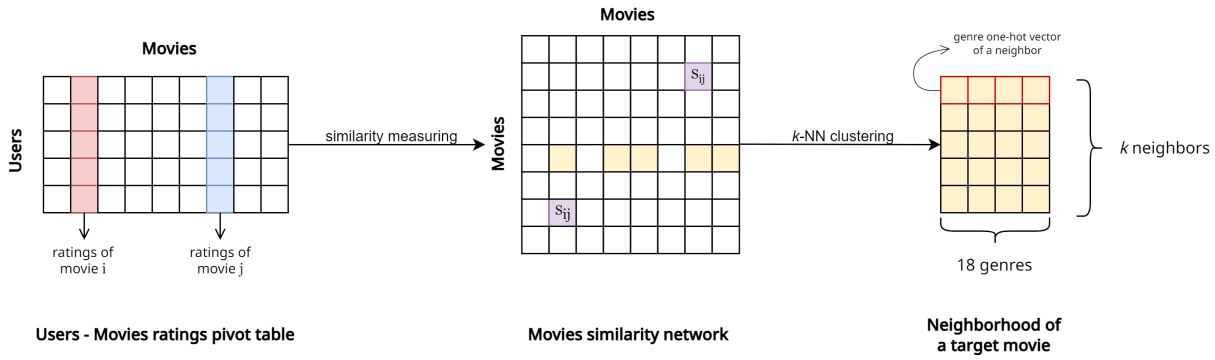

Figure 3: Illustration of the neighborhood clustering process from movies ratings.

Classifier

To classify genres for the target movie based on its neighborhood, and since this is a multi-label classification problem, we will use each neighbor's similarity score to the target movie and the neighbor's genres to calculate the confidence score of the target movie for each candidate genre, then use a threshold to classify the most relevant genres to a target movie (Figure 4).

The confidence score of the movie m with respect to a candidate genre is given by:

$$s_{m,g} = \frac{1}{|N(m)|} \sum_{n \in N(m)} \gamma(n,g).S_{m,n}$$

where $N(m)$ is the set of neighbors of m, $s_{m,n}$ is the similarity of m and n, and $\gamma(n,g)$ indicates whether g is a genre of n; that is, $\gamma(n,g) = 1$ if $g \in G(n)$, 0 otherwise. In other words, the target movie's confidence scores vector is the weighted average of *k* genres one-hot vectors of *k* neighbors in the neighborhood, with the similarity score of each neighbor to the target as weight.

After calculating the confidence score of a movie to each genre, we thresholding the top t_m labels with the highest confidence scores, where t_m is given by a technique called Parametric Adaptive Rank Cut to determine whether a genre is relevant to the target movie, which was proposed by PFEFFER (2020):

$$t_m = round(\alpha.\overline{G_N(m)})$$

where $N(m)$ is the set of neighbors of m, and $\overline{G_N(m)}$ is the average number of genres of the movies in $N(m)$:

$$\overline{G_N(m)} = \frac{1}{|N(m)|} \sum_{n \in N(m)} |G_n|$$

where G_n is the set of genres of movie n.

This collaborative filtering method using the known preferences of users to cluster movies by their ratings and common users, then using the cluster members genres to predict a movie genre that fits in the cluster. In our experiment, this approach is the best effective on this given dataset, out-perform our multi-modal neural network by a very large margin. However, given a completely new movie with only its title, poster image, and no ratings from users in the existing database, this method will fail to classify the target movie genres. For better generalization, our collaborative filtering method will be considered only an experiment for better understanding about machine learning and recommendation systems, not our main model for this task.

3.2 Multi-modal Approach

Multi-label learning deals with the classification of instances into multiple categories simultaneously. Unlike traditional single-label classification, multi-label classification recognizes that instances can belong to multiple classes simultaneously. Multi-modal learning, on the other hand, refers to the integration of information from multiple modalities, such as text, images, audio, and video, to enhance the performance of learning algorithms. By leveraging complementary cues from different modalities, multi-modal learning enables a more comprehensive understanding of complex data.

Label Correlation

Multi-label classification with long-tailed data distribution provokes the problem of samples imbalance between different classes which reduces the model's performance. We decide to divide labels into 8 different groups based on their correlation calculated in the training set. Setting the threshold

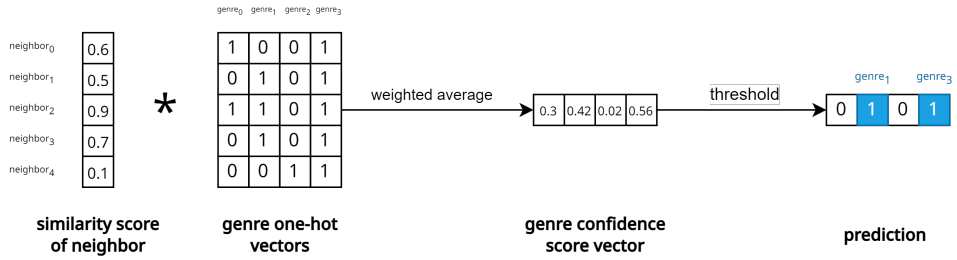

Figure 4: Illustration of the genres classification process for a target movie using its neighborhood.

of 7%, tail classes contributing less than this margin will be grouped to the remaining head classes based on the correlation score below (Figure 5). After re-sampling, the distribution of 8 mixed groups will have a relatively more balanced distribution.

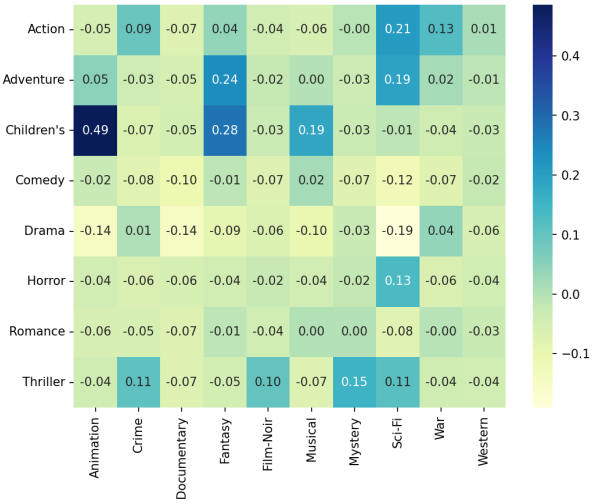

Figure 5: Illustration of the correlation between head classes (y-axis) and tail classes (x-axis).

Text Feature Extraction

XLNet (Yang et al., 2019) is an AutoRegressive Transformer (Vaswani et al., 2017) that leverages the best of both AutoRegressive (AR) Language Modeling (Radford et al., 2018) (Radford et al., 2019) and AutoEncoding (AE) (Devlin et al., 2018) while attempting to avoid their limitations. AR language model is a kind of model that uses the context word to predict the next word, where the context word is constrained to two directions, either forward or backward. Unlike the AR language model, an AE language model aims to reconstruct the original data from corrupted input, and it can see the context on both forward and backward direction, but assumes the predicted (masked) tokens are independent of each other given the unmasked tokens.

Instead of using a fixed forward or backward factorization order as in conventional autoregressive models, XLNet maximizes the expected log likelihood of a sequence with all possible permutations of the factorization order. Thanks to the permutation operation, the context for each position can consist of tokens from both left and right. In expectation, each position learns to utilize contextual information from all positions, i.e., capturing bidirectional context. Additionally, inspired by the latest advancements in autogressive language modeling, XLNet integrates the segment recurrence mechanism and relative encoding scheme of Transformer-XL (Dai et al., 2019) into pretraining, which empirically improves the performance especially for tasks involving a longer text sequence.

To extract textual feature from the movie title, we use the pretrained `xlnet-base-cased` version of XLNet, which was retrieved from Huggingface[4].

Image Feature Extraction

Visual Geometry Group Network (VGG-Net) (Simonyan and Zisserman, 2014) is a well-known deep Convolutional Neural Network (CNN) architecture consists of multiple convolutional layers and is known for its simplicity and effectiveness in image classification tasks.

In our experiment, we choose to use VGG-16, consisting of 16 layers pretrained on ImageNet dataset. To align with our multi-modal task, we replace the last fully-connected layer of 1000 units by 3 fully-connected layers of 1024, 128 and 64, followed with ReLU activation and Dropout layer with probability of 0.2 respectively, with 64 is the hidden size for future fusion with textual features.

Fusion

The choices of fusion timing can vary from early fusion, mid-fusion and late fusion. Our pipeline adopt late fusion approach, utilizing concatenate

[4] https://huggingface.co/xlnet-base-cased

operation to merge the previous embed textual and visual features. The purpose of this fusion operation is to combine the information captured by both title and poster of the movies to predict the final label. The fused units will therefore be passed through a linear transformation layer, resulting in 8 units representing prediction score for each label. A sigmoid activation function will then be applied to get the probability score ranging from 0 to 1:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

4 Experiments

4.1 Data processing

Image Processing. The dataset has 3106 movies in the train set and 777 movies in the test set, in which only 2602 and 654 movies in each set respectively have poster. Movies without posters contribute 16% of the training data, therefore, instead of excluding them from the training set, we choose to fill in the poster of movies having the exact same genres as to maintain the information from their titles. If that incomplete movie has no matching instances based on genres, it will then be removed.

We normalize all images to Resnet mean and standard deviation (std). For the original ResNet models trained on the ImageNet dataset, the mean values are typically `[0.485, 0.456, 0.406]`, and the standard deviation values are `[0.229, 0.224, 0.225]`. Posters in the dataset came in different sizes, in order to work with all posters easily, we resize them to `224x224` images, so they could fit in the VGG-16 model.

Text Processing. Movies' title also include the year in which they were made. We seperated the year out of the title to only train with title.

After processing stage, the training set is splitted into a training set and a validate set with the ratio of `8:2`.

4.2 Training details

In the training process, we implement the weighted *Binary Cross-Entropy loss function* (BCELoss). BCELoss is used to measure loss. It measures the dissimilarity between predicted probabilities and target labels. In scenarios where class imbalance is a big challenge, the BCELoss function can be weighted to address this issue.

The weight assigned to each class is calculated based on the ratio of negative samples to positive samples for that particular label. By incorporating these weights into the loss function, the model is encouraged to pay more attention to the minority class (positive samples) during training. This helps mitigate the effects of class imbalance and ensures that the model does not disproportionately favor the majority class (negative samples).

We config the learning rate to 5e-4 with Adam optimizer and a StepLR schedular to dynamically adjust the learning rate throughout the training iterations. We train the model for 20 epochs, incorporating an Early Stopper to prevent overfitting with patience set to 3 and minimum delta set to 1e-2. A threshold of 0.15 will be applied to the model output for final prediction. As each movie must belong to at least one specific genre, if the prediction is 0s for all labels, we will set value 1 for the label having the highest prediction value. Finally, the 8-genre-prediction will be un-grouped, resulting in a 18-genre-prediction.

4.3 Evaluation Metrics

In our experiment, we evaluate the performance of the model based on precision, recall and F1-score, widely used for multi-label classification tasks. The formulas of those metrics are shown below.

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

where

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

4.4 Results

As can be seen from the Table 3 below, the first method using collaborative filtering shows a superior result on the test set compared with the second one using multi-modal network.

5 Conclusion

The outcomes of our experiments suggest that the rating-based method outperforms multi-modal network approach. In the scenario where rating data is available and prediction made on past observations is allowed, the rating-based method emerges as the optimal choice for addressing the task. On the

Method	Precision	Recall	Macro F1
Title (XLNet)	0.13	0.62	0.17
Poster (VGG-16)	**0.12**	**0.76**	**0.18**
Combine (Early fusion)	0.17	0.38	0.16
Combine (Late fusion)	0.12	0.56	0.17

Table 2: Comparison of separating modality models and combined models with two fusion timing.

Genre	Collaborative Filtering	Multi-modal Network
Action	0.63	0.23
Adventure	0.45	0.17
Animation	0.76	0.09
Children's	0.66	0.15
Comedy	0.68	0.49
Crime	0.24	0.09
Documentary	0.62	0.05
Drama	0.70	0.52
Fantasy	0.63	0.03
Film-Noir	0.47	0.03
Horror	0.80	0.34
Musical	0.47	0.05
Mystery	0.33	0.06
Romance	0.46	0.22
Sci-Fi	0.74	0.14
Thriller	0.53	0.30
War	0.48	0.06
Western	0.43	0.04
Average	**0.56**	**0.17**

Table 3: Comparison of Collaborative Filtering approach and Multi-modal Network approach in macro-F1 for each genre.

other hand, the multi-modal network approach offers the advantage of making predictions on entirely new observations without relying on past support data. However, it is notable that this approach may require a more robust model to effectively learn the patterns and relationships among the provided movie features.

References

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19.

Jonathan L Herlocker, Joseph A Konstan, and John Riedl. 2000. Explaining collaborative filtering recommendations. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pages 241–250.

Tae-Gyu Hwang, Chan-Soo Park, Jeong-Hwa Hong, and Sung Kwon Kim. 2016. An algorithm for movie classification and recommendation using genre correlation. *Multimedia Tools and Applications*, 75:12843–12858.

Joseph A Konstan, Bradley N Miller, David Maltz, Jonathan L Herlocker, Lee R Gordon, and John Riedl. 1997. Grouplens: Applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3):77–87.

Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1):76–80.

JÜRGEN PFEFFER. 2020. A collaborative filtering based approach to classify movie genres using user ratings. *Journal of Data Intelligence*, 1(4):442–467.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Philip Sedgwick. 2012. Pearson's correlation coefficient. *Bmj*, 345.

Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Xiaoyuan Su and Taghi M. Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Adv. in Artif. Intell.*, 2009.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Neural Information Processing Systems*.

Kai Yu, A. Schwaighofer, V. Tresp, Xiaowei Xu, and H.-P. Kriegel. 2004. Probabilistic memory-based collaborative filtering. *IEEE Transactions on Knowledge and Data Engineering*, 16(1):56–69.