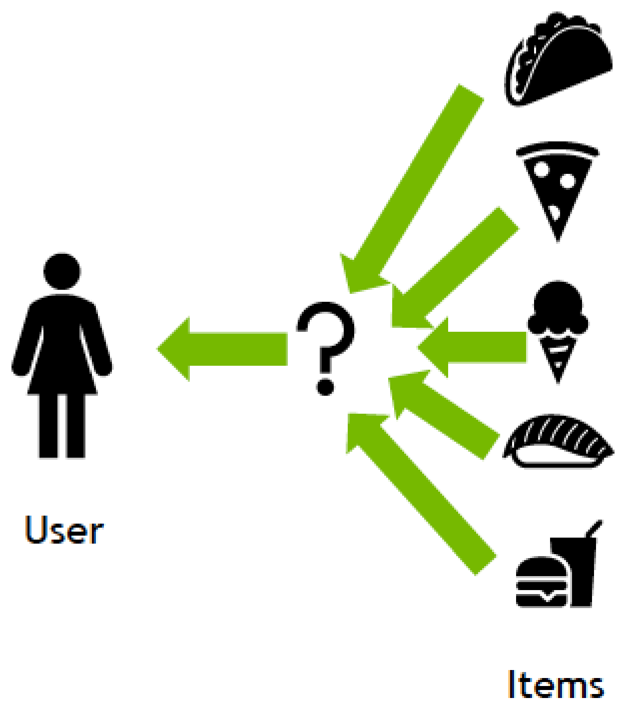


CONTEXT AWARE RECOMMENDATION SYSTEMS USING DEEP LEARNING

DEEP LEARNING MODEL: AUTOENCODER CUM NEURAL NETWORK

SHRUTHI MOHAN (21011101125) | AI&DS-B | 25-10-2023

ABSTRACT

Context-aware recommender systems have emerged as a powerful approach to enhance the quality and relevance of personalized recommendations. These systems consider contextual information, such as time and place, in addition to user-item interactions when making recommendations. To implement such systems, deep learning models are employed. This paper introduces a novel approach that combines a neural network with an autoencoder, where the autoencoder serves as a preprocessor. The autoencoder encodes the input data into a compact form, simplifying the training process and improving the convergence of deep neural networks. By leveraging this approach, the recommender system can provide more accurate and context-aware recommendations, enhancing the overall user experience. This methodology represents a promising direction for developing highly effective and personalized recommendation systems that adapt to users' dynamic needs and preferences.

INTRODUCTION

A context-aware recommender system is a type of recommendation system that takes into account various contextual factors when making recommendations to users. Traditionally, recommendation systems have relied primarily on user-item interactions, such as user ratings or purchase history, to provide personalized recommendations. However, in a context-aware recommender system, additional information, such as time and place, is considered to enhance the quality and relevance of the recommendations.

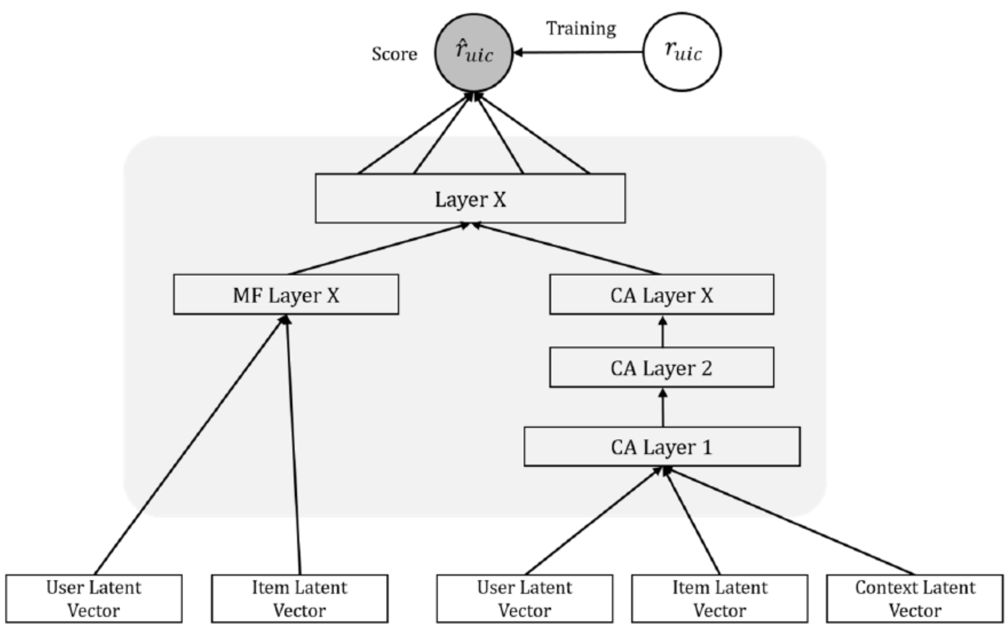
The combination of a neural network and an autoencoder in a context-aware recommender system is a powerful technique for enhancing recommendation quality by considering contextual information. The autoencoder's role as a preprocessor simplifies the learning process, making it more efficient and effective in handling the complexity of deep neural networks while also preserving the essential features of the data. This approach can lead to more accurate and personalized recommendations that take into account not only user-item interactions but also the dynamic context in which users are making decisions.

When there are similar users or items, there can be similar contextual information. In such cases, it is possible to improve the existing recommender system's performance by reflecting the context features. Here we propose a context-aware recommender system that integrates a neural network and an autoencoder. MF, which shows great performance in the existing context-aware recommender system cannot resolve nonlinear problem such as XOR. We can use deep learning to resolve issues that could not be classified in the past. In addition, to reflect contextual features other than user and item characteristics, an autoencoder is combined with a neural network, enabling us to improve the accuracy of prediction scores while extracting contextual features.

The concepts used in this work:

1) **Context-Aware Recommender System**: A traditional recommender system primarily relies on user-item interactions, such as user preferences or historical data, to make recommendations.
2) **Neural Network and Autoencoder Combination:** Deep learning models like neural networks are used to learn and predict user preferences based on various inputs, including contextual information. An autoencoder is a type of neural network used in unsupervised learning. It is composed of an encoder and a decoder.
3) **Autoencoder as a Preprocessor:** By using an autoencoder to encode the input data into a more compact form, it reduces the dimensionality of the data while retaining important features. This compressed representation simplifies the training process for the neural network because it focuses on the most relevant information, which can lead to improved model convergence.

BASE MODEL ARCHITECTURE AND IMPLEMENTATION



The proposed model is formed by connecting multiple AEs and NN. The input data include multiple feature vectors. The input and output data for the autoencoder include user vectors, item vectors and context vectors. The neural network is trained by using score as targets from the hidden layer in the middle. When the dataset has n (x, y) pairs, x represents the data record containing users, items and contextual dimension, and y represents the score given by the user. Each field of x is expressed as a binary vector with one-hot encoding. This is similar to that of factorization machine.

$$L_{AE} = \min_{\theta} \sum_{v \in V} \|X - X'\|^2 + \alpha \sum_{l} \left(\|w_{AE}\|^2 + \|b_{AE}\|^2 \right)$$

To prevent overfitting, regularization terms are added in Formula.

$$L_{NN} = \min_{\theta} \sum_{y \in Y} (y_{uic} - \hat{y}_{uic})^2 + \beta \sum_{l} \left(\|w_{NN}\|^2 + \|b_{NN}\|^2 \right)$$

MF is a technique that decomposes a user-item matrix into a user latent matrix, U, and an item latent matrix, V. The product of the user latent matrix and the item latent matrix is trained to resemble the values in the preference matrix. The matrices U and V are obtained by training the product of the user latent matrix and the item latent matrix to be similar to the values in the preference matrix. Finally, the predicted value of Ro is multiplied by the matrices U and V that are most similar to R, and Ro is the matrix filled with all the values in the matrix. In this way, MF can predict preferences that have not been evaluated by the user.

To train the proposal method, we apply Adaptive Moment Estimation (Adam). Adam is a popular optimization technique.

Algorithm 1 Algorithm of Proposed Model

Input: User vector *U*, Item vector *I*, Context vector C_n, Learning rate *N*, number of iterations *k*

Output: parameter w_{AE}, w_{NN}, b_{AE}, b_{NN}, Rating vector *Y*

Randomly initialize parameters w_{AE}, w_{NN}, b_{AE}, b_{NN}

for $i = 1$ to k **do**

  for $u \in U$ **do**

    Update w_{AE}, b_{AE} using L_{AE}'

    Update w_{NN}, b_{NN} using L_{NN}'

  end for

end for

DATASET

- Restaurant dataset: https://github.com/irecsys/CARSKit/blob/master/context-aware_data_sets/Restaurant_TijuanaRestaurant.zip
- Movies dataset: https://github.com/irecsys/CARSKit/blob/master/context-aware_data_sets/Movie_DePaulMovie.zip

- Music dataset: https://github.com/irecsys/CARSKit/blob/master/context-aware_data_sets/Music_InCarMusic.zip
- Instacart Market Analysis: https://www.kaggle.com/c/instacart-market-basket-analysis

For clear and easy analysis, the keras mnist dataset was used. The images were preprocessed by the autoencoders by building a separate model for encoding and decoding. The preprocessed output was then given to the neural network model.

RESULTS

Table 2. Performance comparison result in Precision.

Precision@10	Music	Restaurant	Movie
UserKNN	0.010082	0.062889	0.063732
SVD++	0.045452	0.062174	0.059719
CAMF_CI	0.024234	0.050423	0.058384
ItemSplitting-BiasedMF	0.045365	0.06249	0.054859
CSLIM_CI	0.003996	0.012763	0.004598
PMF	0.048151	0.068362	0.06807
FM	0.010663	0.047136	0.027956
NN based model	0.050387	0.059913	0.076811
Proposed model	**0.056760**	**0.072748**	**0.102263**

Classical Autoencoder:

```
Epoch 49/50
235/235 [==============================] - 1s 6ms/step - loss: 0.6533 - val_loss: 0.6520
Epoch 50/50
235/235 [==============================] - 2s 6ms/step - loss: 0.6521 - val_loss: 0.6508
```

Autoencoder cum Neural Network:

```
Epoch 14/15
235/235 [==============================] - 1s 5ms/step - loss: 0.0106 - val_loss: 0.0102
Epoch 15/15
235/235 [==============================] - 1s 5ms/step - loss: 0.0105 - val_loss: 0.0102
```

Using autoencoders as preprocessors and neural network for modelling, we can get best accuracy than the normal autoencoders used for modelling.

FURTHER REFINEMENTS

Cons of the Base Model:

- Needs high contextual dimensions for good accuracy
- Needs to reflect reliability- unaware of what context affects the user's preference
- It struggles to capture short-term preference changes

1) **FOR HIGH CONTEXTUAL DIMENSIONS**

- Transfer learning can be incorporated using pre-trained datasets.
- Reinforcement Learning: reward functions that explicitly consider contextual information can be created. This encourages the model to learn from contextual cues and make better recommendations.

2) **FOR RELIABILITY**

- Conversational Systems can be incorporated to help users understand the rationale behind the recommendations and make them more trustworthy.

  We can use already built systems like Google LaMDA (2023) but the UI needs to be user-friendly and shouldn't make them feel like they are attending a survey

3) **FOR LONG-TERM PREFERENCE CHANGES**

  Transfer learning can be incorporated using pre-trained datasets.

- Reinforcement Learning: reward functions that explicitly consider contextual information can be created. This encourages the model to learn from contextual cues and make better recommendations.

CONCLUSION

We propose a deep learning-based context-aware recommender system for capture context feature. Also, a method for learning sparse data that employs autoencoder and neural network is proposed to overcome, the data sparsity problem faced by the existing context-aware recommender system. To learn contextual features, a method for integrating autoencoder for users, items and context is proposed. The model combines autoencoder, a non-supervised learning model that finds input characteristics by learning model and neural network, a supervised learning method that uses score output as targets. Users, items and context dimension are set as inputs and outputs so that the scores reflect each one's characteristics and relationships. This method can be easily applied when there is additional contextual information. The performance results of the proposed method show that it performs better on all datasets compared to most of the comparisons. In addition, it performs better than the comparison target for a dataset with a large amount of contextual information, since the proposed model further reduces the data sparsity problem. Because the proposed model is further reduced to the data sparsity problem.

REFERENCES

- Deep Learning-Based Context-Aware Recommender System Considering Contextual Features (2021): https://www.mdpi.com/2076-3417/12/1/45
- Deep Learning-Based Context-Aware Recommender System Considering Change in Preference (2023): https://www.mdpi.com/2079-9292/12/10/2337
- Leveraging Large Language Models in Conversational Recommender Systems: https://arxiv.org/abs/2305.07961