VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)
CHENNAI

Course Recommendation System

Project Report

Recommender Systems

CSE – 4077

M.Tech CSE with Specialization in Business Analytics

School of Computer Science and Engineering (SCOPE)

VIT Chennai

Fall Semester 2024-25

Slot: F1+TF1

Submitted by

Sonakshi Dinesh – 21MIA1082

Adeline Marion – 21MIA1048

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CHENNAI

CERTIFICATE

This is to certify that the project work entitled “Course Recommendation System” that is being submitted by the above mentioned team members is a record of bonafide work done under my supervision. The content of this project work, in full or in parts, have neither been taken from any other source nor have been submitted for any other course.

Place: Chennai

Date: 11/11/2024

Signature of faculty

Dr. Parvathi R

ACKNOWLEDGEMENT

We would like to express our special thanks and gratitude to our Course faculty **Dr.Parvathi R,** Assistant Professor of School of Computer Science and Engineering who gave us the golden opportunity to do this wonderful project titled "Course Recommendation System", and also for providing us with proper guidance and suggestions to develop this project. This helped us in doing a lot of research and we came to know about so many new things.

We are extremely grateful to **Dr. Ganesan R ,** Dean of School of Computer Science and Engineering, VIT Chennai, for extending the facilities of the School towards our project and for his unstinting support.

We express our thanks to our Head of the Department **Dr. Sivabalakrishnan.M** for his support throughout the course of this project.

We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the course.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

CONTENTS

ABSTRACT

A system for recommending courses is intended to improve the quality of education that students get. Students academic goals, hobbies, and skill levels are taken into account when the system recommends courses based on Deep learning and an Hybrid approach is also to be used. This ensures recommendations are based on strong academic reasoning by including prerequisite requirements and subject expertise. Enhancing relevance, context-aware filtering modifies recommendations based on the user's present learning environment. Furthermore, relationships between users and courses are modelled via graph-based filtering, which captures intricate dependencies and improves recommendation accuracy. This combined method solves the cold start issue, increases the accuracy of the recommendations, and provides a reliable, scalable fix for dynamic educational needs. The system adapts to user interactions through continuous feedback loops and real-time data processing, offering an adaptive learning path that is customised to each student's individual journey in higher education.

INTRODUCTION

With a greater focus on individualised learning experiences, the educational landscape is changing dramatically. Conventional ways of choosing courses, which sometimes depend on static catalogues and little student input, are no longer adequate. This is where sophisticated machine learning and hybrid approach-powered course recommendation systems are useful.

Motivated by the success of e-commerce product recommendation algorithms, these creative ideas seek to transform how students proceed through their academic careers. Course recommendation systems examine a great deal of student data, such as academic records, learning preferences, and styles, to produce customised course recommendations, much like e-commerce platforms use user data to promote related products.

This approach empowers students to make informed course selections that:

- Align with their academic goals: Recommendations consider students long-term aspirations, ensuring their chosen courses contribute to their desired career path.

- Enhance learning: By suggesting courses that match their learning styles and interests, the system fosters a more engaging and enriching educational experience.

- Optimize learning pathways: Recommendations account for pre-requisites and course dependencies, ensuring students build a strong academic foundation and progress efficiently.

DATASET

The dataset used for training and testing the model is from Kaggle. It contains the following attributes - Course Name, University, Difficulty Level, Course Rating, Course URL, Course Description, Skills. It contains about 3000 records.

LITERATURE SURVEY

Paper 1

Title : Hybrid Algorithm Based on Content and Collaborative Filtering in Recommendation System Optimization and Simulation

Author : Steffen Rendle, Walid Krichene, Li Zhang, John Anderson and Lili Wu.

Methodology : The methodology involves developing a hybrid recommendation algorithm that integrates content filtering and collaborative filtering, utilizing a user feature rating matrix and K-means clustering to enhance recommendation accuracy.

Description : This paper presents a hybrid recommendation algorithm that combines content filtering and collaborative filtering to optimize recommendation systems. It addresses challenges such as data sparsity and cold start problems by utilizing a user feature rating matrix instead of the traditional user-item rating matrix. The algorithm employs K-means clustering to group users with similar interests, enhancing the accuracy of recommendations.

Paper 2

Title : Outer Product-based Neural Collaborative Filtering

Author : Xiangnan He, Xiaoyu Du, Xian Wang, Feng Tian, Jinhui Tang and Tat-Seng Chua

Methodology : The methodology involves using an outer product to create an interaction map from user and item embeddings, followed by employing convolutional neural networks to learn high-order correlations from this map for personalized recommendations.

Description : This paper introduces a new neural network framework called ONCF for collaborative filtering, which utilizes an outer product to model pairwise correlations between embedding dimensions, resulting in a more expressive interaction map. The proposed ConvNCF model employs convolutional neural networks to learn high-order correlations from this interaction map.

Paper 3

Title : Hybrid Course Recommendation System Design for a Real-Time Student Automation Application

Author : Alper Arık, Savaş Okyay, Nihat Adar

Methodology : The study proposes a hybrid recommender system combining collaborative filtering and content-based filtering models, utilizing student and course information to provide personalized course recommendations without predefined association rules.

Description : The study presents a hybrid course recommendation system designed to assist students in selecting courses based on their interests and academic performance. It integrates collaborative filtering, which uses grades as ratings, and content-based filtering, which analyzes course and student information through natural language processing. The proposed system aims to overcome challenges such as data sparsity and cold start issues, providing stable and unbiased recommendations.

Paper 4

Title : A Comprehensive Review of Course Recommendation Systems for MOOCs

Author : Mohd Mustafeez ul Haque, Bonthu Kotaiah, Jameel Ahmed

Methodology : This study uses a systematic literature review (SLR) to analyze 76 articles from multiple databases, focusing on MOOC course recommendation systems from 2015-2023. It explores machine learning, deep learning, and traditional filtering methods.

Description : The paper reviews trends and developments in MOOC recommendation systems, highlighting challenges like data sparsity, cold start problems, and data overload. It emphasizes machine learning approaches, including collaborative filtering and hybrid methods, as key solutions. The paper also discusses future directions and unresolved issues in MOOC recommendation systems.

Paper 5

Title : A Course Hybrid Recommender System for Limited User Information Scenarios

Author : Juan Camilo Sanguino, Rubén Manrique, Olga Mariño, Mario Linares-Vásquez, Nicolás Cardozo

Methodology : The paper presents a hybrid recommender system combining collaborative and content-based filtering strategies. User data is inferred from Google Analytics logs to handle the challenge of limited explicit user information.

Description : This study addresses the cold start problem in online learning platforms where user data is scarce. The system estimates user ratings based on session duration and lesson views, leveraging text embeddings to enhance content similarity. The hybrid approach achieved a recommendation precision of 0.4, showing potential for course recommendation in minimal data environments.

Paper 6

Title : Implementation of Course Recommender System for Virtual University of Pakistan

Author : Aleem Akhtar

Methodology : The system uses user-based collaborative filtering to recommend courses based on the similarity between students. Cosine similarity is applied to compute recommendations, and Mean Absolute Error is used to evaluate prediction accuracy.

Description : The recommender system is designed to assist students in selecting courses that align with their competencies. It uses simulated data from 2600 students and 470 courses to predict future performance in courses. Testing showed that predicted grades were heavily influenced by a student's average past performance.

Paper 7

Title : Deep Learning Enabled Course Recommendation Platform

Author : S. Sajiharan & Kisan Pal Singh

Methodology : Developed a Taylor-Competitive Swarm Optimizer (Taylor-CSO) for performance prediction and course recommendation based on learning styles.

Description : This research focuses on improving the accuracy of course recommendation systems using deep learning. It utilizes the E-Khool learning platform and random forest to predict learning styles, followed by a deep residual neural network trained using the proposed Taylor-CSO algorithm. The system enhances course recommendations by predicting learning performance, aiming for higher accuracy and effective personalization.

Paper 8

Title : An Intuitive Implementation of Course Recommendation System Based on Learner's Personality

Author : Asad Jatoi, Memoona Sami, Muzamil Nawaz, Junaid Baloch

Methodology : Naïve Bayes algorithm for learner classification based on learning styles and competencies.

Description : This research categorizes students into theoretical, practical, or hybrid learners. It uses the Naïve Bayes algorithm to classify their proficiency and recommend courses accordingly. The study focuses on personalizing learning based on student types to improve course recommendations.

Paper 9

Title : A Survey of Recommendation Systems: Recommendation Models, Techniques, and Application Fields

Author : Hyeyoung Ko, Suyeon Lee, Yoonseo Park

Methodology : The study analyzes research trends and models in recommendation systems through a literature survey of approximately 100 papers since 2010.

Description : This article explores the evolution and application of recommendation systems, focusing on various models such as Content-Based Filtering, Collaborative Filtering, and Hybrid Systems. It highlights the importance of data mining techniques and the integration of user preferences in enhancing recommendation accuracy. The research also reviews the trends in recommendation services across different application fields, providing insights for future studies.

Paper 10

Title : Collaborative Filtering Recommendation Algorithm for MOOC Resources Based on Deep Learning

Author : Lili Wu

Methodology : The research develops a deep learning-based collaborative filtering recommendation model that incorporates embedding vectors and an attention mechanism.

Description : This study aims to enhance the recommendation performance for MOOC resources by addressing the shortcomings of traditional algorithms. It utilizes embedding vectors derived from metapaths and integrates relational network information through a Laplacian matrix. Experimental results indicate that the proposed model significantly improves accuracy and stability in recommendations.

Paper 11

Title : Network Course Recommendation System Based on Double-Layer Attention Mechanism

Author : Qianyao Zhu

Methodology : The study proposes a recommendation system utilizing a double-layer attention mechanism integrated into a parallel neural network model.

Description : This research addresses the challenges of accurate course recommendations on online teaching platforms. It introduces a dual attention mechanism to enhance feature extraction from student and course data. Experimental results demonstrate improved recommendation accuracy compared to existing algorithms.

Paper 12

Title : Neural Collaborative Filtering vs. Matrix Factorization Revisited

Author : Steffen Rendle, Walid Krichene, Li Zhang, John Anderson

Methodology : The paper compares the performance of dot product and MLP-based learned similarities for collaborative filtering.

Description : This work revisits the experiments of the Neural Collaborative Filtering (NCF) paper, demonstrating that a properly configured dot product outperforms MLP-based similarities. It highlights the challenges of learning a dot product with an MLP and discusses practical issues in production environments. The findings suggest that dot products may be a better default choice for combining embeddings in recommender systems.

Paper 13

Title : A Context-Aware Location Recommendation System for Tourists Using Hierarchical LSTM Model

Author : Wafa Shafqat and Yung-Cheol Byun

Methodology : The study proposes a hierarchical deep learning model utilizing LSTM to predict tourist locations based on user travel history and contextual features.

Description : This research highlights the importance of contextual data in enhancing recommendation systems for tourism. It introduces a two-level LSTM model that incorporates various contextual factors such as weather, location popularity, and distance to improve accuracy. The proposed system achieved an impressive accuracy of 97.2% in predicting the next tourist location.

Paper 14

Title : The Recommendation System of Innovation and Entrepreneurship Education Resources in Universities Based on Improved Collaborative Filtering Model

Author : Li Geng

Methodology : The study proposes a behavior path collaborative filtering recommendation algorithm utilizing a behavior map and Keras Tokenizer for vectorization.

Description : This research addresses the challenge of locating relevant online educational resources, proposing an improved collaborative filtering algorithm to enhance recommendation accuracy. It leverages student behavior data to create behavior paths, enabling personalized recommendations for innovation and entrepreneurship education resources. Experimental results demonstrate the algorithm's effectiveness compared to traditional methods.

Paper 15

Title : Systematic Review of Recommendation Systems for Course Selection

Author : Shrooq Algarni and Frederick Sheldon

Methodology : A systematic literature review (SLR) analyzing 35 key studies from 1938 academic papers.

Description : This article reviews the role of course recommender systems in enhancing student decision-making in education. It highlights challenges such as the cold start problem and the need for personalized recommendations. The study synthesizes empirical data to identify effective methodologies and suggests future research directions in the field.

Paper 16

Title : A Content-Based Recommender System Using Stacked LSTM and an Attention-Based Autoencoder

Author : Kapil Saini,Ajmeer Singh

Methodology : The paper proposes a novel content-based recommender system that integrates stacked Long Short-Term Memory (LSTM) networks with an attention-based autoencoder for enhanced recommendation accuracy.

Description : This research addresses the challenges of scalability and cold start problems in recommender systems by leveraging minimal product information. The proposed method demonstrates improved accuracy and computational efficiency while effectively capturing low-dimensional latent representations. Simulations on Amazon product datasets validate the effectiveness of the approach.

Paper 17

Title : Deep Neural Architecture for News Recommendation

Author : Vaibhav Kumar, Dhruv Khattar, Shashank Gupta, Manish Gupta, Vasudeva Varma

Methodology : The study utilizes a deep neural network with attention mechanisms to address the dynamic nature of user interests in news recommendation.

Description : This research highlights the challenges of varying user interests over time in news consumption and proposes a hybrid approach that combines user-item interactions with news content. Extensive experiments demonstrate significant improvements in recommendation performance, particularly in handling user and item cold-start problems. The model effectively captures both static and dynamic user interests, enhancing the relevance of news recommendations.

Paper 18

Title : Research of Online Courses Recommendation Based on Deep Learning

Author : Yuxuan Zhao, Chuantao Yin, Xi Wang, Yanmei Chan

Methodology : The study employs deep learning techniques, including graph neural networks and sequential recommendation methods, to enhance course recommendation systems.

Description : This research addresses the limitations of traditional recommendation methods in online learning environments by proposing novel algorithms that improve recommendation accuracy and personalization. By analyzing user behavior patterns and course attributes, the approach aims to enhance learning outcomes and experiences in e-learning platforms. The findings contribute to the advancement of e-learning technology and the application of deep learning in smart education.

Paper 19

Title : A Graph Convolution Network Based on Improved Density Clustering for Recommendation System

Author : Yue Li

Methodology : The study proposes a recommendation algorithm that utilizes graph convolution networks combined with an improved density clustering method to enhance recommendation accuracy.

Description : This research addresses the limitations of traditional recommendation systems that primarily rely on Euclidean data by leveraging graph structures to better capture user and item relationships. The proposed model clusters users based on their influence and extracts features through a graph convolution network, resulting in improved recommendation performance. Experiments validate the effectiveness of the model in processing large-scale data and enhancing user experience.

Paper 20

Title : Deep Density Networks and Uncertainty in Recommender Systems

Author : Yoel Zeldes, Stavros eodorakis, Efrat Solodnik, Aviv Rotman

Methodology : The study introduces a hybrid deep neural network model, Deep Density Networks (DDN), that integrates content-based deep learning with collaborative filtering to model and estimate uncertainty in recommendations.

Description : This research addresses the challenges of balancing exploration and exploitation in online recommendation systems by incorporating uncertainty estimations into the recommendation process. The DDN model effectively captures complex interactions between user preferences and content features, leading to improved recommendation accuracy. Results from real-world applications demonstrate the practical benefits of using DDN, including enhanced performance in serving billions of recommendations daily.

TOOLS USED

A range of technologies and techniques are used to create a strong course recommendation system, each with a specific function in data processing, model training, and system implementation.

1. Python : Python's large library, readability, and community support make it the primary programming language used to create the recommendation system. Python is the best option for developing and implementing recommendation systems because it offers the flexibility required to apply and test deep learning, machine learning, and data processing techniques.

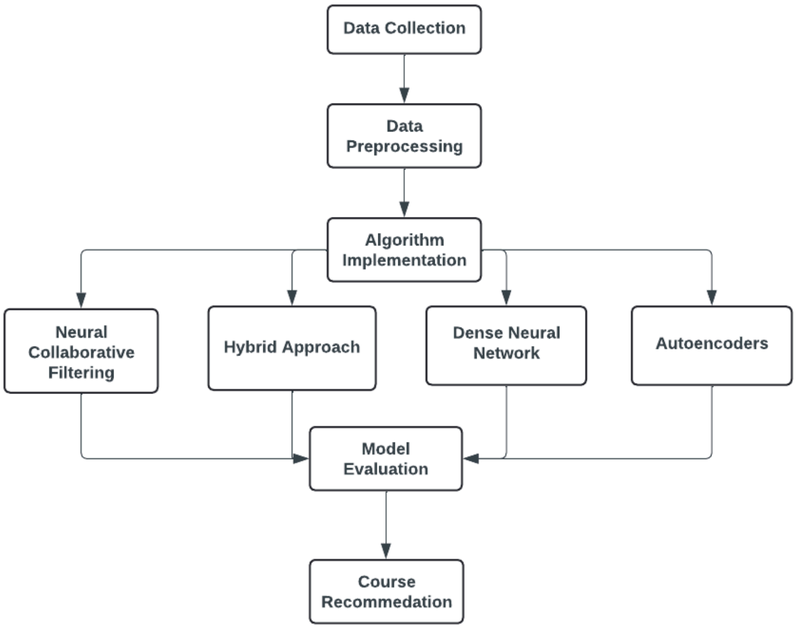
2. TensorFlow and Keras : The recommendation system's deep learning models are constructed and trained using TensorFlow, an open-source deep learning framework, and Keras, its high-level API. Effective model training is made possible by TensorFlow's broad neural network design capabilities and GPU support, while Keras offers a user-friendly and accessible API for developing and optimising the neural network layers utilised in the hybrid approach.

3. Pandas and NumPy : For preprocessing and data manipulation, pandas and numpy are crucial tools that enable effective management of big datasets. Pandas makes data analysis chores like data transformation, filtering, and cleaning easier. These processes are essential for getting the dataset ready for recommendation model training. In turn, NumPy offers quick and effective multidimensional array operations, which are crucial for managing the numerical calculations needed for machine learning.

4. Scikit-Learn :A robust machine learning package, Scikit-Learn covers fundamental methods like feature selection, data partitioning, and normalisation. As part of the hybrid method, Scikit-Learn is utilised to preprocess data, assess model performance, and apply specific machine learning algorithms, such as collaborative and content-based filtering, for this recommendation system.

5. Google Colab : Google Colab are great tools for experimenting, prototyping, and fine-tuning models. Developers may test different deep learning architectures, assess model performance, and visualise training progress with tools like Matplotlib and Seaborn thanks to these interactive environments that make it simple to iterate and visualise.

PROPOSED METHODOLOGY



- **Data Collection** is the initial phase, where the system gathers diverse data sources, including student demographics, academic history, interests, course descriptions, prerequisites, and interaction data like enrollments and ratings.

- **Data Preprocessing** follows, cleaning and preparing the collected data for analysis. This involves handling missing values, removing inconsistencies, and creating new features to enhance the recommendation process.

- **Algorithm Implementation** is the core of the system, where various algorithms are employed to generate recommendations. These algorithms include Neural Collaborative Filtering, Hybrid Approaches, Dense Neural Networks, and Autoencoders, each with its own strengths in capturing complex user-item relationships and extracting meaningful patterns.

- **Model Evaluation** is a crucial step to assess the performance of the chosen algorithms. Metrics like MSE, MAE and RMSE are used to evaluate the quality of recommendations.

- **Course Recommendation** is the ultimate goal, where the trained model generates personalized course suggestions for students.

WORKING MODULES

Neural Collaborative Filtering: Neural Collaborative Filtering (NCF): This model simulates random users in order to suggest courses based on user IDs. It depends on user-item interactions and mainly uses collaborative filtering to personalise suggestions. However, the model's flaws were not disclosed because of the inadequate data, indicating that NCF could need more detailed user data in order to obtain reduced errors. Without enough user history, NCF might not be as good at making suggestions as other models.

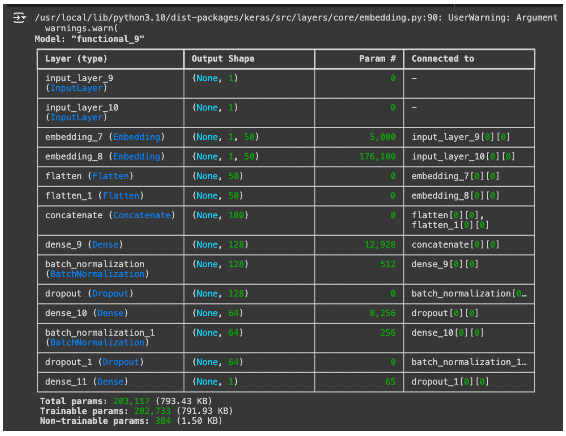
Content-and Collaborative-Based Hybrid Model: This hybrid technique suggests courses that are comparable to a given course by combining collaborative and content-based screening. It balanced user preferences and course similarities with moderate error rates. The hybrid model fared rather well in comparison to the other models, making it a flexible choice for people looking for courses on a particular subject.

Dense Neural Network: This model prioritises the top five related courses as recommendations after calculating similarity scores based on a given course. Dense Neural Networks typically seek to identify intricate patterns in data, which may provide more accurate rankings than NCF, even when precise error estimates are not disclosed.

Autoencoders: The Autoencoder model suggests courses depending on skills supplied as input by using skill-based recommendations. This model showed high accuracy and the lowest error rates. Autoencoders are the most successful for skill-based recommendations in this study because of their superior precision when compared to other models. Because of its low error metrics, Autoencoders may be able to offer highly appropriate courses based on particular talents, exceeding other models in terms of accuracy.

Implementation Screenshots :

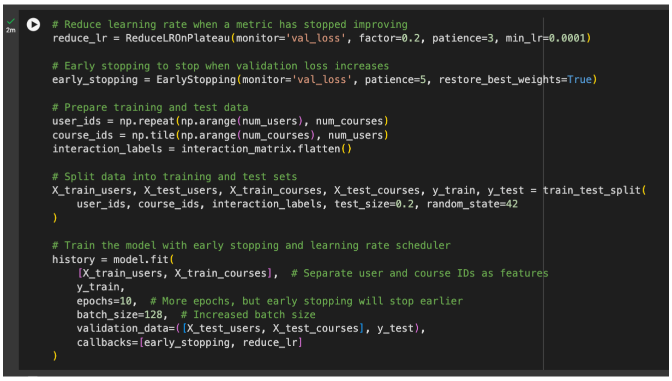
(i) Neural Collaborative Filtering :



```
/usr/local/lib/python3.10/dist-packages/keras/src/layers/core/embedding.py:90: UserWarning: Argument
  warnings.warn(
Model: "functional_9"
```

Layer (type)	Output Shape	Param #	Connected to
input_layer_9 (InputLayer)	(None, 1)	0	-
input_layer_10 (InputLayer)	(None, 1)	0	-
embedding_7 (Embedding)	(None, 1, 50)	5,000	input_layer_9[0][0]
embedding_8 (Embedding)	(None, 1, 50)	176,100	input_layer_10[0][0]
flatten (Flatten)	(None, 50)	0	embedding_7[0][0]
flatten_1 (Flatten)	(None, 50)	0	embedding_8[0][0]
concatenate (Concatenate)	(None, 100)	0	flatten[0][0], flatten_1[0][0]
dense_9 (Dense)	(None, 128)	12,928	concatenate[0][0]
batch_normalization (BatchNormalization)	(None, 128)	512	dense_9[0][0]
dropout (Dropout)	(None, 128)	0	batch_normalization[0…
dense_10 (Dense)	(None, 64)	8,256	dropout[0][0]
batch_normalization_1 (BatchNormalization)	(None, 64)	256	dense_10[0][0]
dropout_1 (Dropout)	(None, 64)	0	batch_normalization_1…
dense_11 (Dense)	(None, 1)	65	dropout_1[0][0]

```
Total params: 203,117 (793.43 KB)
Trainable params: 202,733 (791.93 KB)
Non-trainable params: 384 (1.50 KB)
```



```
# Reduce learning rate when a metric has stopped improving
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=3, min_lr=0.0001)

# Early stopping to stop when validation loss increases
early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)

# Prepare training and test data
user_ids = np.repeat(np.arange(num_users), num_courses)
course_ids = np.tile(np.arange(num_courses), num_users)
interaction_labels = interaction_matrix.flatten()

# Split data into training and test sets
X_train_users, X_test_users, X_train_courses, X_test_courses, y_train, y_test = train_test_split(
    user_ids, course_ids, interaction_labels, test_size=0.2, random_state=42
)

# Train the model with early stopping and learning rate scheduler
history = model.fit(
    [X_train_users, X_train_courses],  # Separate user and course IDs as features
    y_train,
    epochs=10,  # More epochs, but early stopping will stop earlier
    batch_size=128,  # Increased batch size
    validation_data=([X_test_users, X_test_courses], y_test),
    callbacks=[early_stopping, reduce_lr]
)
```

(ii) Hybrid Model :

```python
def collaborative_filtering_recommendations(course_title, top_n=5):
    # Ensure 'Course Rating' is numeric
    df['Course Rating'] = pd.to_numeric(df['Course Rating'], errors='coerce')

    # Fill NaN values with 0
    df['Course Rating'].fillna(0, inplace=True)

    # Create a pivot table for collaborative filtering
    ratings_matrix = pd.pivot_table(df, index='Course Name', columns='Skills', values='Course Rating', fill_value=0)

    # Check if the course is in the pivot table
    if course_title not in ratings_matrix.index:
        return pd.DataFrame()

    # Compute similarity matrix
    similarity_matrix = cosine_similarity(ratings_matrix)

    # Create a DataFrame for similarity matrix
    sim_df = pd.DataFrame(similarity_matrix, index=ratings_matrix.index, columns=ratings_matrix.index)

    # Get the similarity scores for the given course
    sim_scores = sim_df[course_title].sort_values(ascending=False)

    # Exclude the course itself
    sim_scores = sim_scores.drop(course_title)

    # Get the top_n most similar courses
    similar_courses = sim_scores.head(top_n).index

    return df[df['Course Name'].isin(similar_courses)]
```

```python
# Content-Based Filtering
def content_based_recommendations(course_title, top_n=5):
    if course_title not in df['Course Name'].values:
        return pd.DataFrame()  # Return empty DataFrame if course is not found

    # Create TF-IDF vectors for the 'Course Description'
    tfidf_vectorizer = TfidfVectorizer(stop_words='english')
    tfidf_matrix = tfidf_vectorizer.fit_transform(df['Course Description'])

    # Compute cosine similarity
    cosine_sim = linear_kernel(tfidf_matrix, tfidf_matrix)

    # Get the index of the course that matches the title
    idx = df.index[df['Course Name'] == course_title].tolist()[0]

    # Get similarity scores
    sim_scores = list(enumerate(cosine_sim[idx]))
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)

    # Get the top_n most similar courses
    sim_scores = sim_scores[1:top_n+1]
    course_indices = [i[0] for i in sim_scores]

    return df.iloc[course_indices]
```

```python
[28] # Hybrid Recommendation Function
     def get_hybrid_recommendations(course_title, top_n=5):
         content_based = content_based_recommendations(course_title, top_n)
         collaborative = collaborative_filtering_recommendations(course_title, top_n)

         # Combine recommendations
         combined_recommendations = pd.concat([content_based, collaborative]).drop_duplicates()

         return combined_recommendations
```

(iii) Dense Neural Network :

```python
# Define model parameters
input_dim = embedding_dim
hidden_dim1 = 512
hidden_dim2 = 256
output_dim = input_dim  # Output dimension same as input dimension for reconstruction

# Define the Dense network model
inputs = Input(shape=(input_dim,))
x = Dense(hidden_dim1, activation='relu')(inputs)
x = Dense(hidden_dim2, activation='relu')(x)
outputs = Dense(output_dim, activation='sigmoid')(x)

model = Model(inputs, outputs)

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy')

# Train the model
history = model.fit(X_train, X_train, epochs=20, batch_size=32, validation_split=0.2)
```

(iv) Autoencoders :

Model: "functional_9"

Layer (type)	Output Shape	Param #
input_layer_13 (InputLayer)	(None, 1)	0
dense_23 (Dense)	(None, 64)	128
batch_normalization_2 (BatchNormalization)	(None, 64)	256
dropout_4 (Dropout)	(None, 64)	0
dense_24 (Dense)	(None, 64)	4,160
dropout_5 (Dropout)	(None, 64)	0
dense_25 (Dense)	(None, 1)	65

Total params: 4,609 (18.00 KB)
Trainable params: 4,481 (17.50 KB)
Non-trainable params: 128 (512.00 B)

```python
# Define the dimensions
input_dim = X_train.shape[1]  # Input dimension (skills)
encoding_dim1 = 64  # First encoding dimension
encoding_dim2 = 32  # Second encoding dimension (bottleneck layer)
decoding_dim1 = 64  # First decoding dimension (mirroring the encoder)
dropout_rate = 0.2  # Dropout rate to avoid overfitting

# Define the input
input_layer = layers.Input(shape=(input_dim,))

# Encoder part
encoded = layers.Dense(encoding_dim1, activation='relu')(input_layer)
encoded = layers.BatchNormalization()(encoded)  # Add batch normalization (introduces non-trainable params)
encoded = layers.Dropout(dropout_rate)(encoded)  # Add dropout for regularization


# Freeze the bottleneck layer
encoded.trainable = False  # Freezing this layer, so its weights won't be updated

# Decoder part
decoded = layers.Dense(decoding_dim1, activation='relu')(encoded)
decoded = layers.Dropout(dropout_rate)(decoded)  # Add dropout for regularization
decoded = layers.Dense(input_dim, activation='sigmoid')(decoded)  # Reconstruct original input

# Build the model
autoencoder = Model(inputs=input_layer, outputs=decoded)

# Compile the model
autoencoder.compile(optimizer='adam', loss='mean_squared_error')

# Summary of the model
autoencoder.summary()

# Train the model
history = autoencoder.fit(X_train, X_train,
                          epochs=10,
                          batch_size=32,
                          shuffle=True,
                          validation_data=(X_test, X_test))
```

RESULTS

Sample Recommendations obtained using different algorithms :

(i) Neural Collaborative Filtering :

```
[24] # Predict interaction scores for courses
predicted_scores = model.predict([user_ids, course_ids]).flatten()

# Create a DataFrame for predicted scores
predicted_df = pd.DataFrame({
    'User': np.repeat(np.arange(num_users), num_courses),
    'Course': np.tile(data['Course Name'], num_users),
    'Predicted Score': predicted_scores
})

# Recommend top courses for a specific user
user_id = 10
top_courses = predicted_df[predicted_df['User'] == user_id].sort_values(by='Predicted Score', ascending=False).head(10)
print(f"Top recommended courses for user {user_id}:\n", top_courses[['Course', 'Predicted Score']])

# Predictions on test data
test_preds = model.predict([X_test_users, X_test_courses]).flatten()
```

```
11007/11007 ━━━━━━━━━━━━━━━━━━━━ 20s 2ms/step
Top recommended courses for user 10:
                                              Course  Predicted Score
35265  Fundamentals of Funding, with Goldman Sachs 10...         0.507430
37792  Introduction to PicsArt for Social Media Marke...         0.506507
35772               Object Oriented Programming in Java         0.506494
35687  Foundations of Teaching for Learning: Developi...         0.506468
38707      Taxation of Business Entities I: Corporations         0.506451
35631  Build a Deep Learning Based Image Classifier w...         0.506412
35296           ART of the MOOC: Public Art and Pedagogy         0.506355
35537    Create Charts and Dashboard using Google Sheets         0.506263
36446  Soul Beliefs: Causes and Consequences - Unit 3...         0.506219
37613  Finding Hidden Messages in DNA (Bioinformatics I)         0.506155
2202/2202 ━━━━━━━━━━━━━━━━━━━━ 4s 2ms/step
```

(ii) Hybrid Model :

```
# Example usage
course_title = 'Cyber Threats and Attack Vectors'
recommendations = get_hybrid_recommendations(course_title)
print(recommendations)
```

```
                                            Course Name  \
459   Detecting and Mitigating Cyber Threats and Att...
3208         Introduction to Cybersecurity for Business
931                      Proactive Computer Security
2594  Cybersecurity Capstone:  Breach Response Case ...
1173                  "Making" Progress Teach-Out
1799                  MongoDB Aggregation Framework
1958                  Modern JavaScript: ES6 Basics
2331                             Modern Art & Ideas
3494  Modern Robotics, Course 1:  Foundations of Rob...

                      University Difficulty Level Course Rating  \
459   University of Colorado System     Intermediate           4.7
3208  University of Colorado System         Beginner           4.7
931   University of Colorado System         Beginner           4.4
2594                            IBM         Beginner           4.7
1173               Emory University         Beginner           4.2
1799                   MongoDB Inc.         Advanced           4.5
1958       Coursera Project Network         Beginner           4.4
2331        The Museum of Modern Art         Beginner           4.7
3494        Northwestern University     Intermediate           4.6
```

(iii) Dense Neural Network :

```
[35] # Get the learned representations
learned_representations = model.predict(embeddings_matrix)

# Function to recommend courses
def recommend_courses(course_index, top_n=5):
    course_rep = learned_representations[course_index]
    similarities = cosine_similarity([course_rep], learned_representations).flatten()
    similar_courses = np.argsort(similarities)[::-1][1:top_n+1]

    print(f"Top {top_n} recommended courses for '{data.iloc[course_index]['Course Name']}':")
    for idx in similar_courses:
        print(f"Course Name: {data.iloc[idx]['Course Name']}, University: {data.iloc[idx]['University']}, Similarity: {similarities[idx]:.4f}")

# Example usage: Recommend courses similar to the course at index 0
recommend_courses(1, top_n=5)
```

```
111/111 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
Top 5 recommended courses for 'Business Strategy: Business Model Canvas Analysis with Miro':
Course Name: Clustering Geolocation Data Intelligently in Python, University: Coursera Project Network, Similarity: 0.9306
Course Name: Neural Networks and Deep Learning, University: DeepLearning.AI, Similarity: 0.9266
Course Name: Internet of Things: How did we get here?, University: University of California San Diego, Similarity: 0.9253
Course Name: Quantitative Foundations for International Business, University: University of London, Similarity: 0.9244
Course Name: Action-Driven Business Plan: From the �Classroom� to the World, University: Technion - Israel Institute of Technology, Similarity: 0.9228
```

(iv) Autoencoders :

```
# Example: Recommend a course based on a test skill
skill_test_example = X_test[0]  # Use a skill from the test set
recommended_course = recommend_course(skill_test_example)
print(f"Recommended Course: {recommended_course}")
```

```
23/23 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
23/23 ━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step
Recommended Course: ['"Making" Progress Teach-Out']
```

Errors obtained by models

Models	MSE	MAE	RMSE
NCF	0.2500	0.4999	0.5000
Hybrid	0.3	0.4	0.55
DNN	0.0544	0.1908	0.2332
Autoencoders	**0.00088**	**0.02676**	**0.02972**

CONCLUSION AND FUTURE WORKS

The course recommendation system offers a potent solution for individualised education by utilising cutting-edge Deep Learning and Hybrid approaches. Through an analysis of multiple course features, including rating, skills, difficulty level, and in-depth descriptions, the system makes intelligent matches between students and courses that suit their interests, learning objectives, and academic background. The Autoencoders model has demonstrated impressive accuracy with minimal error values: MAE of 0.1908, MSE of 0.0544, and RMSE of 0.2332. The recommendations are not only accurate but also flexible enough to respond to changing student demands using Autoencoders, DNN, Hybrid approaches, and Neural Collaborative Filtering. By giving students the ability to make informed decisions, this method enhances learning and supports their goals for both their academic and professional careers. As a result of ongoing improvement and learning, the system becomes more efficient and supports a more customised and engaging learning experience.

Despite showing encouraging results, the suggested course suggestion system provides a number of opportunities for further research. Including real-time feedback mechanisms to dynamically modify recommendations in response to student success and changing preferences is one such approach. The recommendations' personalisation could be further improved by including social learning elements into the system by taking peer recommendations and cooperative learning opportunities into account. Additionally, investigating how to optimise recommendations over time using reinforcement learning approaches based on performance metrics and student involvement is a potential direction. The objective is to develop an even more potent and efficient tool that enables students to make wise decisions and accomplish their academic objectives by iteratively improving the system and adding these improvements.

REFERENCES

1. Li, L., Zhang, Z., & Zhang, S. (2021). Hybrid algorithm based on content and collaborative filtering in recommendation system optimization and simulation. Scientific Programming, 2021(1), 7427409.

2. He, X., Du, X., Wang, X., Tian, F., Tang, J., & Chua, T. S. (2018). Outer product-based neural collaborative filtering. arXiv preprint arXiv:1808.03912.

3. Arık, A., Okyay, S., & Adar, N. (2021). Hybrid course recommendation system design for a real-time student automation application. Avrupa Bilim ve Teknoloji Dergisi, (26), 85-90.

4. Mustafeez ul Haque, M., Kotaiah, B., & Ahmed, J. (2024). A Comprehensive Review of Course Recommendation Systems for MOOCs. International Journal of Computing and Digital Systems, 16(1), 1-15

5. Perez, J. C. S., Manrique, R. F., Mariño, O., Vásquez, M. L., & Cardozo, N. (2022). A course hybrid recommender system for limited information scenarios. Journal of Educational Data Mining, 14(3), 162-188.

6. Akhtar, A. (2020). Implementation of course recommender system for virtual university of pakistan. arXiv preprint arXiv:2008.07867.

7. Sajiharan, S., & Singh, K. P. (2022). Deep learning enabled course recommendation platform.

8. Jatoi, A., Sami, M., Nawaz, M., & Baloch, J. (2022). An Intuitive Implementation of Course Recommendation System Based on Learner's Personality. Acta Scientific COMPUTER SCIENCES Volume, 4(5).

9. Ko, H., Lee, S., Park, Y., & Choi, A. (2022). A survey of recommendation systems: recommendation models, techniques, and application fields. Electronics, 11(1), 141

10. Wu, L. (2021). Collaborative filtering recommendation algorithm for MOOC resources based on deep learning. Complexity, 2021(1), 5555226.

11. Zhu, Q. (2021). Network Course Recommendation System Based on Double-Layer Attention Mechanism. Scientific Programming, 2021(1), 7613511.

12. Rendle, S., Krichene, W., Zhang, L., & Anderson, J. (2020, September). Neural collaborative filtering vs. matrix factorization revisited. In Proceedings of the 14th ACM Conference on Recommender Systems (pp. 240-248).

13. Shafqat, W., & Byun, Y. C. (2020). A context-aware location recommendation system for tourists using hierarchical LSTM model. Sustainability, 12(10), 4107.

14. Geng, L. (2022). The recommendation system of innovation and entrepreneurship education resources in universities based on improved collaborative filtering model. Computational Intelligence and Neuroscience, 2022(1), 7228833.

15. Algarni, S., & Sheldon, F. (2023). Systematic Review of Recommendation Systems for Course Selection. Machine Learning and Knowledge Extraction, 5(2), 560-596.

16. Saini, K., & Singh, A. (2024). A content-based recommender system using stacked LSTM and an attention-based autoencoder. Measurement: Sensors, 31, 100975.

17. Kumar, V., Khattar, D., Gupta, S., Gupta, M., & Varma, V. (2017, September). Deep Neural Architecture for News Recommendation. In CLEF (Working Notes).

18. Zhao, Y., Yin, C., Wang, X., Chai, Y., Chen, H., & Ouyang, Y. (2024). Research of online courses recommendation based on deep learning. Procedia Computer Science, 242, 219-227.
19. Li, Y. (2022). A graph convolution network based on improved density clustering for recommendation system. Information Technology and Control, 51(1), 18-31.

20. Zeldes, Y., Theodorakis, S., Solodnik, E., Rotman, A., Chamiel, G., & Friedman, D. (2017). Deep density networks and uncertainty in recommender systems. arXiv preprint arXiv:1711.02487.

Xpxp