

# A semantic similarity search engine for movies

Ahmad Mustafa, Hammam Mheidat, Adam Shatnawi

Department of Computer Information Systems, Jordan University of Science and Technology, Irbid, Jordan

---

## Article Info

### Article history:

Received Feb 27, 2024

Revised Aug 6, 2024

Accepted Aug 14, 2024

---

### Keywords:

Deep learning

Machine learning

Natural language processing

Recommender system

Search engine

Semantic similarity

---

## ABSTRACT

Semantic similarity has been gaining traction in the field of natural language processing. It is a measure of how similar two pieces of text are in terms of their meaning. It can be used to improve search engine results. We propose a deep learning-based approach to build a semantic similarity search engine for movies based on a movie summary. Filmmakers can gain insight into audience preferences and trends, allowing them to create more engaging and successful films. The dataset used in this study was gathered from internet movie database (IMDb), it includes movie summaries along with their corresponding name movies. The test dataset was generated using ChatGPT to be very close to human input. The universal sentence encoder (USE) model presented promising results in semantic similarity, the model results show that for the top 5 similar movies, the model returned 176 out of 300 movies (58.6%). For the top 10 similar movies, the model returned 211 out of 300 movies (70.3%). Additionally, for the top 15 similar movies, the model returned 229 out of 300 movies (76.3%). And, for the top 20 similar movies, the model returned 249 of 300 movies (83%). This method can be applied to movie recommendation systems or to organize films in a collection automatically.

*This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.*



---

## Corresponding Author:

Ahmad Mustafa

Department of Computer Information Systems, Jordan University of Science and Technology

Irbid, Jordan

Email: ammustafa@just.edu.jo

---

## 1. INTRODUCTION

Recommendation systems play an essential role in recommending items, particularly in streaming services. Streaming movie services such as Netflix rely on recommendation systems to assist consumers in selecting new movies to watch. These systems take into account several aspects including reviews, cast, narrative, crew, genre, and popularity to ensure optimal outcomes. It takes a user preference into consideration and hence gives results depending on the user's particular taste and decision. Users demand lightning-fast, efficient, and accurate results. As a result, a search engine system for movies has become an important service for the audience in retrieving a movie name without remembering its name. Currently, there are various types of recommender systems available [1], [2].

Natural language utterances are typically quite ambiguous due to polysemous words, which have multiple possible meanings, or malapropism, which is the confusion of an intended word with a word that sounds or is spelled similarly but has a very different and malapropos meaning. Generally, the only way to determine the interpretation of an utterance is to consider the context in which it occurred. Algorithms and programs, however, lack the advantage of human linguistic expertise [3].

An interaction model between humans and computers is essential for extracting the title of a film from a user's query. This article introduces a search engine that can accurately determine the title of a movie

using only a brief and succinct description. To handle this problem, we prepare a model that understands and interprets the human language for the computer, using universal sentence encoder (USE) [4], which is a pre-trained deep learning model developed by Google. It is designed to convert natural language text into a high-dimensional continuous vector representation, also known as an embedding [5]. These embeddings can then be used as input for a variety of natural language processing tasks, such as text classification, semantic similarity, and machine translation. The USE is trained on a diverse range of text data and can generate embeddings for a wide variety of sentence types and styles. It can also handle input text in multiple languages. USE can be used to encode multiple sentences at once and return a matrix, which can be used in tasks such as text clustering, retrieval and so on.

USE and sentence bidirectional encoder representations from transformers (SBERT) [6] can be used to compute the semantic similarity of sentences. However, based on the results of our test dataset, it is evident that USE is more suitable for this particular task. USE sentence embedding design, which is pre-trained on a vast and varied text dataset, may enhance its capacity to capture the entire textual meaning. However, sentence BERT (SBERT) is a specialized model designed specifically for generating sentence embeddings [7]. It exhibits strong generalization abilities across a wide range of natural language processing (NLP) applications. It surpasses USE in numerous ways. While both models possess their own merits, it is essential to conduct a comparison in order to determine the most suitable model for a specific NLP task.

The sections of the paper are organized as follows. Sections 2, 3, and 4. discuss the related work, methodology, experiments and results, respectively. The results of the proposed methods are discussed in section 5. Section 6 concludes the work.

## 2. RELATED WORK

We start by looking at the fascination of the scientific literature with movie classification or recommendation system. The approaches can be divided into classical machine learning [8]–[11] and deep learning [12]–[14]. Recent approaches suggested multimodal data[15]. Many have used collaborative filtering [16], [17]. Azaria *et al.* [18] proposed that the recommendation system designer should consider business revenue.

Chinnapottu *et al.* [19] have applied various machine learning models (such as logistic regression, random forest, naive Bayes, and support vector machine) to predict the name of a movie using input text that has a semantic similarity to the movie's script. They used a dataset of 11 movie scripts scraped from the internet movie script database (IMSDb) website and preprocessed using various natural language processing techniques, including stop word removal, punctuation removal, tokenization, stemming, and vectorization. The results of the study showed that the naive Bayes classifier and support vector machine models performed better than the logistic regression and random forest models in predicting the name of the movie. It is worth noting that the small size of the training data (11 movies' scripts) and testing data (166 paraphrased sentences from the movies' scripts) may have limited the generalizability of the results.

Yoo *et al.* [20] have developed a model that can accurately identify a movie when searching with a sequence of words. They proposed a model that consists of three parts: one for extracting word embeddings using a skip-grams based approach, one for part-of-speech tagging using hidden markov models and Viterbi algorithms, and one for named entity recognition using a bidirectional long short-term memory (Bi-LSTM) approach. For model evaluation, 50 participants were randomly selected and asked to choose a title from a list of 30 movies. The results showed that the proposed model outperformed existing search systems such as Naver and Google. It was able to accurately identify the correct movie when searching with a sequence of words.

Bhattacharya and Lundia [21] have crafted a web application that assists users in discovering movies that align with their preferences. The application generates a list of highly similar movies based on inputs from the user, based on some characteristics of the movie such as the summary plot, actors, and director. To accomplish this, the team utilized IMDb data and created a representation for each movie in the form of a "bag of words" that encapsulated the plot summary, actors, and directors. The key to the application's ability to suggest similar movies was the use of cosine similarity. This mathematical technique was applied to measure the similarity between the user's input and the words that represent movies, thereby enabling the identification of the movies that were most similar the user's input.

Roy *et al.* [22] have employed a semi-supervised learning method to develop a movie recommendation system. The authors proposed three approaches; popularity-based Recommender system, simple recommender

system, and content-based filtering approach. Besides highlighting the advantages and disadvantages of each approach, the study also demonstrated how one approach was mitigated by another. The project was built using Django, a popular web application development technology, which enabled the creation of a user-friendly interface for the movie recommendation system. The approaches were tested and evaluated on the well-known movie lens dataset, which contains a large collection of movie ratings and reviews from users. The results showed that the combination of the three approaches resulted in more accurate and personalized recommendations for users.

Walek and Fojtik [23] proposed a hybrid system called predictory that combines a collaborative filtering system, a content-based system, and a fuzzy expert system to recommend movies to users. The collaborative filtering system uses singular value decomposition (SVD) algorithm to predict the evaluation of movies. The content-based system uses the properties of the movies, such as the genre and plot, to make recommendations. The fuzzy expert system evaluates the level of importance of movies for the final list of recommended movies based on various parameters, such as the average movie rating, the number of movie ratings, and the level of similarity with already rated movies. The system was validated on a group of users using the Movie Lens dataset and showed promising results. The results were evaluated using standard metrics such as precision, recall, and F1-measure, with values of 81%, 83%, and 82% respectively. The system was also compared with other typical approaches, such as a pure content-based system, a pure collaborative filtering system, and a weighted hybrid system. The results showed that the proposed hybrid system using an expert system achieved the highest ratio of relevant movies marked by the users during testing. This means the system was able to recommend the most relevant movies to the users.

The movie recommender system proposed by Ahuja *et al.* [9] is designed to make personalized movie recommendations to users by clustering similar movies together and using K-nearest neighbor algorithm to recommend movies that are like the ones a user has previously liked. The system utilizes k-means clustering algorithm to group similar movies together based on their attributes such as genre, director, and cast. This clustering is done to make the movie recommendation more accurate by recommending movies that are like the movies a user has previously liked. The data for this system is taken from the movie lens dataset, which is a widely used dataset in movie recommendation systems research. This dataset contains information on movies and the ratings given to them by users. The system is implemented in Python programming language, which is a popular choice for machine learning projects due to its ease of use and many libraries available for machine learning tasks. The system demonstrates improved root mean squared error (RMSE) compared to existing techniques. RMSE is a measure of how well the recommendations of a system match the actual ratings given by users.

Ye *et al.* [24] introduced a solution to Xbox One's movie search job that derives granular relevance of training samples from session data. Several user-engagement measures have shown that the strategy produces improved test results. When the score is computed, it also gives a large performance boost. Search engines rely significantly on their ability to interpret the intent and context of a user's query. It is challenging to create a search engine that recognizes these phrases and pages, especially if the query is brief and vague. To solve this issue, several models have been presented; below are some of the greatest examples from the literature. Topic indexing nonlinear models can be used to the demands of semantic matching for search engines. Examples of nonlinear models include probabilistic latent analysis (PLSI) and latent dirichlet allocation (LDA). One important application of these models is to analyze the similarity between query documents at the topic and semantic level. Another set of latent semantic models are neural network models, which can be used to calculate the query-document similarity.

### 3. METHOD

#### 3.1. Training dataset

To obtain a comprehensive dataset of movies, we scrape the dataset from IMDb website using *BeautifulSoup4*. To ensure that the dataset is as specific and relevant as possible, we applied a series of filters to the website's advanced search. These filters include a release date range of between January 1<sup>st</sup> 1980 and January 1<sup>st</sup> 2023, a minimum rating count of 10,000, and a requirement that the movies be in English. Additionally, we set the "including adults title" filter to "true" to ensure that the dataset would include a diverse range of movie titles. As a result of these filters, we collected 7,653 movies that met our criteria. This dataset includes 9 features such as the movie's title, director, release year, summary, duration, rating, vote count, gross revenue

in millions of dollars, and genre. For a visual representation of the distribution of movies in this dataset over the years, please refer to Figure 1.

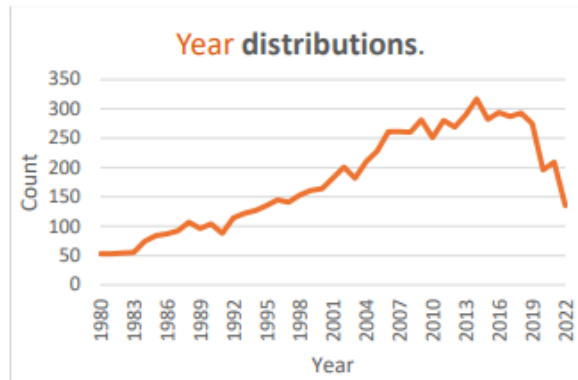


Figure 1. The distribution of movies over years

### 3.2. Testing dataset

Given the limited availability of datasets in the field of natural language processing, we used ChatGPT to generate a dataset for our research. ChatGPT is a large language model developed by OpenAI, which is based on the generative pre-training transformer (GPT) architecture. It is trained using a vast dataset of text, allowing it to produce human-like responses to text-based prompts. This versatile model can be fine-tuned for various natural language processing tasks such as language translation, text summarization, text completion, and question answering, as well as more creative endeavors like writing poetry and other forms of text. ChatGPT is designed to be user-friendly, with a simple API that can be integrated into various applications such as chatbots, voice assistants, and language understanding systems. To simulate human input (query), we used a prompt that asked ChatGPT to “generate a description about a movie without mentioning the name of the movie or the names of actors and characters in one sentence.” This generated 400 descriptions. These descriptions represents the user queries. The dataset contains two columns the movie description (query) and the target movie name. Examples of movies are shown in Table 1.

Table 1. Examples of ChatGPT generated descriptions for the movies: baby driver, American History X, and The Lion King

| The description generated by ChatGPT   | Movie name         |
|--|--------------------|
| A young, talented getaway driver tries to leave his criminal past behind and start a new life but is pulled back into the world of crime by a group of dangerous bank robbers.   | Baby Driver        |
| A young man who becomes involved in white supremacist activity as a result of a tragic family event is sent to prison, where he undergoes a transformation and works to prevent his younger brother from following in his footsteps. | American History X |
| A young lion prince is exiled from his kingdom after the death of his father, but eventually returns to reclaim his place as rightful king and end the reign of his power-hungry uncle.  | The Lion King      |

### 3.3. Preprocessing

For the training dataset (which is collected from IMDB), we drop unnecessary columns (i.e., “director,” “release year,” “duration,” “rating,” “vote count,” “gross,” and “genre”) because we aim to train the model with the movie title and the summary. Preprocessing is not typically necessary when using the USE because it is a pre-trained model that is able to handle a wide variety of input text, including text with different capitalization, punctuation, and grammatical structures. Additionally, the USE is designed to handle input text that may contain typos, misspellings, or other errors. Therefore, it can produce useful embeddings even when applied to raw, unprocessed text. For the testing dataset after we look for the data, we found descriptions of movies such that their movie name is in the descriptions. So, after removing them the dataset becomes 300 movies. Figure 2 shows the methodology applied in our research.

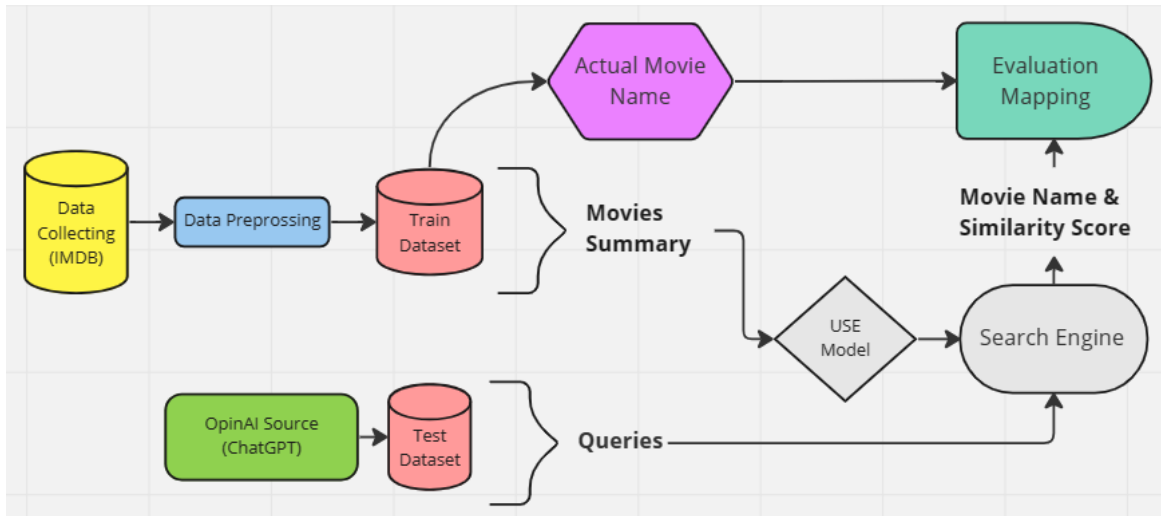


Figure 2. The methodology

### 3.4. Evaluation

It was not possible to use mean reciprocal rank (MRR) [25] because all the results retrieved for movie titles were relevant to the query. The same applies to mean average precision (MAP) [26]. Based on the above, we were only looking at whether the requested movie title was on the retrieved list, regardless of its order among the remaining movies. We divided the number of queries returning the intended movie title by the total number of queries from the list of titles retrieved. Therefore, our value differed according to the number of films in the retrieved list.

$$Accuracy = \frac{Q_{True}}{N} \tag{1}$$

where  $N$  is the total number of queries and  $Q_{True}$  is the number of queries that returned the correct movie among the top  $k$ .

## 4. RESULTS

The results from using the USE model for finding the most similar movies are as follows: for the top 5 similar movies, the model returned 176 out of 300 movies (58.6%). For the top 10 similar movies, the model returned 211 out of 300 movies (70.3%). Additionally, for the top 15 similar movies, the model returned 229 out of 300 movies (76.3%). And, for the top 20 similar movies, the model returned 249 out of 300 movies (83%). For further details, please refer to Table 2.

In Table 3 we show the accuracy reported by two selected related work [19], [23]. Both approaches were applied to movie semantic search. However, they search using movie summary and scenario. In [19]. The original movie name is predicted by using random paraphrased sentences from movie scripts. On the other hand, in [23] movies are recommended by evaluating their importance. So, the testing dataset is different than ours.

Table 2. The results we obtained from using USE

|                           | Top 5 | Top 10 | Top 15 | Top 20 |
|---------------------------|-------|--------|--------|--------|
| Number of movies returned | 176   | 211    | 229    | 249    |
| Accuracy                  | 58.6% | 70.3%  | 76.3%  | 83%    |

Table 3. The accuracy obtained by the related work

| Approach | Accuracy |
|----------|----------|
| [19]     | 59.63%   |
| [23]     | 81%      |
| Our Work | 83%      |

We also used the test dataset to evaluate ChatGPT's performance as a movie search engine for the Top K movies. To compare with our presented method using the USE model, we also used different models from SBERT [27] (namely, all-mpnet-base-v2, msmarco-distilbert-base-v2, multi-qa-mpnet-base-cos-v1, and paraphrase-distilroberta-base-v1). Although the SBERT model outperforms the USE model in various textual similarity tasks, USE is not our contribution, and the results shown in Table 4 illustrate that USE outperforms SBERT in our case.

Table 4. The accuracy obtained by our model vs. other models on the same testing data

|       |                  | Our Model | Chat GPT | multi-qa-mpnet-base-cos-v1 | msmarco-distilbert-base-v2 | all-mpnet-base-v2 | paraphrase-distilroberta-base-v1 |
|-------|------------------|-----------|----------|----------------------------|----------------------------|-------------------|----------------------------------|
| Top5  | Retrieved movies | 176       | 157      | 185                        | 155                        | 162               | 164                              |
|       | Accuracy (%)     | 58.6      | 52.3     | 61.7                       | 51.7                       | 54                | 54.7                             |
| Top10 | Retrieved movies | 211       | 173      | 203                        | 178                        | 184               | 186                              |
|       | Accuracy (%)     | 70.3      | 57.6     | 67.67                      | 59.33                      | 61.33             | 62                               |
| Top15 | Retrieved movies | 229       | 59.7     | 211                        | 185                        | 195               | 197                              |
|       | Accuracy (%)     | 76.3      | 179      | 70.3                       | 61.7                       | 65                | 65.7                             |
| Top20 | Retrieved movies | 249       | 197      | 219                        | 190                        | 215               | 209                              |
|       | Accuracy (%)     | 83        | 65.7     | 73                         | 63.3                       | 71.7              | 69.7                             |

Figure 3 illustrates an example of a movie search using (ChatGPT) description of "Fast and Furious: Tokyo Drift." Table 4 shows the top 10 similar movies to the query "A person who moved to another country because of his obsession with racing cars." The actual movie summary is "A teenager becomes a major competitor in the world of drift racing after moving in with his father in Tokyo to avoid a jail sentence in America." The results indicate that the movie "Fast and Furious: Tokyo Drift" is ranked number one with a similarity score of 0.5318.

| index | title                                      | Score    |
|-------|--|----------|
| 0     | 5400 The Fast and the Furious: Tokyo Drift | 0.531872 |
| 1     | 4733 Cars 2                                | 0.509859 |
| 2     | 6596 Replicas                              | 0.489092 |
| 3     | 7193 The Brown Bunny                       | 0.450578 |
| 4     | 5387 Speed Racer                           | 0.449860 |
| 5     | 6063 Planes                                | 0.446802 |
| 6     | 7544 The Love Guru                         | 0.444060 |
| 7     | 2873 Cars 3                                | 0.426371 |
| 8     | 5223 Days of Thunder                       | 0.422008 |
| 9     | 5037 Mortal Engines                        | 0.421937 |

Figure 3. Example for retrieved movies when using the description of "Fast and Furious: Tokyo Drift"

## 5. DISCUSSION AND LIMITATIONS

Although we produced encouraging results utilizing USE on our test data compared to other models on the same movie data, it is evident that there is much potential for accuracy improvement as we explore the realm of search engines. One plausible approach to addressing this issue could involve the integration of an expanded dataset or the utilization of storylines and scenarios from the movies as a method for training the model. USE may struggle to comprehend the significance of idiomatic or figurative language due to its improper usage, especially when it represents a general summary of a specific movie.

A potential consequence of this phenomenon is the decrease in performance when engaging in tasks that strongly depend on comprehending the subtleties of language, such as sarcasm or metaphor. Furthermore, the model may encounter challenges in comprehending the sentence's context or the interconnections among various concepts in the text. This phenomenon may result in a decline in performance when evaluating texts that include identical semantic content but exhibit distinct surface-level grammar. Another thing that makes it hard to use is that it relies on a transformer-based architecture that is not really made for semantic similarity

but for understanding language in general. This may result in reduced efficiency or accuracy in some specific cases. Despite the current level of precision, the results indicate that model USE achieved the highest level of accuracy when compared to models SBERT and ChatGPT, even though we used model ChatGPT to generate the test data. This is still remarkable and can serve as a benchmark for evaluating the effectiveness of a synopsis search engine, especially for movie enthusiasts who frequently look for a specific movie based on a synopsis they have heard or encountered, even without knowing anything about the title of the film.

## 6. CONCLUSION

In this paper, we highlight the promising results of using the USE for movie semantic search, outperforming other models like SBERT and ChatGPT in accuracy. However, we identified potential for improvement, particularly in handling idiomatic language and understanding context and connections in text. We suggest expanding the dataset and incorporating movie storylines for better training. Our findings instill hope for both avid movie enthusiasts and academics in the search engine and natural language processing fields.




## REFERENCES

- [1] J. Lund and Y.-K. Ng, "Movie recommendations using the deep learning approach," in *2018 IEEE International Conference on Reuse and Integration (IRI)*, Jul. 2018, pp. 47–54, doi: 10.1109/IRI.2018.00015.
- [2] K. Raj, A. A. Das, A. Guha, P. Sharma, and M. K. S., "Movie recommendation system," *International Journal of Computer Sciences and Engineering*, vol. 7, no. 4, pp. 1024–1028, 2019, doi: 10.26438/ijcse/v7i4.10241028.
- [3] H. Ezzikouri, M. Erritali, and M. Oukessou, "Semantic similarity/relatedness for cross language plagiarism detection," in *2016 13<sup>th</sup> International Conference on Computer Graphics, Imaging and Visualization (CGiV)*, Mar. 2016, vol. 1, no. 2, pp. 372–374, doi: 10.1109/CGiV.2016.78.
- [4] D. Cer *et al.*, "Universal sentence encoder for English," in *EMNLP 2018 - Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Proceedings*, 2018, pp. 169–174, doi: 10.18653/v1/d18-2029.
- [5] S. Lai, K. Liu, S. He, and J. Zhao, "How to generate a good word embedding?," *IEEE Intelligent Systems*, pp. 1–1, 2017, doi: 10.1109/mis.2017.2581325.
- [6] J. Devlin, M.-W. Chang, K. Lee, K. T. Google, and A. I. Language, "BERT: Pre-training of deep bidirectional transformers for language understanding," *Naacl-Hlt 2019*, pp. 4171–4186, 2018.
- [7] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using siamese BERT-networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9<sup>th</sup> International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 3980–3990, doi: 10.18653/v1/D19-1410.
- [8] S. Sridhar, D. Dhanasekaran, and G. C. P. Latha, "Content-based movie recommendation system using MBO with DBN," *Intelligent Automation and Soft Computing*, vol. 35, no. 3, pp. 3241–3257, 2023, doi: 10.32604/i-asc.2023.030361.
- [9] R. Ahuja, A. Solanki, and A. Nayyar, "Movie recommender system using K-means clustering and K-nearest neighbor," in *2019 9<sup>th</sup> International Conference on Cloud Computing, Data Science and Engineering (Confluence)*, Jan. 2019, pp. 263–268, doi: 10.1109/CONFLUENCE.2019.8776969.
- [10] M. R. Hasan and J. Ferdous, "Dominance of AI and machine learning techniques in hybrid movie recommendation system applying text-to-number conversion and cosine similarity approaches," *Journal of Computer Science and Technology Studies*, vol. 6, no. 1, pp. 94–102, 2024, doi: 10.32996/jcsts.2024.6.1.10.
- [11] H. Leng *et al.*, "Finding similar movies: Dataset, tools, and methods," *Computer Science Research Notes*, vol. 2802, pp. 115–124, 2018, doi: 10.24132/CSRN.2018.2802.15.
- [12] A. Kutlimuratov and N. Atadjanova, "Movie recommender system using convolutional neural networks algorithm," *Science and Innovation*, vol. 2, no. 3, 2023.
- [13] K. K. Yadav, H. K. Soni, G. Yadav, and M. Sharma, "Collaborative filtering based hybrid recommendation system using neural network and matrix factorization techniques," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 12, no. 8s, pp. 695–701, 2024.
- [14] S. Reddy and J. K. Sarode, "Hybrid recommendation system using graph neural network and BERT embeddings," *arXiv:2310.04878*, 2023.
- [15] Y. Mu and Y. Wu, "Multimodal movie recommendation system using deep learning," *Mathematics*, vol. 11, no. 4, 2023, doi: 10.3390/math11040895.
- [16] A. Li, X. Liu, and B. Yang, "Item attribute-aware graph collaborative filtering," *Expert Systems with Applications*, vol. 238, 2024, doi: 10.1016/j.eswa.2023.122242.
- [17] V. L. Chetana and H. Seetha, "Handling massive sparse data in recommendation systems," *Journal of Information and Knowledge Management*, vol. 23, no. 3, 2024, doi: 10.1142/S0219649224500217.




- [18] A. Azaria, A. Hassidim, S. Kraus, A. Eshkol, O. Weintraub, and I. Netanel, "Movie recommender system for profit maximization," *AAAI Workshop - Technical Report*, vol. WS-13-11, pp. 2–8, 2013.
- [19] B. Chinnapottu, G. Arikatla, and S. Memeti, "Movie prediction based on movie scripts using natural language processing and machine learning algorithms," M.S. thesis, Faculty of Computing, Blekinge Institute of Technology, 2021.
- [20] H. Yoo, M. Kang, and K. Oh, "A semantic search model using word embedding, POS tagging, and named entity recognition," in *2018 International Conference on Computational Science and Computational Intelligence (CSCI)*, Dec. 2018, pp. 1204–1209, doi: 10.1109/CSCI46756.2018.00231.
- [21] S. Bhattacharya and A. Lundia, "Movie recommendation system using bag of words and Scikit-Learn," *International Journal of Engineering Applied Sciences and Technology*, vol. 04, no. 05, pp. 526–528, 2019, doi: 10.33564/ijeast.2019.v04i05.076.
- [22] S. Roy, M. Sharma, and S. K. Singh, "Movie recommendation system using semi-supervised learning," in *2019 Global Conference for Advancement in Technology (GCAT)*, Oct. 2019, pp. 1–5, doi: 10.1109/GCAT47503.2019.8978353.
- [23] B. Walek and V. Fojtik, "A hybrid recommender system for recommending relevant movies using an expert system," *Expert Systems with Applications*, vol. 158, 2020, doi: 10.1016/j.eswa.2020.113452.
- [24] X. Ye, Z. Qi, X. Song, X. He, and D. Massey, "Generalized learning of neural network based semantic similarity models and its application in movie search," in *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, Nov. 2015, pp. 86–93, doi: 10.1109/ICDMW.2015.34.
- [25] N. Craswell, "Mean reciprocal rank," *Encyclopedia of Database Systems*, pp. 2217–2217, 2018, doi: 10.1007/978-1-4614-8265-9\_488.
- [26] E. Zhang and Y. Zhang, "Average precision," *Encyclopedia of Database Systems*, pp. 192–193, 2009, doi: 10.1007/978-0-387-39940-9\_482.
- [27] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using siamese BERT-networks," in *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, 2019, pp. 3982–3992, doi: 10.18653/v1/d19-1410.

## BIOGRAPHIES OF AUTHORS






**Ahmad Mustafa**    received the Ph.D. degree in computer science from the University of Texas at Dallas in 2018. He is currently an assistant professor at the Jordan University of Science and Technology. His current research interests include continual learning, active learning, and natural language processing. He can be contacted at email: ammustafa@just.edu.jo.



**Hammam Mheidat**    is a master student and teaching assistant in Data Science Program in the Jordan University of Science and Technology. His research interests include machine learning, deep learning, computer vision, and natural language processing. He can be contacted at email: hmmheidat206@cit.just.edu.jo.



**Adam Shatnawi**    is a master student in Data Science Program in the Jordan University of Science and Technology. His areas of research encompass deep learning, data analytics and visualization, as well as natural language processing. He can be contacted at email: azshatnawi20@cit.just.edu.jo.