JOURNAL OF INFORMATION SCIENCE AND ENGINEERING, XXX-XXX (2022)
DOI: 10.6688/JISE.20200X_36(X).00XX

An Attentive Hybrid Filtering Network for News Recommendation

YUN-CHENG CHOU, DUEN-REN LIU,*

Institute of Information Management, National Yang Ming Chiao Tung University, Hsinchu 300, Taiwan

** Corresponding author: Professor Duen-Ren Liu dliu@nycu.edu.tw*

Hybrid filtering recommendation models play an important role in the field of recommender systems by helping users to find preferred items and make decisions effectively. Hybrid filtering generally integrates strategies from collaborative filtering and content-based filtering together to reinforce their advantages and alleviate their disadvantages. Most existing works of hybrid filtering focus on modeling user-item behavior data and contextual data, such as text or image. However, few studies consider all three aspects: user-item behavior data, contextual data and item-item co-view structure information. In this study, we propose a novel attentive hybrid filtering network (AHFN) that integrates latent factors, word- and sentence-level contextual information, and item-item co-view relationship to construct item features. Additionally, user features are attentively constructed based on their recent behaviors. Specifically, we apply a combination of the latent factor model (LFM), convolutional neural network (CNN), sentence bidirectional encoder representations from transformers (SBERT) and graph convolutional network (GCN) to build these user and item features. A prediction network then takes these user and item features as input to predict user preferences. We conducted experiments using data collected from an online news website in Taiwan to demonstrate the top-*N* recommendation performance of proposed model. The experimental results show that our proposed model outperforms several baselines with statistical significance.

Keywords: Recommendation Systems, Hybrid Filtering, Latent Factor Model, Convolutional Neural Network, Bidirectional Encoder Representations from Transformers, Graph Convolutional Network

1. INTRODUCTION

The recommendation or recommender system is an information filtering technique that aims to extract valuable and preferred information for users from the large amounts of data. The system helps users to find information they really need in a more efficient way, saving their time and addressing the so-called information overload problem. Nowadays, the recommendation systems are widely applied in various industries and continue to keep be a focus of research literature. For example, the recommendation system for movie [30, 32], music [1, 13], products [10, 20]. In addition, there are researches for online news recommendations [12, 33], where the items to recommended to users are mainly consist of words, sentences, and paragraphs.

Recommendation systems, based on the techniques they adopt to construct or train their model can mainly be classified into three categories: collaborative filtering, content-based filtering, and hybrid filtering models. Collaborative filtering models, such as the latent factor model (LFM) [28] or matrix factorization (MF) [19], are trained primarily using the interactions or the historically data between users and items. These models discover the hidden structure of user-item interactions and construct features that represent the characteristics of items and the preferences of users. As a result, the system can predict the user preferences for unobserved items and make relevant recommendations. Content-based models, on the other hand, construct features for users and items mainly based on the content of items (e.g., words, sentences, or paragraphs for news recommendation models). These models leverage the valuable information extracted from items to enhance system's abilities to understand users' preferences, thereby improved recommendation performance. Hybrid filtering model integrate collaborative filtering and content-based filtering to overcome the limitations of using either approach alone.

In the development of modern machine learning techniques, advanced models such as Bidirectional Encoder Representations from Transformers (BERT) [11] help machines better understand semantic and knowledge hidden in text, while Graph Convolutional Network (GCN) are effective at recognizing hidden relationship in graph-structured data [43]. In the field of recommendation system research, many studies have been proposed to improve the performance of hybrid filtering models. However, few studies

focus on integrating BERT and GCN. In this study, we propose a novel hybrid filtering recommendation model named attentive hybrid filtering network (AHFN). This model constructs user and item features using latent factors and also incorporates word-level and sentence-level information from CNN and BERT. Additionally, items' co-view relationships are modeled in an undirected graph with hidden features extracted using GCN model. Finally, comprehensive attentive user features are constructed using an attention mechanism to further enhance performance in top-*N* recommendation tasks. To demonstrate the effectiveness of the proposed model, we conduct experiments using data collected from a well-known online news website in Taiwan. We summarize the contributions of this study as follow:

(1) A novel hybrid filtering recommendation model is proposed, integrating matrix factorization, CNN, BERT, and GCN to construct user and item features, this allows the model to consider latent factors, word-level and sentence-level contextual information, and items' co-view relationships.
(2) We perform experiments using data collected from a well-known online news website in Taiwan and compare the proposed model with several baselines. The experimental results show the proposed model outperforms baselines with statistical significance.

The remainder of this paper is organized as follows: Section 2 provides a brief overview of related literature. Section 3 mainly introduces the proposed AHFN recommendation model. Section 4 describes the evaluation methodology and demonstrates the top-*N* recommendation performance of the proposed model compared with several baselines. Finally, Section 5 offers conclusions and suggestions for future directions.

2. RELATED WORK

2.1. Recommender System

With the rapid development of computer science and the World Wide Web, people receive a larger amount of information from various devices every day. Much of this information may be irrelevant, making it difficult for people to make decisions effectively [42]. This is referred to as the information overload problem. Recommender or recommendation systems aim to overcome the information overload problem by using statistical or machine learning models with historical data. For example, large amounts of historical data stored in databases reflect interactions between users and items. A value of 1 can indicate that a user has interacted with an item (a positive sample), while a value of 0 indicates the opposite (a negative sample) [29]. A recommendation model can be trained or constructed using such implicit data and then applied to predict users' preferences for unknown items, recommending only the items they are likely to prefer. The advantage of implicit data is that it is easy to collect, allowing system owners to implement the system at a lower cost. On the other hand, the model can also be trained or constructed using explicit data, such as 5-star ratings**,** where a user expresses their preference by giving a rating of 5 if they like the item very much, and a rating of 1 to indicate disappointment [35].

Recommendation systems can generally be categorized into various types: collaborative filtering, content-based filtering, and hybrid filtering models**,** depending on the information model adopted to generate recommendations for users [17, 24]. Collaborative filtering models provide recommendations for a user based on information collected from other users or items. Content-based filtering models generate recommendations based on the contextual features extracted from items. Hybrid filtering models integrate the other two types into one, thereby mitigating the disadvantages of both collaborative and content-based filtering models.

2.2. Collaborative Filtering

The collaborative filtering (CF) can be further categorized into neighborhood model and latent factor model [5, 15]. In the conventional neighborhood CF models, recommendations are made for a specific user based on the information or opinions gathered from other users or items. The criteria for selecting the information usually based on the correlations between users and items. For example, user-based CF

models analyze correlations between users by using Pearson correlation coefficient or cosine similarity [6, 35], and predict preferences for unobserved items by applying weighted-sum. This sum is based on target user's average historical preferences and the preferences of positively correlated neighbors. Similarly, item-based CF models analyze correlations between items and make predictions for a target user on unobserved items based on similar items that the user has rated in the past [31, 36].

The latent factor model generally adopted matrix factorization (MF) with a more holistic goal of learning latent features or factors that explain observed user-item preferences [19]. The basic form of MF assumes the existence of a user-item preference matrix denote as $R \in \mathbb{R}^{|U|\times|I|}$ where U and I represent the set of users and items, respectively. This matrix R can be decomposed into two submatrices $P^{f\times|U|}$ and $Q^{f\times|I|}$, which represent f-dimensional latent factor vectors for users and items. Consequently, matrix R can be approximated as $R = P^T Q$. These latent factors for users and items are learned by optimizing the objective function shown in Eq. (1):

$$\min_{p^*,q^*} \sum_{r_{u,i}\ is\ obseverd} \left(r_{u,i} - p_u^T q_i\right)^2 + \lambda(\|p_u\|^2 + \|q_i\|^2) \tag{1}$$

where $r_{u,i}$ is observed feedback for user u on item i, and prediction can be obtained using inner product of user and item latent factor vectors, denoted as $p_u^T q_i$. The λ is a weight for the regularization term to prevent over fitting. Following to the success of deep neural network (DNN), a modern latent factor model called neural collaborative filtering (NCF) has proposed [14]. NCF learns latent features for users and items using a DNN architecture and formulates the predictive model as shown in Eq. (2):

$$\hat{y}_{ui} = f\left(P^T v_u^U, Q^T v_i^I | P, Q, \theta_f\right) \tag{2}$$

$$f(P^T v_u^U, Q^T v_i^I) = \phi_{out}\left(\phi_X\left(\dots \phi_2\left(\phi_1(P^T v_u^U, Q^T v_i^I)\right)\dots\right)\right) \tag{3}$$

where v_u^U and v_i^I are the one-hot encoded user and item vectors, which are used as the model inputs so that model can identify the features for user and item. The matrices $P \in \mathbb{R}^{|U|\times K}$ and $Q \in \mathbb{R}^{|I|\times K}$ denote the latent factor matrices for users and items, respectively. The θ_f represents the trainable model parameters of the interaction function f. The function f is defined as a multi-layer neural network, as shown in Eq. (3), where ϕ_{out} and ϕ_x denote the output layer and the x-th CF layer in a neural network architecture with a total of X layers. In the NCF framework, it is easy to construct a model that combines both generalized matrix factorization (GMF) and multi-layer perceptron (MLP), as shown in Eq. (4), to capture linear and non-linear user-item behavior structure in the data.

$$\hat{y}_{ui} = \sigma(h^T a(p_u \odot q_i + W[p_u\ q_i]^T + b)) \tag{4}$$

where $\odot$ denotes the element-wise product for user and item latent factors in GMF. W and b is the trainable parameters and bias for the MLP, respectively.

Although collaborative filtering models are popular for modern recommendation systems, they suffer from the data sparsity problem, where there is insufficient user feedback to train the models, and cold-start problem, where the system may be unable to provide accurate recommendations if there is insufficient data for new users or items [16, 37].

2.3. Content-based Filtering

Content-based filtering (CBF) models recommend items to users based on the analysis of the relationship between the target user's profile and the content of candidate items. Typically, popular techniques from the field of information retrieval (IR) are adopted by conventional CBFs to analyze such relationships—for example, the vector space model, term frequency-inverse document frequency (TF-IDF), and cosine similarity [26]. Beyond that, a generative probabilistic model called Latent Dirichlet Allocation (LDA) has been proposed [8]. LDA assumes that documents are represented as random mixtures over K latent topics, and each topic is constructed as a distribution over words. The model provide significant benefits to the field of recommendation system [3, 7].

Due to the successful development of deep neural networks, an efficient word representation method called Word2Vec was proposed [27]. Word2Vec maps each word in a corpus into a fixed-dimensional vector space using a shallow neural network and either a continuous bag-of-words or skip-gram architecture. Since neighboring words in a sentence are considered during the training phase, the semantic structures of words in the corpus can be discovered, which helps with contextual analysis. Furthermore, following the success of convolutional neural networks (CNNs) in the field of image recognition [2, 21], the idea of combining Word2Vec and CNNs in recommendation systems has attracted significant attention. Such work typically constructs item features by first representing an item as a set of consecutive words in the form of Word2Vec vectors, and then applying CNNs to extract semantic structures as item features [20, 40].

A modern language model named Bidirectional Encoder Representations from Transformers (BERT) has been proposed. The architecture of BERT is a multi-layer bidirectional Transformer encoder [38]. It performs masked language modeling and next sentence prediction during the pre-training step to construct sequence embeddings for the input sequence. The downstream tasks can fine-tune the initial parameters of BERT using task-related labeled data [11]. Sentence-BERT (SBERT) was further proposed to construct semantically meaningful sentence embeddings, rather than simply averaging all token embeddings in a sentence or using the embedding of the CLS token to represent the sentence [34].

Compared to CF models, CBF models alleviate the new item problem by using an item's content information to find appropriate user profiles for recommendations. However, CBF suffers from the problem of insufficient content information [23], making it difficult for the models to understand the characteristics of items and construct user profiles.

2.4. Hybrid Filtering

Hybrid filtering recommendation models integrate two or more strategies of recommendation systems to reinforce advantages and reduce disadvantages [9]. Collaborative filtering (CF) and content-based filtering (CBF) are the most commonly integrated approaches in a hybrid filtering model for making recommendations. CF can provide appropriate user and item features for predicting preferences when items' contextual information is limited, while CBF can offer better recommendations when interactions between users and items are not available.

The collaborative topic modeling (CTM) [39] is one of the most well-known hybrid filtering models. It represents an item *j*'s latent vector in the form $v_j = \epsilon_j + \theta_j$, where ϵ_j is a latent factor offset and θ_j represents item *j*'s latent topic proportions constructed using LDA. Then, CTM assumes a preference r_{ij} belongs follows a normal distribution $r_{ij} \sim N\left(u_i^T v_j, c_{ij}^{-1}\right)$, and it aims to optimize model with the goal $\mathbb{E}\left[r_{ij} | u_i, \theta_j, \epsilon_j\right] = u_i^T\left(\epsilon_j + \theta_j\right)$. Thus, CTM is able to learn latent factors for users and both latent factors and topics for items together using interaction data and contextual data.

2.5. Graph Convolution Network

The Graph Convolution Network (GCN) aims to model graph-structured data and adopt the constructed features for various machine learning tasks. Many real-world applications involve data that is naturally graph-structured, and the structural information can be encoded to model the relationships among entities [44]. For an undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \boldsymbol{A}\}$ where $\mathcal{V}$ denotes a set of nodes, $\mathcal{E}$ is a set of edges, and $\boldsymbol{A}$ is an adjacency matrix. In the adjacency matrix, $\boldsymbol{A}(i,j) = 0$ if there is no connection between nodes *i* and *j*; otherwise, $\boldsymbol{A}(i,j)$ can be set a scalar value representing the degree of the relationship between nodes *i* and *j*. Alternatively, one can set $\boldsymbol{A}(i,j) = 1$ for all *i*, *j* in the case of an unweighted graph. The work in [43] proposed a multi-layer GCN for document classification, the model can be defined as Eq. (5):

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right) \tag{5}$$

where $\tilde{A} = I + A$ represents the adjacency matrix combined with the identity matrix, allowing the self-connection of each node. Thus, the information from each node itself can be considered alongside the information from its neighbors during model training. $W^{(l)}$ represents the trainable parameters in the *l*-

th layer, $H^{(l)}$ is the feature representation for all nodes as the input of l-th layer, and $H^{(0)}$ represents the initial features. $\widetilde{D}^{-\frac{1}{2}}$ is an inversed degree matrix with normalization, the average features of all nodes are obtained in the form $\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}H^{(l)}$. Finally, the feature representations $H^{(l+1)}$ for all nodes in layer l+1 are obtained after applying an activation function σ.

Graph neural networks have been widely adopted in the study of recommendation systems, as data in recommenders inherently has a graph structure, including user-item interactions, user behavior sequences, social relationships, and knowledge entities [41]. Recently, UltraGCN has been proposed to construct an enhanced collaborative filtering model based on GCN and various types of relationships, such as item-item co-occurrence graphs [25]. On the other hand, the DHGCN model [22] has been developed to establish not only user-item connections but also user-user and item-item connections using GCN. Furthermore, this model incorporates a gating mechanism to integrate the content information of the top TF-IDF words into the feature representation of item nodes.

3. PROPOSED MODEL

3.1. Overview

We propose an attentive hybrid filtering network (AHFN) that constructs item features by considering four types of sources: (1) word-level characteristics generated using Word2Vec and CNN models based on the content of items, (2) latent factors for users and items learned from interaction data through a general matrix factorization (GMF) model, (3) sentence-level representations of items constructed using the SBERT model, and (4) item-item co-view relationships encoded in an undirected graph, with item features generated using a GCN model. Furthermore, attentive user features are constructed based on the user's most recently interacted items using an attention mechanism. The item features and attentive user features are then fed into a prediction network to estimate user preferences for unobserved items. Finally, a top-N recommendation list is provided to users. The overall architecture of our proposed attentive hybrid filtering network is illustrated in Fig 1.

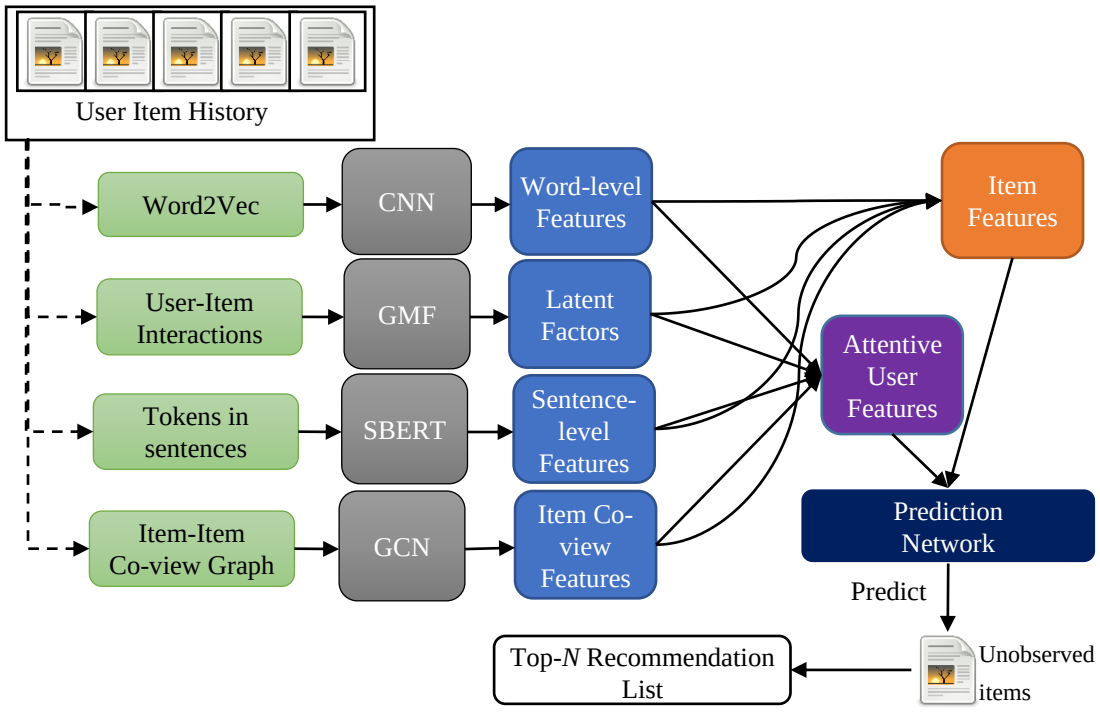


Fig 1. Overview of the Attentive Hybrid Filtering Network

3.2. Item Features and Attentive User Feature Construction

3.2.1. Word-level Features

The attentive contextual user feature considers the target item and users' recent actions, constructing user features based on the items' contextual information. Both the user features and item features are then fed into the prediction network for the subsequent recommendation task. The basic item information is represented by word sequences, with each word converted into a high-dimensional vector space. A CNN model is employed to extract contextual features from both the users' recent action items and the unobserved target item. Finally, the user's contextual features are constructed by applying an attention mechanism to the target item and the users' recent action items.

For the attentive contextual user feature $\mathcal{F}_u^{AC}$ for user u, the user's most recently interacted $|I^u|$ items are taken as input. Each item i in the set I^u can be represented as a matrix $D_i \in \mathbb{R}^{d \times L}$ consists of L words. Each word in D_i, denoted as $w_{(i,l)} \in \mathbb{R}^d$ corresponding the l-th word and a d-dimensional word embedding constructed using Word2Vec model. A contextual embedding for each item i defined as $f_i^C \in \mathbb{R}^{(nF \times nKS)}$, is constructed by applying a CNN model as follows:

$$c_i^f = \sigma(W^f * D_{i,(:,l:(l+ks-1))} + b^f) \tag{6}$$

$$e_i^{ks} = [\max(c_i^1), \max(c_i^2), \max(c_i^3), \dots, \max(c_i^{nF})] \tag{7}$$

$$f_i^C = [e_i^1, e_i^2, e_i^3, \dots, e_i^{nKS}] \tag{8}$$

where $W^f \in \mathbb{R}^{d \times ks}$ is a shared weight matrix for the CNN model's filter f, with filter size set to ks, and $b^f \in \mathbb{R}$ is the bias for f. The symbol $*$ denotes the convolution operator, and σ represents a non-linear activation function. Here the rectified linear unit (ReLU) is adopted over other functions such as sigmoid and tanh to avoid the gradient vanishing problem. The contextual feature vector $c_i^f \in \mathbb{R}^{l-ks+1}$ is constructed by operating the convolution with filter f and a kernel size ks along with the word sequence in the item i, as defined as Eq. (6). The number of filters nF is pre-defined for the CNN model, allowing the extraction of diverse structural features hidden in the items' word sequences. As a result, a total of nF contextual feature vectors are generated. We then apply max-pooling to obtain maximum value from each contextual feature vector, constructing a kernel-size-dependent maximum contextual feature vector $e_i^{ks} \in \mathbb{R}^{nF}$, where $ks = 1,2,\dots,nKS$, and nKS is the total number of kernel size settings, as defined in Eq. (7). There are different kernel sizes allow the construction of contextual features with varying word sequence lengths. Finally, all nKS contextual feature vectors are concatenated to form the representative contextual embedding for item I denoted as f_i^C, as defined in Eq. (8).

To considering user's recent actions on items while constructing the user feature, an attention mechanism is applied. Let the matrix $F_u^C \in \mathbb{R}^{(nF \times nKS) \times |I^u|}$ represents the contextual embeddings of items that user u has recently interacted with, contained in the set I^u. The contextual embedding of the target item t is denoted as f_t^C. The target item refers to the item that is unobserved by user u. Thus, we can obtain a weight vector that represents the importance between the target item t and user u's recent items by Eq. (9):

$$a^{C,u,t} = softmax(F_u^{C\,T} f_t^C) \tag{9}$$

where $F_u^{C\,T}$ is the transpose of the recent item embedding matrix for user u, and softmax function produce a weight vector $a^{C,u,t} \in \mathbb{R}^{|I^u|}$, with elements that summing to 1, the vector captures the correlations between the recent items of user u and the target item t. An attentive contextual user feature $\mathcal{F}_u^{AC} \in \mathbb{R}^{(nF \times nKS)}$ can then be obtained by applying the weight vector to user u's recent item embedding matrix, as defined in Eq. (10).

$$\mathcal{F}_u^{AC} = F_u^C a^{C,u,t} \tag{10}$$

3.2.2. Latent Factors

The latent factor model, such as matrix factorization, aims to extract hidden features or factors from the interactions between users and items. Traditionally, the user's latent factor vector is directly constructed during the iterations of gradient descent. In this work, we propose applying an attention mechanism to account for the relationship between the target item and the user's most recent action items in constructing attentive user latent factors.

First, each user's item interactions are taken as model input. The matrix factorization model serves as a latent factor learner, learning the users' latent factor matrix $P \in \mathbb{R}^{|U| \times d^{factor}}$ and the items' latent factor matrix $Q = \mathbb{R}^{d^{factor} \times |I|}$. The learner aims to find a solution for predicting the user-item interaction matrix $R \in \mathbb{R}^{|U| \times |I|}$, where $R = PQ$. Each element $r_{u,i} = p_u q_i$ in matrix *R* represents the interaction of user uuu with item iii. A value of 1 indicates that user *u* has interacted with or liked item *i,* while a value of 0 indicates no interaction or dislike. Thus, we can obtain a d^{factor}-dimensional latent factor vector $p_u \in \mathbb{R}^{d^{factor}}$ and $q_i \in \mathbb{R}^{d^{factor}}$ for user *u* and item *i*, respectively.

Beyond the user latent factors, the items' latent factors from user's recently interacted item set I^u are also considered to construct the user's attentive latent factor features $\mathcal{F}_u^{AMF} \in \mathbb{R}^{d^{factor}}$, defined as:

$$\mathcal{F}_u^{AMF} = Q_{I^u} a^{MF,u,t} \tag{11}$$

$$a^{MF,u,t} = softmax(Q_{I^u}^T q_t) \tag{12}$$

where Q_{I^u} is the item latent factor matrix representing user *u*'s recently interacted items I^u, and q_t is the item latent factors of the target item *t*. The softmax function applied to the product of Q_{I^u} and q_t results in a weight vector $a^{MF,u,t} \in \mathbb{R}^{|I^u|}$. This vector represents the importance between target item and user's recent items, with its elements lying in the interval [0, 1] and summing to 1. The model aims to construct attentive user latent factor features that align more closely with target item, predicting a higher recommendation score if the user's recent items have strong correlations with the target item. Conversely, the model will assign a low prediction score to the target item if the user has no relevant recent interactions.

At the end, the original user factor vector p_u learned directly by the matrix factorization model, is concatenated with the attentive user factor feature vector $\mathcal{F}_u^{AMF}$ to form the final user latent factor representation. Along with the user latent factors, the target item *t*'s latent factor q_t is also feed into the prediction network for the subsequent recommendation task.

3.2.3. Sentence-level Features

In this study, we applied the Sentence-Bidirectional Encoder Representations from Transformers (SBERT) model to construct sentence-level item embeddings to obtain rich contextual information for recommendation. Compared to conventional word embeddings, the item embeddings extracted by SBERT consider the relationships between tokens in the word sequences of items in both forward and backward directions. The features hidden in the sentences of items, which are semantically meaningful, can be extracted. Based on these item features, a more comprehensive attentive user feature set can then be constructed.

For constructing the contextual embedding of items, we consider the tokens of item sentences rather than individual words. Let N^S be the number of tokens considered for constructing item *i*'s SBERT embedding $f_i^B \in \mathbb{R}^{d^{SBERT}}$, where d^{SBERT} is the dimension of a SBERT embedding. For each user *u*, we also take their recent item set I^u as the input, and each item in the set is used to derived an item embedding. Thus, we can obtain a matrix $F_u^B = \mathbb{R}^{d^{SBERT} \times |I^u|}$ that represents the SBERT embedding of the user's recent items. Next, we can construct user *u*'s attentive features based on the sentence-level embeddings of items by applying attention mechanism, as defined below:

$$\mathcal{F}_u^{AB} = F_u^B a^{B,u,t} \tag{13}$$

$$a^{B,u,t} = softmax(F_u^B f_t^B) \tag{14}$$

where $f_t^B \in \mathbb{R}^{d^{SBERT}}$ is the target item t's sentence-level BERT embedding, and $a^{B,u,t} \in \mathbb{R}^{|I^u|}$ is a weight vector derived using softmax function, representing the relevance importance between the target item and user's recent items. The vector $\mathcal{F}_u^{AB} \in \mathbb{R}^{d^{SBERT}}$ represents the user's attentive features constructed using sentence-level BERT embedding. Finally, both the user features $\mathcal{F}_u^{AB}$ and item features f_t^B are feed into the prediction network to predict the recommendation score for user u and unobserved target item t.

3.2.4. Item Co-view Features

In conventional recommendation methods, such as association rules, items are recommended to users not based on personal features extracted or learned from the user, but rather on item-to-item rules. For example, if a user views item A, they are likely to view item B next. Such rules can assist recommendation systems and are constructed by analyzing user interactions with items. These rules can also be converted into an item-to-item co-view undirected graph, as illustrated in Fig 2. In this graph, each node represents an item, and each edge indicates the relationship between two nodes. This relationship can be quantified using the correlation or degree of association between the items.

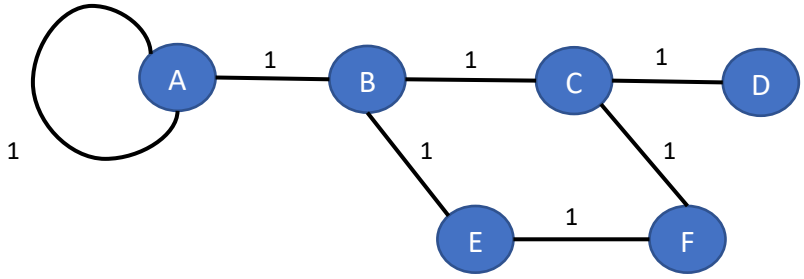


Fig 2. Item-to-item co-view graph.

In this study, we assign a value of 1 to an edge if two items were viewed by the same user on the same day, and a value of 0 otherwise. Thus, the graph represents the item co-view relationship. Furthermore, we can construct an item co-view adjacency matrix for the undirected graph, where an element value of 1 indicates a co-view relationship between two items, and 0 otherwise. The diagonal elements of the matrix are set to 1, representing the self-connection of each node. This allows the model to consider not only the information from an item's neighbors but also the information from the item itself when performing graph convolution.

Once the item co-view adjacency matrix is constructed, we can combine it with the base item features from the Word2Vec (W2V) or BERT models and apply a Graph Convolutional Network (GCN) to generate item co-view embeddings. These embeddings represent more comprehensive features for each item. Finally, users' attentive features can be constructed based on these item co-view embeddings using an attention mechanism. The GCN takes the item co-view adjacent matrix and item features as inputs, and the item co-view embedding matrix $\mathcal{F}^G \in \mathbb{R}^{|I|\times d}$ can be constructed using Eq. (15):

$$\mathcal{F}^G = \sigma(A^{CV} \dots \sigma(A^{CV}\sigma(A^{CV}H^0W^0 + b^0)W^1 + b^1) \dots W^L + b^L) \tag{15}$$

where $A^{CV} \in \mathbb{R}^{|I|\times|I|}$ is the item co-view adjacent matrix with element values of 1 or 0. $H^0 \in \mathbb{R}^{|I|\times d}$ is the initial d-dimensional feature matrix for all items, where item features can either be constructed using the simple average of word embeddings, or by applying the BERT model to the item's sentences. In layer-0 of the GCN, the matrix $W^0 \in \mathbb{R}^{d\times d}$ is the trainable parameters, $b^0 \in \mathbb{R}^{|I|}$ is the bias, and σ is the ReLU non-linear activation function. Each layer l produces an item embedding matrix H^{l+1}, which serves as the input for next layer $l+1$. We can stack up to L layers in the GCN to generate comprehensive co-view item embeddings. Notably, as the number of layers increases, the final node or item embedding $\mathcal{F}^G$ capture more global information from its neighborhood.

Furthermore, user u's recent item feature matrix $F_u^G = \mathbb{R}^{d^{GCN} \times |I^u|}$ can be retrieved from $\mathcal{F}^G$. The feature matrix F_u^G represents user u's recent preferences on items, and the feature dimension d^{GCN} depended on chosen base item feature model of GCN. Next, we construct user u's attentive features based on the co-view relationship embeddings by applying an attention mechanism to the user's recent items and the target item, as defined below:

$$\mathcal{F}_u^{AG} = F_u^G a^{G,u,t} \tag{16}$$

$$a^{G,u,t} = softmax(F_u^G f_t^G) \tag{17}$$

where $f_t^G \in \mathbb{R}^{d^{GCN}}$ is the target item t's features extracted from the co-view item embedding matrix $\mathcal{F}^G$. The weight vector $a^{G,u,t} \in \mathbb{R}^{|I^u|}$ is derived by using softmax function, representing the relevance between the target item and the user's recent items. The vector $\mathcal{F}_u^{AG} \in \mathbb{R}^{d^{GCN}}$ denotes the user's attentive features constructed using the embedded co-view item relationships. Finally, both the user features $\mathcal{F}_u^{AG}$ and the item features f_t^G are fed into the prediction network to predict the recommendation score for user u and the unobserved target item t.

3.3. Prediction Network

The prediction network predicts the preference of user u for the unobserved target item t based primarily on the attentive user features and the target item features, which are constructed from four core components: (1) Item contextual features based on word-level embeddings learned using the Word2Vec and CNN models; (2) Latent factors for both users and items, learned through general matrix factorization (GMF); (3) Item contextual features based on sentence-level embeddings learned using the SBERT model; (4) Item co-view relationships encoded by a undirected graph with the GCN model.

Based on the four base components, to construct comprehensive user features as input to the enhanced prediction network when making predictions for user u on item t, the following four base user features are considered: (1) attentive contextual user feature $\mathcal{F}_u^{AC}$, derived from the word-level representation of the item using CNN model; (2) the original user factors vector p_u, learned directly using matrix factorization model, and the attentive user factor feature vector $\mathcal{F}_u^{AMF}$; (3) $\mathcal{F}_u^{AB}$, representing the user's sentence-level attentive features, constructed using sentence BERT model; (4) $\mathcal{F}_u^{AG}$, representing the user's attentive features, constructed using embedded co-view item relationship and GCN model. These four base user features are then concatenated into a more comprehensive user feature, denoted as F_u^{CP} and defined in Eq.(18). Similarly, the comprehensive item feature ℓ_i^{CP} for item t is constructed by concatenating item features from four base components, as defined in Eq. (19).

$$\mathrm{F}_u^{CP} = [p_u, \mathcal{F}_u^{AC}, \mathcal{F}_u^{AMF}, \mathcal{F}_u^{AB}, \mathcal{F}_u^{AG}] \tag{18}$$

$$\ell_t^{CP} = [q_t, f_t^C, f_t^B, f_t^G] \tag{19}$$

$$\hat{r}_{u,t} = AHFN(\mathrm{F}_u^{CP}, \ell_i^{CP}) \tag{20}$$

Then, all the comprehensive user and item features are concatenated and feed into a multi-layered fully connected (FC) network with ReLU activation. In the network, each FC layer is followed by a layer normalization (LN) layer to reduce the internal covariate shift problem during model training. The final FC layer produce a scalar that indicated the degree of the preference for user u on item t. A sigmoid function then transfers the scalar into the range [0, 1], denote as $\hat{r}_{u,t}$, as shown in Eq. (20), where the function named AHFN represents our attentive hybrid filtering network. The objective function is defined as the binary cross-entropy loss, as described in Eq. (21).

$$L = -\sum_{(u,t)\in S} log(\hat{r}_{u,t}) - \sum_{(u,t)\in S^-} log(1 - \hat{r}_{u,t}) \tag{21}$$

where S represents the set of positive samples in the training data, where each user u and target item t

pair indicated that user liked the item. On the other hand, S^- is the set of negative samples in the training data, where user-item pair are chosen randomly. We assume that such randomly chosen items are irrelevant to user u. To minimize the loss function, the model needs to predict as 1 if the training sample is from the positive sample set, and 0 if the sample is from the negative sample set. The loss function can be minimized using the stochastic gradient decent method.

4. EXPERIMENT AND EVALUATION

4.1. Dataset and Evaluation Protocol

We used a dataset collected from a popular online news website in Taiwan called NiusNews (https://www.niusnews.com/), as described in Table 1. The dataset was split into a training set covering the period from 2018-10-01 to 2019-02-28, which was used for the model to learn the data structure. The next consecutive 3 days were used as the validation set for tuning suitable hyperparameters, and the 7 days following the validation set were used as the test set to evaluate the recommendation performance on future data. We adopted this rolling cross-validation approach because it aligns with real-world recommender systems, which train models on historical data collected up to the present and predict users' preferences for the following day.

Table 1. NiusNews Dataset Description

Dataset	**NiusNews**	Avg. Items	Std. Items
Time Period	2018-10-01 ~ 2019-03-10		
#items	4,723	26.54	68.84
#users	6,451	Avg. Records	Std. Records
#records	203,632	31.56	99.84

To demonstrate the performance of the proposed models, we adopted the leave-one-out methodology for top-*N* recommendation tasks, which is widely used in the literatures [4, 18]. In the experiments, for both the validation and test sets, we collected user-item pair where the item genuinely observed or read by the user to represent the user's true preference. Additionally, we randomly selected 1000 items as candidates for each true user-item pair, assuming that these 1000 items are irrelevant to the target user. Thus, each 1001-item recommendation task constitutes one test case. For each test case, the recommendation models must rank the user's preferred item in the top-*N* positions to register a hit.

One performance metric adopted in this study is recall, defined in Eq. (22), where T is the test case set and #hits indicate the number of test cases in which the user's preferred item is ranked in the top-N position among all candidate items. The other metric we used is normalized discounted cumulative gain (nDCG), defined in Eq. (23), where DCG (discounted cumulative gain) is defined in Eq. (24) and IDCG represents the ideal discounted cumulative gain. In DCG, rel_i is 0 or 1, indicating whether the item ranked at position i is the user's preferred item. The model achieves a high DCG score if it ranks the user's preferred item at the top of the candidate items. The main difference between recall and nDCG is that nDCG considers the ranking position, while recall does not. A good recommendation system or model should rank the user's preferred items at the top of the candidate list.

$$recall@N = \frac{\#hits}{|T|} \tag{22}$$

$$nDCG@N = \frac{DCG@N}{IDCG@N} \tag{23}$$

$$DCG@N = \sum_{i=1}^{N} \frac{2^{rel_i} - 1}{log_2(i+1)} \quad (24)$$

We adopted several baseline models to compare recommendation performance with our proposed model on the collected dataset in the experiments. The baselines are described in Table 2, where each row lists the model's name used in the experiments along with a description of the model.

Table 2. Baseline description.

Model Name	Description
Random	It is a very simple baseline that ranks all candidates for a user using uniform random guess.
Top-Popular	A commonly used recommendation method in the real-world ranks candidates for a user using the popularity of the items. For example, the item that has been read by the most users will be recommended first.
GMF	A matrix factorization model that constructs user and item features or latent factors using a neural network architecture [14]. This model is trained using only user and item interaction data.
MLP	A model that constructs user and item features or latent factors using a multi-layer neural network. It has the capability to capture non-linear behaviors among user and item interactions [14]. This model is trained using only user and item interaction data.
NCF	A model that integrates the capabilities of the GMF and MLP models to construct user and item features or latent factors for making recommendations [14].
LDA	The item features are constructed using the Latent Dirichlet Allocation (LDA) [8] model based on the content of the items. The user features are then constructed as the average of the item features interacted with by the user. This model is primarily trained on the contextual information of the items.
CTM	The model trains user and item features or latent factors, which are combined with the items' latent topics from the LDA model [39]. Therefore, this model is trained based on the contextual information of the items and the interaction logs between users and items.
CNNA	The model is inspired by [40] and constructs item features using a CNN model along with the contextual information of Word2Vec embeddings for the items. User features are then constructed based on the item features from the user's recent behavior. An attention mechanism is applied to the user's history and the target item to obtain summarized user features for making recommendation predictions.

In addition, in the experiments, we denote three alternatives of the proposed AHFN as follows: (1) CGMF-A, which integrates word-level contextual features and latent factors; (2) CGMF-BA, which incorporates sentence-level contextual features into CGMF-A; and (3) CGMF-BGA, which incorporates item-item co-view features into CGMF-BA. For all models, the parameter learning rate is searched within the range {0.1, 0.01, 0.001, 0.0001, 0.00001}, and the L2 normalization weight is also searched in the same range {0.1, 0.01, 0.001, 0.0001, 0.00001}. The batch size is searched in the range {32, 64, 128, 256}. We adopted the Adam optimizer to train the models for a maximum of 100 epochs, and the training process is stopped if the nDCG@10 on the validation set does not increase for 10 successive epochs.

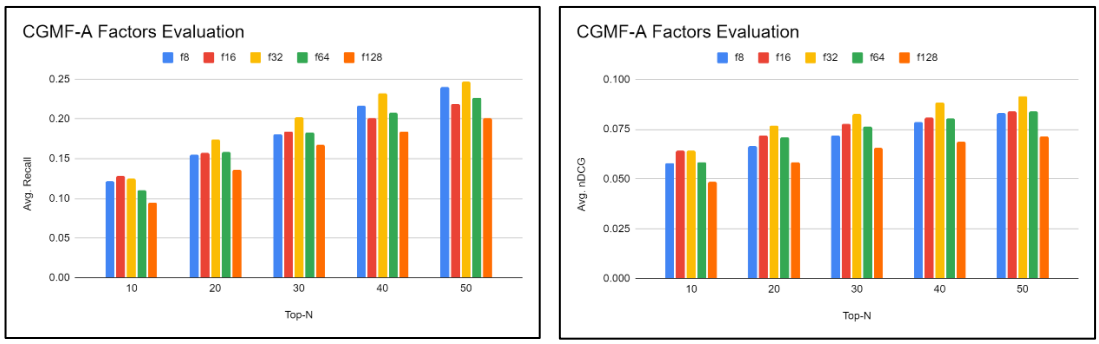
4.2. Model Evaluation

4.2.1. Evaluation of Integration of Word-level Features and Latent Factors

The CGMF-A model is based on the integration of CNN and GMF, which makes predictions using

word-level contextual features and latent factors. Furthermore, it employs the attention mechanism to construct attentive user features from latent factors. We set the hyperparameters of the CNN as follows: two kernel sizes of 1 and 2, each with 128 filters, and a maximum of 30 words with embeddings of 100 dimensions will be input to the model for each item. Since a larger number of kernels, filters, and words may lead to better model performance but also incur higher computational costs, the CNN's hyperparameter settings are fixed.

The hyperparameter we need to determine is the number of latent factors for CGMF-A that suit our dataset, using a 3-day validation set. We evaluate latent factors ranging from 8 to 128 in the top-*N* recommendation task, with *N* ranging from 10 to 50. The recommendation performance, in terms of average recall and average nDCG for CGMF-A, is reported in Fig 3. Experiment results show that a setting of 32 factors is effective for CGMF-A, yielding good performance in both recall and nDCG metrics across almost all recommendation tasks. Lower or higher factor sizes may negatively impact performance due to underfitting or overfitting of the model.

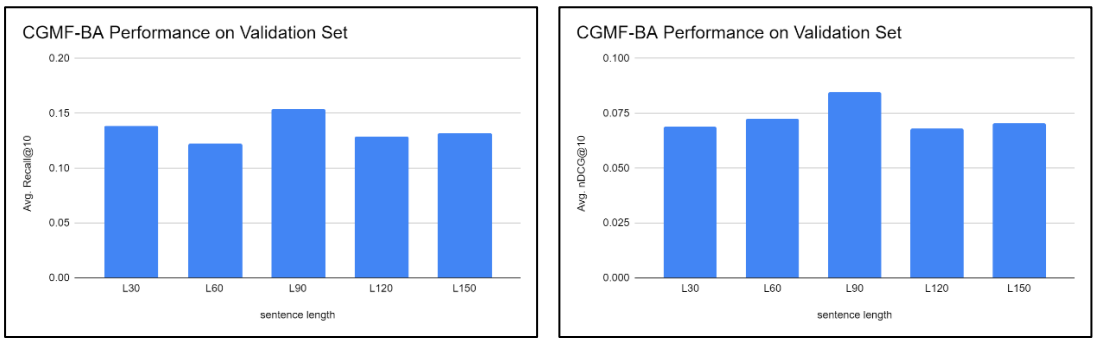


(a) (b)

Fig 3. Factors evaluation for CGMF-A models using validation set and recall (a), nDCG (b).

4.2.2. Evaluation of Integration of Sentence-level Features

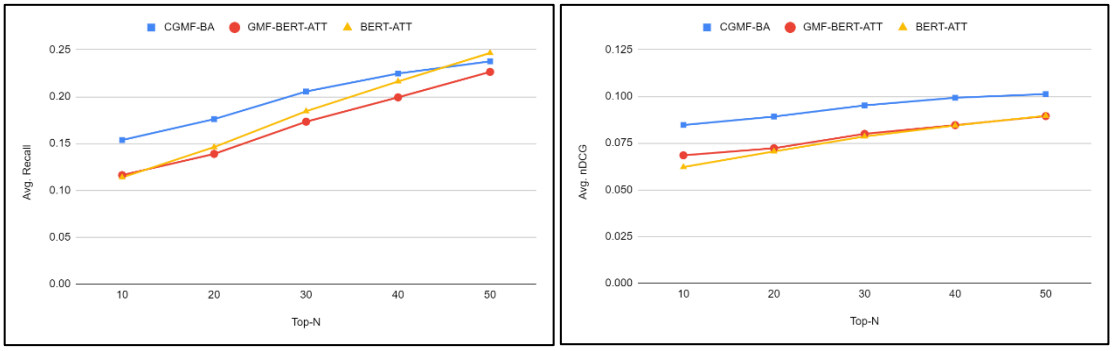
In the CGMF-BA model, we leverage the Sentence BERT (SBERT) model and the tokens in the content of an item to construct the sentence-level contextual features for that item. Thus, determining the number of tokens in a sentence or the maximum sentence length considered in feature construction is necessary. Here, the hyperparameters for CGMF-A are fixed, including number of latent factors, kernel size and filter size of CNN, and dimension of word-level features. We determine the sentence length using the validation set and the hardest top-10 recommendation task. The sentence length ranges from 30 to 150 tokens. The experimental results, reporting the average recall and nDCG metrics, are described in Fig 4. The results show that we can achieve the highest average recall and nDCG when using a maximum sentence length of 90 tokens.



(A) (b)

Fig 4. Average Recall (a) and nDCG (b) at the top-10 task for sentence length in CGMF-BA.

Now we evaluate the performance of CGMF-BA against two alternatives in top-*N* recommendation tasks using the validation set. GMF-BERT-A is the variant where the CNN module is removed from CGMF-BA, representing the integration of latent factors and SBERT features. BERT-ATT is the variant where both the CNN and GMF modules are removed from CGMF-BA, predicting the user's preference based solely on SBERT features. The maximum sentence length for SBERT in all models is fixed at 90 tokens. The average recall and nDCG are reported in Fig 5.



(a) (b)

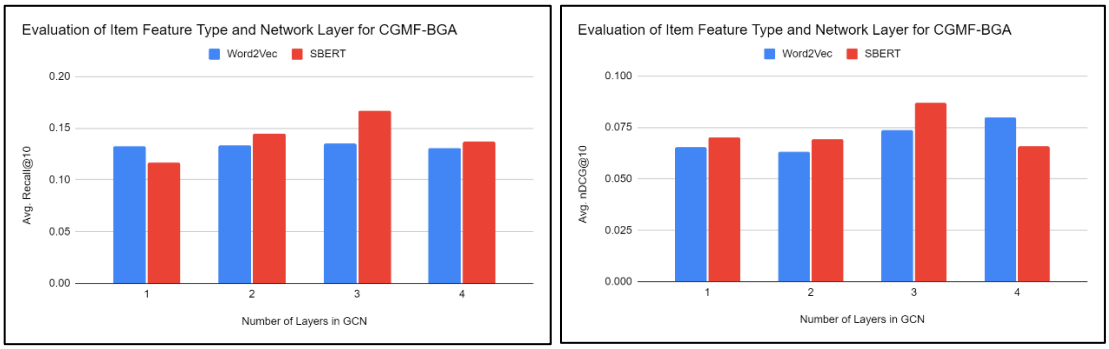
Fig 5. Average Recall(a) and nDCG(b) of CGMF-BA in top-n recommendation tasks.

The validation results show that the CGMF-BA model outperforms both alternatives in almost all cases except for the recall in the top-50 task. This indicates not only the effectiveness of integrating SBERT but also the benefits of constructing contextual features from both SBERT and CNN. This improvement may be attributed to the more comprehensive contextual item and user features learned from both word and token levels. On the other hand, there is a slight performance improvement when combining the latent factor and SBERT, which is better than the model that only considers SBERT, particularly in the harder top-10 recommendation task or a more ranking-related metric, nDCG. However, the improvement is not substantial compared to the model that takes all three components into consideration.

4.2.3. Evaluation of Integration of Item Co-view Features

The alternative CGMF-BGA of our proposed AHFN model that incorporates item-item co-view relationships to the CGMF-BA model using an undirected graph. For constructing such relations as the features for all items, the graph is modeled as an adjacency matrix, and together with the item features as the inputs of Graph Convolutional Network (GCN). Thus, there are two main hyperparameters that need to be determined for the GCN. The first is the method for constructing features for items or the item feature model, and the second is the number of layers for the GCN to learn the embeddings for items.

For item features, we can either use the simple average of 100-dimensional word embeddings from the Word2Vec model or leverage the 384-dimensional token embeddings from the sentence BERT (SBERT) model, as adopted in the CGMF-BA model. Regarding the number of layers in our GCN, we may choose from a range of 1 to 4. It is important to note that a higher number of layers allows the GCN to learn item embeddings by considering information from longer-distance neighborhoods. We conduct experiments to determine the optimal learning rate, number of epochs, and the two hyperparameters using the validation set. For the CGMF-BA model, there are also two kernels, each with 128 filters for the CNN, and a maximum of 30 words with 100-dimensional embeddings as input. The number of latent factors is fixed at 32, and the maximum sentence length for SBERT is set at 90, according to the experimental results. The evaluation results for the item feature model and layer size for the GCN are presented in Fig 6.

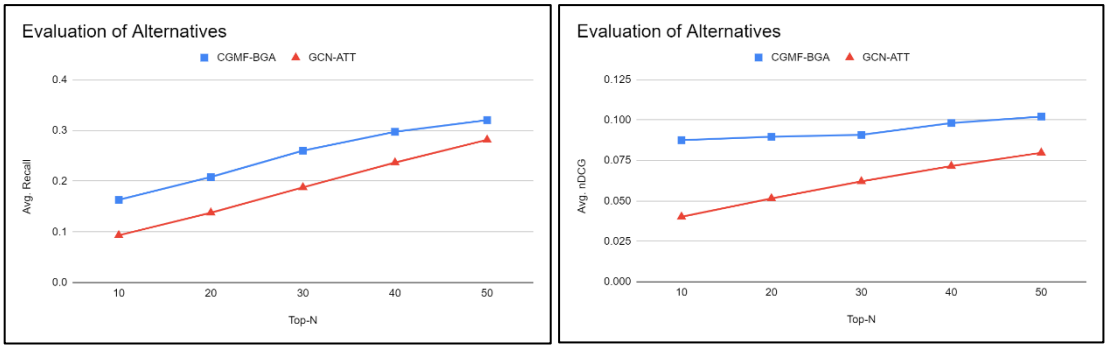


(a) (b)

Fig 6. The average recall (a) and nDCG (b) for item feature model and number of layers in GCN.

The results show that the 384-dimensional item features constructed from sentence BERT achieved the highest average recall and nDCG using the 3-layer GCN model. The experimental performance indicates that the GCN model benefits recommendation tasks by utilizing higher-dimensional item and user feature representations, as well as information from longer-distance item neighbors. In contrast, the 100-dimensional word embeddings performed poorly in many cases, likely due to their simplified representation of item characteristics, which utilizes lower dimensions and averages the item features.

We also evaluate the performance of CGMF-BGA against an alternative model that only employs the GCN and feature attention mechanism, denoted as GCN-ATT, on the validation set. The GCN-ATT also utilizes SBERT item features and a 3-layer GCN, with hyperparameters tuned to achieve optimal performance. The experimental results for top-*N* recommendation tasks are presented in Fig 7.



(a) (b)

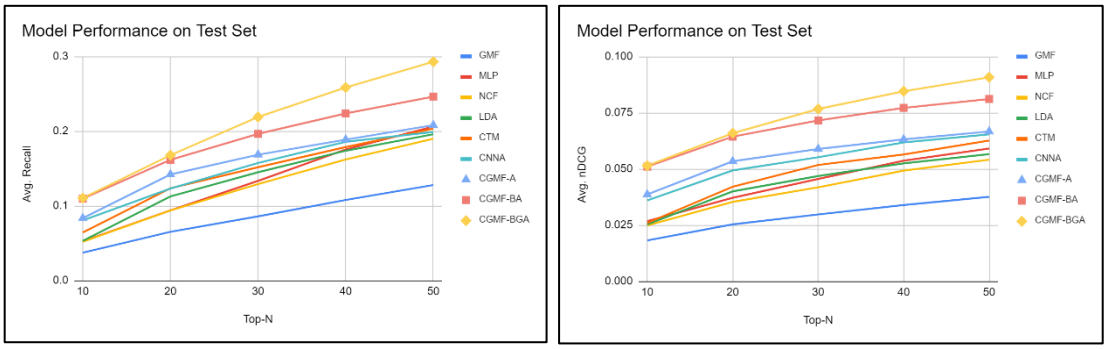
Fig 7. Average recall(a) and nDCG(b) comparison between CGMF-BGA and alternative.

The experimental results on the validation set indicated that CGMF-BGA outperformed the alternative in both average recall and nDCG metrics across all recommendation tasks. This reveals that the integration of latent factors, contextual features constructed using both CNN and SBERT, and item co-view relations embedded using the GCN model positively contributes to performance improvement. Additionally, the results suggest that the GCN model alone is not sufficient to achieve adequate performance in these recommendation tasks when compared to our proposed model.

4.3. Experiment Results

We now further compare our proposed models, including the alternatives CGMF-A, CGMF-BA, and CGMF-BGA, with several baselines on the test set in top-*N* recommendation tasks. The hyperparameters for all comparison models were searched and fixed using the validation set. The experimental results for average recall and nDCG are reported in Fig 8. Among our three proposed alternatives, CGMF-BGA

achieved the best performance, followed by CGMF-BA in second place and CGMF-A in third. The results also shows that our proposed model, CGMF-BGA, outperforms all baselines in both average recall and nDCG metrics for every recommendation task. The performance improvement increases as the N recommendation size increases.



(a) (b)

Fig 8. Average recall for CGMF-BGA and baselines on test set in top-N recommendation tasks.

Here we also summarized the experimental figures in Table 3 and Table 4 for average recall and nDCG metrics, respectively. We conducted a one-sample t-test and set the confidence level at 95%, thus, a p-value < 0.05 indicates that the proposed model CGMF-BGA has statistically significantly higher average recall or nDCG than the baseline. In such cases, we mark a "*" beside the figure of the baseline. Additionally, we use boldface figures to report the proposed model if it outperforms the baseline.

According to the summarized figures, our proposed model CGMF-BGA outperform all baselines, including our alternative CGMF-A and CGMF-BA models, in terms of average recall and nDCG across all recommendation tasks. Although there is no significant difference between CGMF-BGA and CGMF-BA. However, if we focus on nDCG metric and compared with CNNA model and other baselines in top-10 to top-30 recommendation tasks, the obtained improvements are statistically significant. For the top-40 and top-50 tasks, the CGMF-BGA model achieves statistically significantly higher average nDCG compared with the proposed CGMF-A. These results clearly reveal that the item's co-view relationship is an important factor for performance improvement. The co-view relationship, embedded using GCN and integrated with the features constructed using CNN, SBERT, and latent factors, can improve the recommendation system. This model integration is effective in helping users find their truly preferred items among the pool of candidates.

Table 3. Summary of average recall and significance for CGMF-BGA and the baselines.

Avg. Recall					
Model	**Top-10**	**Top-20**	**Top-30**	**Top-40**	**Top-50**
Random	0.00954 *	0.01788 *	0.03013 *	0.03778 *	0.04742 *
Top-Popular	0.00956 *	0.02192 *	0.03166 *	0.03605 *	0.04933 *
GMF	0.03707 *	0.06509 *	0.08575 *	0.10771 *	0.12772 *
MLP	0.05239 *	0.09384 *	0.13328 *	0.17542 *	0.20600 *
NCF	0.05173 *	0.09373 *	0.12894 *	0.16197 *	0.18978 *
LDA	0.05301 *	0.11240 *	0.14488 *	0.17364 *	0.19569 *
CTM	0.06425 *	0.12334 *	0.15157 *	0.17863 *	0.20319 *
CNNA	0.08042	0.12300 *	0.15730 *	0.18562 *	0.19881 *
CGMF-A	0.08336	0.14191	0.16836 *	0.18851 *	0.20788 *
CGMF-BA	0.10955	0.16150	0.19628	0.22354	0.24610
CGMF-BGA	**0.11027**	**0.16768**	**0.21880**	**0.25838**	**0.29284**

* indicates that the performance difference between CGMF-BGA and the baseline is statistically significant.

Table 4. Summary of average nDCG and significance for CGMF-BGA and baselines.

Avg. nDCG					
Model	**Top-10**	**Top-20**	**Top-30**	**Top-40**	**Top-50**
Random	0.00464 *	0.00670 *	0.00930 *	0.01078 *	0.01253 *
Top-Popular	0.00401 *	0.00732 *	0.00893 *	0.00973 *	0.01254 *
GMF	0.01815 *	0.02530 *	0.02971 *	0.03389 *	0.03753 *
MLP	0.02672 *	0.03714 *	0.04553 *	0.05367 *	0.05906 *
NCF	0.02483 *	0.03532 *	0.04180 *	0.04920 *	0.05409 *
LDA	0.02528 *	0.04000 *	0.04683 *	0.05246 *	0.05655 *
CTM	0.02593 *	0.04205 *	0.05175 *	0.05639 *	0.06261 *
CNNA	0.03598 *	0.04932 *	0.05518 *	0.06177 *	0.06536 *
CGMF-A	0.03860	0.05338	0.05888	0.06314 *	0.06664 *
CGMF-BA	0.05101	0.06435	0.071568	0.07715	0.08109
CGMF-BGA	**0.05140**	**0.06587**	**0.07667**	**0.08455**	**0.09075**

* indicates that the performance difference between CGMF-BGA and the baseline is statistically significant.

5. DISCUSSIONS AND CONCLUSIONS

In this work, we propose a novel attentive hybrid filtering network (AHFN) to model three types of data: user-item behavior, word and sentence-level contextual information, and item-item co-view relationships. Specifically, our proposed model can be classified into three types: (1) the integration of latent factors for users and items constructed using a latent factor model (LFM), along with word-level contextual item features using a convolutional neural network (CNN), denoted as CGMF-A; (2) CGMF-A combined with sentence-level item features constructed using sentence bidirectional encoder representations from transformers (SBERT), denoted as CGMF-BA; and (3) CGMF-BA integrated with the encoded item co-view relationships using an undirected graph structure and graph convolutional network (GCN), denoted as CGMF-BGA. The attentive user features in all three types are constructed based on item features related to the user's recent behavior. Finally, preferences for items are predicted for users using user-item features and a prediction network.

The experiments are conducted on data collected from an online news website in Taiwan, and performance is compared with several baselines for top-n recommendation tasks. The experimental results show that CGMF-A outperforms collaborative filtering models that are constructed using only interaction data. However, such improvement is not strong enough when compared to NCF, which has the capability to model both linear and non-linear user behavior, or to CTM and CNNA, which model user behavior and contextual information together. The results demonstrate the effectiveness of CGMF-BA, which integrates word-level and sentence-level item features with latent factors. On the other hand, on the test set, the CGMF-BA model outperforms Random, Top-Popular, GMF, MLP, NCF, and LDA in terms of average Recall and nDCG in the top-10 to top-30 recommendation tasks, with improvements that are statistically significant.

For the CGMF-BGA model, which uses item features constructed with SBERT and a 3-layer GCN, the performance on the validation set outperforms models that make recommendations using GCN only. The results show the effectiveness of integrating latent factors, word-level and sentence-level contextual features, and encoded item-item co-view relationships. On the test dataset, CGMF-BGA outperforms all baselines in terms of average Recall and nDCG in the top-20 to top-50 recommendation tasks, and the experimental results are statistically significant. The findings also reveal the effectiveness of the proposed model, demonstrating that latent factors, word-level and sentence-level contextual features, and encoded item-item co-view relationships are important factors for improving the performance of news recommendation systems.

There are several suggestions for future study directions: (1) In this work, the proposed model was only evaluated on one collected news dataset. It would be beneficial to further confirm the performance of the proposed model on additional public datasets to explore the model's application areas. (2) Although several baselines were compared in the experiments, some modern models, such as UltraGCN and

DHGCN, were not considered. Understanding the performance differences between the proposed model and these modern models would be valuable. (3) In this work, we modeled the item-item co-view relationship using GCN. It would be an interesting topic to explore whether it is possible to model a knowledge graph that represents the relationships between items and to incorporate such relationships with GCN to enhance recommendation performance.

Acknowledgement

This research was supported by the National Science and Technology Council of Taiwan under grant number: NSTC112-2410-HA49-015-MY2.

REFERENCES

1. Adiyansjah, A. A. S. Gunawan, and D. Suhartono, "Music Recommender System Based on Genre using Convolutional Recurrent Neural Networks," *Procedia Computer Science,* Vol. 157, No. 2019 pp. 99-109.
2. S. M. Anwar, M. Majid, A. Qayyum, M. Awais, M. Alnowami, and M. K. Khan, "Medical Image Analysis using Convolutional Neural Networks: A Review," *Journal of Medical Systems,* Vol. 42, No. 11, 2018.p. 226.
3. D. V. Bagul and S. Barve, "A novel content-based recommendation approach based on LDA topic modeling for literature recommendation," *2021 6th International Conference on Inventive Computation Technologies (ICICT)*, 2021, pp. 954-961.
4. I. Bayer, X. He, B. Kanagal, and S. Rendle, "A Generic Coordinate Descent Framework for Learning from Implicit Feedback," presented at the *Proceedings of the 26th International Conference on World Wide Web*, Perth, Australia, 2017.
5. R. M. Bell and Y. Koren, "Lessons from the Netflix prize challenge," *SIGKDD Explor. Newsl.,* Vol. 9, No. 2, 2007 pp. 75–79.
6. A. Bellogin and J. Parapar, "Using graph partitioning techniques for neighbour selection in user-based collaborative filtering," presented at the *Proceedings of the sixth ACM conference on Recommender systems*, Dublin, Ireland, 2012.
7. S. Bergamaschi and L. Po, Comparing LDA and LSA Topic Models for Content-Based Movie Recommendation Systems, 2015.
8. D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.,* Vol. 3, No. null, 2003 pp. 993–1022.
9. E. Çano and M. Morisio, "Hybrid recommender systems: A systematic literature review," *Intell. Data Anal.,* Vol. 21, No. 6, 2017 pp. 1487–1524.
10. K. Choi, D. Yoo, G. Kim, and Y. Suh, "A hybrid online-product recommendation system: Combining implicit rating-based collaborative filtering and sequential pattern analysis," *Electronic Commerce Research and Applications,* Vol. 11, No. 4, 2012 pp. 309-317.
11. J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *North American Chapter of the Association for Computational Linguistics*, 2019.
12. C. Feng, M. Khan, A. U. Rahman, and A. Ahmad, "News Recommendation Systems - Accomplishments, Challenges & Future Directions," *IEEE Access,* Vol. 8, No. 2020 pp. 16702-16725.
13. F. Fessahaye, L. Perez, T. Zhan, R. Zhang, C. Fossier, R. Markarian, C. Chiu, J. Zhan, L. Gewali, and P. Oh, "T-RECSYS: A Novel Music Recommendation System Using Deep Learning," *2019 IEEE International Conference on Consumer Electronics (ICCE)*, 2019, pp. 1-6.
14. X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural Collaborative Filtering," presented at the *Proceedings of the 26th International Conference on World Wide Web*, Perth, Australia, 2017.
15. Y. Hu, Y. Koren, and C. Volinsky, "Collaborative Filtering for Implicit Feedback Datasets," *2008 Eighth IEEE International Conference on Data Mining*, 2008, pp. 263-272.
16. Z. Huang, H. Chen, and D. Zeng, "Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering," *ACM Trans. Inf. Syst.,* Vol. 22, No. 1, 2004 pp. 116–142.

17. H. Ko, S. Lee, Y. Park, and A. Choi, "A Survey of Recommendation Systems: Recommendation Models, Techniques, and Application Fields," *Electronics,* Vol. 11, No. 1, 2022.p. 141.
18. Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," presented at the *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, Las Vegas, Nevada, USA, 2008.
19. Y. Koren, R. Bell, and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," *Computer,* Vol. 42, No. 8, 2009 pp. 30-37.
20. A. Kumar Sharma, B. Bajpai, R. Adhvaryu, S. Dhruvi Pankajkumar, P. Parthkumar Gordhanbhai, and A. Kumar, "An Efficient Approach of Product Recommendation System using NLP Technique," *Materials Today: Proceedings,* Vol. 80, No. 2023 pp. 3730-3743.
21. S. Lawrence, C. L. Giles, T. Ah Chung, and A. D. Back, "Face recognition: a convolutional neural-network approach," *IEEE Transactions on Neural Networks,* Vol. 8, No. 1, 1997 pp. 98-113.
22. T. Liang, L. Ma, W. Zhang, H. Xu, C. Xia, and Y. Yin, "Content-aware Recommendation via Dynamic Heterogeneous Graph Convolutional Network," *Knowledge-Based Systems,* Vol. 251, No. 2022.p. 109185.
23. P. Lops, M. de Gemmis, and G. Semeraro, "Content-based Recommender Systems: State of the Art and Trends," in Recommender Systems Handbook, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds., ed Boston, MA: Springer US, 2011, pp. 73-105.
24. J. Lu, D. Wu, M. Mao, W. Wang, and G. Zhang, "Recommender system application developments: A survey," *Decision Support Systems,* Vol. 74, No. 2015 pp. 12-32.
25. K. Mao, J. Zhu, X. Xiao, B. Lu, Z. Wang, and X. He, "UltraGCN: Ultra Simplification of Graph Convolutional Networks for Recommendation," presented at the *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, Virtual Event, Queensland, Australia, 2021.
26. R. Meteren, "Using Content-Based Filtering for Recommendation," 2000.
27. T. Mikolov, K. Chen, G. S. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," *International Conference on Learning Representations*, 2013.
28. A. Mongia, N. Jhamb, E. Chouzenoux, and A. Majumdar, "Deep latent factor model for collaborative filtering," *Signal Processing,* Vol. 169, No. 2020.p. 107366.
29. R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang, "One-Class Collaborative Filtering," *2008 Eighth IEEE International Conference on Data Mining*, 2008, pp. 502-511.
30. N. Pavitha, V. Pungliya, A. Raut, R. Bhonsle, A. Purohit, A. Patel, and R. Shashidhar, "Movie recommendation and sentiment analysis using machine learning," *Global Transitions Proceedings,* Vol. 3, No. 1, 2022 pp. 279-284.
31. L. T. Ponnam, S. D. Punyasamudram, S. N. Nallagulla, and S. Yellamati, "Movie recommender system using item based collaborative filtering technique," *2016 International Conference on Emerging Trends in Engineering, Technology and Science (ICETETS)*, 2016, pp. 1-5.
32. S. Rajarajeswari, S. Naik, S. Srikant, M. K. Sai Prakash, and P. Uday, "Movie Recommendation System," Singapore %@ 978-981-13-5953-8, 2019, pp. 329-340.
33. S. Raza and C. Ding, "News recommender system: a review of recent progress, challenges, and opportunities," *Artificial Intelligence Review,* Vol. 55, No. 1, 2022 pp. 749-800.
34. N. Reimers, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," *arXiv preprint arXiv:1908.10084,* 2019.
35. P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens: an open architecture for collaborative filtering of netnews," presented at the *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, Chapel Hill, North Carolina, USA, 1994.
36. B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," presented at the *Proceedings of the 10th international conference on World Wide Web*, Hong Kong, Hong Kong, 2001.
37. Y. Shi, M. Larson, and A. Hanjalic, "Collaborative Filtering beyond the User-Item Matrix: A Survey of the State of the Art and Future Challenges," *ACM Comput. Surv.,* Vol. 47, No. 1, 2014.p. Article 3.
38. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," presented at the *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Long Beach, California, USA, 2017.
39. C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles,"

presented at the *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, San Diego, California, USA, 2011.
40. H. Wang, F. Zhang, X. Xie, and M. Guo, "DKN: Deep Knowledge-Aware Network for News Recommendation," presented at the *Proceedings of the 2018 World Wide Web Conference*, Lyon, France, 2018.
41. S. Wu, F. Sun, W. Zhang, X. Xie, and B. Cui, "Graph Neural Networks in Recommender Systems: A Survey," *ACM Comput. Surv.,* Vol. 55, No. 5, 2022.p. Article 97.
42. C. C. Yang, H. Chen, and K. Hong, "Visualization of large category map for Internet browsing," *Decision Support Systems,* Vol. 35, No. 1, 2003 pp. 89-102.
43. L. Yao, C. Mao, and Y. Luo, "Graph convolutional networks for text classification," presented at the *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, Honolulu, Hawaii, USA, 2019.
44. S. Zhang, H. Tong, J. Xu, and R. Maciejewski, "Graph convolutional networks: a comprehensive review," *Computational Social Networks,* Vol. 6, No. 1, 2019.p. 11.

Yun-Cheng Chou received the M.S. degree in Information Management from National Chiao Tung University, Taiwan, in 2012. He is currently pursuing the Ph.D. degree with the Institute of Information Management, National Yang Ming Chiao Tung University, Taiwan. His research interests include data mining, machine learning, and recommender systems.

Duen-Ren Liu is a professor of the Institute of Information Management at the National Yang Ming Chiao Tung University of Taiwan. He received the B.S. and M.S. degrees in Computer Science from the National Taiwan University, Taiwan, in 1985 and 1987, respectively. He received the PhD degree in Computer Science from the University of Minnesota, USA, in 1995. His research interests include data mining, knowledge engineering, e-commerce and recommender systems.