HyCoNN: Hybrid Cooperative Neural Networks for Personalized News Discussion Recommendation

Julian Risch, Victor Künstler, Ralf Krestel
Hasso Plattner Institute, University of Potsdam
Potsdam, Germany
julian.risch@hpi.de, victor.kuenstler@student.hpi.de, ralf.krestel@hpi.de

***Abstract*—Many online news platforms provide comment sections for reader discussions below articles. While users of these platforms often *read* comments, only a minority of them regularly *write* comments. To encourage and foster more frequent engagement, we study the task of personalized recommendation of reader discussions to users. We present a neural network model that jointly learns embeddings for users and comments encoding general properties. Based on explicit and implicit user feedback, we sample relevant and irrelevant reader discussions to build a representative training dataset. We compare to several baselines and state-of-the-art approaches in an evaluation on two datasets from *The Guardian* and *Daily Mail*. Experimental results show that the recommendations of our approach are superior in terms of precision and recall. Further, the learned user embeddings are of general applicability because they preserve the similarity of users who share interests in similar topics.**

***Index Terms*—recommender systems, social media, online discussions, document representation, neural networks, natural language processing, personalization**

I. Introduction

Comment sections on online news platforms allow readers to discuss article topics and communicate directly with each other. Many readers take advantage of this opportunity, and these reader discussions thereby enrich the content of the platforms. For instance, a study found out that 78% of U.S. Americans read comments on news, and 55% contribute them [1]. 19% of users who post a comment even spend more time with the comments than with the article.

In this paper, we aim to encourage engagement in online discussions by recommending specific articles' comments to individual platform users. These recommendations not only can lead to higher user loyalty, higher retention rates, and increased website traffic; they can also make the group of users who contribute to a discussion more diverse by encouraging users who would otherwise stay passive. To this end, we model users and their participation in online news discussions using comment texts and commenter co-occurrence. We propose HyCoNN (Hybrid Cooperative Neural Networks), which jointly learns representations of users and reader discussions. We conduct experiments on datasets of past user comments from the websites of Daily Mail and Guardian. For the offline evaluation of the recommendation performance, we use a ranking task: Given a specific time and user, we reconstruct the state of the reader discussions available at that time. We then rank these discussions by estimating whether the user is likely to contribute to a particular discussion or not.

Since commenting is a highly social activity, we build on the idea of homophily (the tendency of users to associate with people who appear similar). We leverage node2vec to learn user embeddings on a bipartite graph that connects platform users with the discussions they participated in. Consequently, pairs of users who often co-occur in reader discussions because of rivalry, friendship, or shared interests appear closer to each other in the embedding space.

In summary, we make the following contributions: First, we present HyCoNN (Hybrid Cooperative Neural Networks), a model that jointly learns representations of users and reader discussions. In contrast to previous work in the related domain of product reviews and rating prediction, such as DeepCoNN (Deep Cooperative Neural Networks) [2], we combine content-based and user-co-occurrence-based approaches. Second, we train our model to solve a discussion recommendation task on two real-world comment datasets by sampling appropriate positive and negative discussions. We evaluate eight different methods and find that our method HyCoNN achieves the best precision@k and recall@k for $k \in \{5, 10, 15\}$. Finally, we also evaluate the quality of the learned user embeddings by assessing whether they preserve similarities between users. The results show that HyCoNN preserves these similarities best, meaning that the embeddings are not specific to the recommendation task but could be reused for other tasks.

II. Related Work

Recommending news discussions is a novel task, which became relevant after online news comments became tremendously popular in recent years. In contrast to news article recommendation [3], [4], where the goal is typically to find interesting articles for the user to read, the focus of recommending discussions is to foster engagement by suggesting the comments of particular articles that users might want to contribute to. While the problem setting is similar, there are three major differences: (1) Users need to authenticate themselves on the news platform to author comments, which allows recommender systems to create user profiles [5]. (2) The data available to make informed recommendations is much richer. Besides the article itself, there are previous comments, along with their authors' information. (3) This leads to a more subtle difference, which is the reason *why* someone comments. While a recommendation for reading an article can mostly be based on the topical interest of the user, the reason for commenting

could be the article, other comments, or the fact that one or more particular users have commented. In our approach, we actually refrain from using the article text as a source of information due to the weak signal in comparison to the comments from other users. Further, topical interest as derived from the article itself is too coarse-grained and shifts over time [6].

While our focus is on recommending entire discussions, there is related work on recommending single comments as well. Both tasks have the common goal to foster user engagement. Comment recommendations are either personalized [7] or based on a community's preferences [8]–[10]. These recommended comments can be integrated into online platforms by adjusting the standard chronological ranking of comments by their estimated relevance to users. Further, a threaded (hierarchical) presentation of comments increases reciprocity compared to a linear presentation: users more often reply back when another user replies to their earlier comment [11].

In the field of recommending discussions, previous approaches typically combine collaborative filtering and content-based recommenders, thus exploiting the available data: co-commenting patterns and article content. In contrast to these approaches, we make use of the comment text instead of the article text for assessing relevance to a user. Most work along these lines employs topic modeling to model users and content. Bansal et al. [12] combine collaborative topic modeling [13] with matrix factorization [14] to identify comment-worthy articles. Because of the time-consuming Gibbs sampling and the constraints on vocabulary size, the model is tailored to data of smaller size and lower topical diversity. This limitation makes it suitable for specialized blogs with a few thousand users but renders the large-scale deployment for news platforms with more than one hundred thousand users infeasible. Another approach [15] combines collaborative filtering and topic modeling in a learning-to-rank setting. The approach of Shmueli et al. [16] combines memory-based collaborative filtering (CF) and latent factor models for both tag-based and co-commenting patterns but ignores comment content. Their evaluation reveals a challenge of static train-test data splits, also identified by Aharon et al. [17]: They do not take into account that the comments in a discussion are added gradually rather than all at once. Further, if users did not join a particular discussion it might just be that they were inactive during that time. Notably, it cannot be inferred that the discussion topic is irrelevant to them. In our study, we build on these findings and design a more realistic evaluation scenario. We model *when* each user was active and what comments were published at that time.

The neural network architecture that we propose in this study extends earlier work on review rating prediction by Zheng et al. [2]. They propose deep cooperative neural networks (DeepCoNN), which consist of two networks that are joined by a final shared layer. The first network models user behavior using the texts of a particular user's reviews, while the second network models item properties using the texts of all reviews on a particular item. The final layer learns the interaction of users and items to predict review ratings. Both networks are convolutional neural networks (CNNs) that get a sequence of words as input and provide latent features of that text as output. The concatenation of the resulting user embedding and item embedding is used as input to a factorization machine [18] to predict review ratings.

Seo et al. [19] also join two networks to learn user and item representations to predict review ratings. However, instead of putting word embeddings directly into a CNN, they introduce an attention layer to learn the importance of words locally and globally. The learned item embeddings are evaluated by using a linear support vector machine for multi-class classification of items into categories. For both approaches, the main difference to our work is that they are solely content-based and that they are tailored to the domain of product reviews. We adapt the architecture for the domain of online discussions and combine it with an additional neural network for collaborative filtering, which captures user co-occurrence patterns.

Several model-based collaborative filtering approaches apply matrix factorization [20] to discover latent factors and reduce the dimensionality [21], [22]. Inspired by deep learning techniques, they combine CNNs with matrix factorization [22] or factorize the user co-occurrence matrix in a similar way to the factorization of the word co-occurrence matrix, e.g., in word2vec or GloVe [21]. In comparison to memory-based techniques, model-based techniques manage the sparsity and scalability issues better. However, besides the expensive model-building, Koren et al. [20] still describe sparsity as a major problem of these methods.

Most deep learning methods generate embeddings in their learning process to model users and/or items (in our case reader discussions). To this end, it is reasonable to check whether these embeddings are meaningful representations of the real world entities. A common way [19], [23] to qualitatively evaluate user embeddings involves dimensionality reduction and visualization of the embeddings. This approach allows evaluating whether they form clusters of similar users. However, it does not quantitatively measure whether they are useful. Blandfort et al. [24] propose a novel measure, called Pair-Distance Correlation. It quantifies whether the learned embeddings reflect that similar users have similar embeddings. This measure was designed for the rating prediction task. We tailored it to our task and used it in our evaluation.

III. HyCoNN Recommender System

We introduce HyCoNN (Hybrid Cooperative Neural Networks), which combines a content-based and a user-co-occurrence-based recommendation approach. To this end, we first adapt the DeepCoNN model [2] to the task of discussion recommendation. Second, we describe how to use node2vec [25] on a graph of user co-occurrences in reader discussions. Finally, we describe how to combine both approaches in our proposed HyCoNN architecture, visualized in Figure 1. Our implementation is available on GitHub.[1]

[1] https://hpi.de/naumann/projects/repeatability/text-mining.html

A. Adaptation of DeepCoNN

We use a CNN architecture for text processing similar to the one proposed for the DeepCoNN architecture [2]. The input is a sequence of N words $w_0, \ldots, w_{N-1}$. The first layer translates every word to its corresponding pre-trained word embedding using a lookup table. For DeepCoNN and also HyCoNN, we use 300-dimensional fastText word embeddings, which were pre-trained on the English-language Common Crawl dataset [26]. The resulting embeddings function as input to a convolutional layer that consists of n neurons. Each neuron has an associated filter kernel $K \in \mathbb{R}^{d \times o}$, where d corresponds to the word embedding dimension and o to the window of words to which the kernel is applied. After the convolution, we apply a ReLU activation function to the output. From the resulting $N - o + 1$ features, we extract one feature by using a max-pooling operation. This pooling operation allows extracting features regardless of the input length. Finally, the concatenation of all n max-pooling outputs is fed into a fully connected layer of size l, with a ReLU activation function.

To predict how a user would rate an item, the DeepCoNN model uses two CNNs. We adapt this model to our problem in the following way: We use the users' past comments as input to the first CNN. The output of that CNN serves as a user representation. The second CNN receives as input the concatenated texts of all comments of an article. The output is the corresponding discussion representation. We apply a dropout layer to the outputs of both CNNs for regularization.

We also concatenate the user representation and the individual discussion representation and use it as input for a factorization machine (FM) [18], which we implemented in the same way as proposed by Zheng et al. [2]. The FM learns interactions between features in the input vector and is also well suited for such large and complex datasets as we encounter in this work. To apply this model to our binary classification problem, we feed the output of the FM into a sigmoid function. The output of the model can be interpreted as the probability that a user adds a comment to a given discussion at a particular time. Further, we use cross-entropy as the loss function. We refer to that adopted model in the following as DeepCoNN.

B. Utilizing User Co-Occurrences

Commenting on online news platforms is a highly social activity and involves interacting with other users' comments. By leaving a comment, users implicitly join a community of people who share an interest in that particular discussion. Thus, there exists a tie between these users. That phenomenon is related to the idea of homophily. Users that frequently participate in the same discussions are similar in the way that they co-occur in our dataset. If we encode that information of co-occurrence, we can compute the similarity between users that already participate in a discussion and the user for whom we want to generate recommendations.

To leverage this kind of information, we create an undirected bipartite graph. Nodes represent either users or discussions. A node that represents a user has an edge to a discussion node if that user participated in that discussion. Consequently, if users co-occur in a discussion, the length of the shortest path between their nodes in the graph is two.

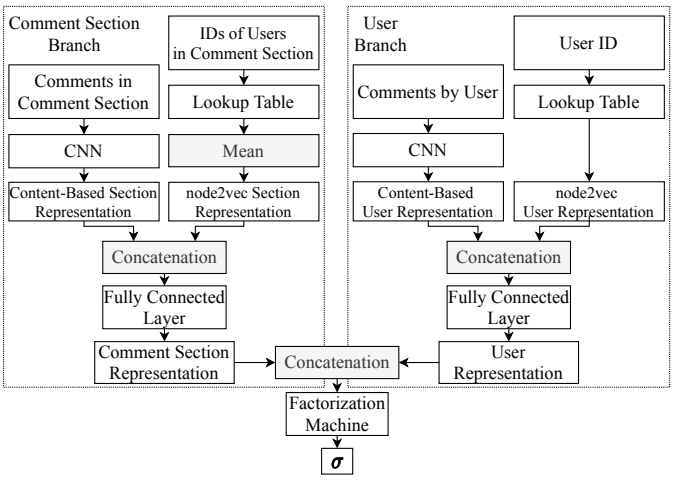

Fig. 1. HyCoNN comprises a content-based and a user-based model branch.

To obtain user embeddings, we use the node2vec framework [25]. An alternative would be embedding techniques tailored to bipartite graphs [27]. In the resulting embedding space, embeddings of users are more similar if the users co-occur frequently or their neighbors co-occur frequently. This is supported by the sampling strategy of node2vec which explores the graph using breadth-first or depth-first search. Consequently, the embedding pairs of users who share interests will appear more similar to each other. Moreover, the embeddings can depict the structure of the user communities.

C. HyCoNN Architecture

Figure 1 depicts how we incorporate the user embeddings learned with node2vec into the approach of jointly learning reader discussion and user representations with DeepCoNN. We name this model HyCoNN (Hybrid Cooperative Neural Networks) since it combines DeepCoNN (Deep Cooperative Neural Networks), which utilizes content, and the user embeddings learned with node2vec, which utilizes user co-occurrences. Our model consists of two branches: one to model discussions, i.e., the comments of articles, and one to model users.

To integrate the node2vec user embedding into the user branch of the network, we introduce a lookup table that translates a user ID into the corresponding embedding. After that, we concatenate the user representation learned from the text content using the CNN and the node2vec user embedding, and feed it into a fully connected layer with ReLU as its activation function. The output of that layer serves as the user representation.

On the other branch, the model translates all user IDs present in a discussion at a certain time into the learned node2vec embeddings using the same lookup table as in the first branch. After that, we calculate the mean of all user

embeddings. If there is no user embedding for any user in the comment section, we use a zero vector. To combine the output of the CNN on the comment texts in a discussion and the user embeddings, we concatenate them and feed them into another fully connected layer with ReLU as the activation function. The output of this layer is the discussion representation. For regularization, we apply dropout to the output of the fully connected layers of both branches. Similar to the DeepCoNN architecture, we use the concatenation of the resulting user and discussion representation as input to a factorization machine. We use a sigmoid function on the output and binary cross-entropy as the loss function. Similar to our DeepCoNN adaptation, the output can be interpreted as the probability that a user posts a comment. For training both DeepCoNN and HyCoNN, we use the Adam optimizer.

IV. Evaluation

We consider two real-world comment datasets for the evaluation. To ensure a realistic evaluation setting, we select appropriate relevant (positive) and irrelevant (negative) samples of discussions for each user in the training and test set. Appropriate means that we consider only those discussions and only those comments that were available at the time when the user visited the website. Our main experiment is based on a recommendation task for a hold-out set of comments, where we aim to predict to which discussion a user contributed by posting a comment. In addition to that, we evaluate the user embeddings of the different approaches with regard to pair-similarity correlation (PSC).

A. Datasets

Our first dataset comprises all user comments from Daily Mail, which span from 2009 until the end of 2018 and our second dataset contains all comments from Guardian, from 2008 until the end of 2018. We created a subset of these data because (1) we want to model only users who are still active; (2) we want to limit the dataset to comments that were posted under similar conditions (no platform changes, e.g., introduction of stronger moderation strategies); and (3) we want a time-wise split of training, validation, and test set, where every user who appears in the test set or the validation set also appears in the training set. To achieve this, we first sort all comments by publication timestamp of the corresponding news article. Then, we do a time-wise split. Thereby, during training, the model has no access to information from the future. Further, to avoid inconsistencies that could result from a new moderation policy introduced by Guardian in 2016, we limit the dataset to a subset of 2017. For Guardian training dataset, we select articles and comments published between March 1st and May 31st, 2017. For the validation set, we choose the time between June 1st and June 30th, and for the test set the time between July 1st and July 31st, 2017.

The Daily Mail dataset contains a lot more comments. Therefore, we set smaller time frames for this dataset to have similar sizes of the datasets for both news platforms. Consequently, we set the time frame for the training dataset to April 1st to May 31st, for the validation set to June 1st to June 15th, and for the test set to June 15th to June 30th, 2017. For both datasets, Guardian and the Daily Mail, we picked users for validation and testing who appear at least four times in the training dataset. Table I lists the sizes of the resulting datasets.

TABLE I
Size of comment datasets from The Guardian and Daily Mail.

	Guardian			Daily Mail		
	Users	Comments	Articles	Users	Comments	Articles
Training	111,961	2,278,816	13,419	128,927	2,892,083	22,906
Validation	63,042	690,229	4,004	98,413	1,415,855	10,070
Test	66,143	744,918	4,223	99,474	1,508,050	11,315

B. Recommendation Task

We evaluate the recommendation performance by using a hold-out set of comments and simulate the website's state at the time the user actually posted a comment. For every situation when a user was active in the test dataset, we recreate the state of the discussion at that time for one positive sample (a discussion where the user contributed) and 50 randomly chosen negative samples (discussions where the user did not contribute). The negative samples correspond to reader discussions at the time a user wrote the comment in the positive sample. We choose 50 negative samples because this is approxiamtely the number of daily discussions on the platforms. We consider different top-k recommendation settings with $k \in \{1, 3, 5, 10, 15\}$ and assess the performance with regard to recall and precision at k. In our scenario, recall is more important than precision because a single discussion is labeled as relevant but others — implicitly labeled as irrelevant — might actually also be relevant and might have been overlooked by the reader. Some related work uses ROC-AUC as an evaluation metric in similar scenarios [12], [16]. It corresponds to the probability that a relevant discussion is ranked higher than an irrelevant one. If a recommender system ranks a relevant discussion higher than most of the irrelevant discussions it achieves a good ROC-AUC score. However, recall and precision at k are better suited to evaluate that the top recommendations are relevant. A problem with ROC-AUC is that different recommender systems achieve similar scores if the same discussions are irrelevant to most users and therefore can be easily identified and ranked down, i.e., discussions about unpopular topics [16].

For data preprocessing, we pass all comments through a word tokenizer by NLTK and lowercase every token. We create a single vocabulary based on the training dataset to have a fair comparison of the neural network models and the baselines. It is the same for all methods: TF-IDF, DeepCoNN, and HyCoNN. We keep all tokens that occurred in no more than 50% of the comments and no less than five comments.

We only use comments from the training dataset to represent a user in our validation and testing process to mimic a realistic application scenario. In contrast to that, Catherine

and Cohen [28] describe that DeepCoNN's predictions of item ratings are only good if the review text by the target user for the target item is already known. This limitation is not acceptable in our scenario, which is why we strictly learn only based on the training dataset. In the training process of DeepCoNN, and HyCoNN, we omit a comment for computing the user representation if that comment was written to the respective discussion in the positive training sample. The omission of comments guarantees that the models do not learn direct relations between that comment and the respective comment sections. Otherwise, predictions on the validation or test dataset would not be comparable since those relations only appear in the training dataset.

The validation of node2vec uses a pairwise ranking task. We examine the similarity between a given user u and a corresponding reader discussion that functions as a positive sample S_p, and a corresponding reader discussion that serves as a negative sample S_n. If the similarity between the user embedding of u and the mean user embedding of the positive sample S_p is higher than the similarity between the user embedding of u and the mean user embedding of the negative sample S_n, the ranking is correct, otherwise, it is incorrect. During test time, we obtain recommendations by calculating ranking scores based on the cosine similarity of a user embedding and the mean user embeddings of participants in a given discussion.

a) Baselines: We implement a TF-IDF vector space model as a baseline approach. This method aims to rank reader discussions higher if the discussion is more similar to the comments the user wrote in the past. To this end, we use the previously created vocabulary and calculate the inverse document frequency for all terms in the training set. For the user representation, we average the TF-IDF vectors of the user's comment texts in the training dataset. For the representation of a discussion, we average the TF-IDF vectors of the comments present in that discussion at a particular point in time. Finally, the cosine similarity between the user representation and the discussion representation corresponds to the ranking score.

For a collaborative filtering (CF) baseline, we build a matrix for users and discussions on the training dataset. Each row represents a user, and each column represents a discussion. Each cell holds the number of times a user has commented in a specific discussion. This method computes higher scores for discussions, where the mean of the participants' representations is more similar to the user. It retrieves the representation of a user from its corresponding row in the matrix. To obtain a representation of a discussion, it calculates the mean of all representations of users who participated in a particular discussion. The ranking score is determined by calculating the cosine similarity between that mean representation of the participants and the representation of a user.

Since HyCoNN combines a content-based method and a user-based method, we use a fusion strategy to compare against the combination of the two baselines, content-based TF-IDF and user-based CF. To this end, we apply Reciprocal Rank Fusion (RRF) [29] for combining the individual rankings. Moreover, we also use RRF to combine the rankings produced by our node2vec-based approach and the DeepCoNN approach to compare whether the combination of both methods in HyCoNN is superior to a rank fusion strategy that combines the results of both methods. We refer to that approach as NDRF (Node2Vec DeepCoNN Rank Fusion) and to the combination of TF-IDF and CF as BRF (Baseline Rank Fusion).

b) User Representation and Graph Construction: For each article in the test datasets, we recreate the corresponding discussion for at least one positive sample and 50 negative samples. The resulting dataset includes $53,185$ different states of reader discussions for GUARDIAN and $59,125$ for DAILY MAIL. For GUARDIAN, we set the maximum number of comments to represent users to 42 and to 22 for DAILY MAIL. With these limits, we are able to represent 90% of the users in the training dataset from GUARDIAN and 80% of the users in the training dataset from DAILY MAIL with all their comments. For the minority of users who wrote more than 42, respectively 22, comments in the training set, we choose their newest comments to represent them. We maintain the temporal order of comments in the reader discussions. That means, we sort the comments by descending timestamp and concatenate the comments afterward.

To construct the bipartite graph, we use the 42 newest comments for GUARDIAN and the 22 newest comments for DAILY MAIL for every user in the respective training dataset. We include every user in the training dataset, no matter how many comments he or she wrote. The reason for that is that, although we only consider making recommendations to users that wrote at least four comments in the training datasets, users with fewer comments in the training dataset can also appear in the validation dataset or test dataset. Hence, the user embeddings of users with less than four comments can improve the prediction and recommendation performance for the test and validation users. The resulting graph for GUARDIAN has $125,172$ nodes and $639,317$ edges with an average degree of 10.22. The DAILY MAIL graph has $150,137$ nodes and $863,157$ edges with an average degree of 11.5.

To compare whether the CF baseline performs worse using only the 42 (respectively 22) newest comments to represent users, we consider limitations of the CF baseline. We refer to these baselines as CF42 and CF22 in the following and to the baseline that includes *all* comments as CF. Moreover, we also generate the user representations of users with less than three comments in the training dataset similar to our approach with the learned user embeddings using node2vec. Consequently, we can compare CF and node2vec in a fair manner.

c) Hyperparameter Optimization: For node2vec, we use Bayesian optimization to tune the number of walks per source $\in \{10, 20, 30\}$, the walk length $\in \{10, 20, 30, 50, 100\}$, the context window size $\in \{10, 20, 30\}$, and the embedding size $\in \{25, 50, 100\}$ on the validation dataset. On the GUARDIAN dataset, this optimization leads to 20 walks with a length of 10, a window size of 10, and 25-dimensional embeddings. We give equal weight to local and global structures by

setting node2vec's hyperparameters p and q to the default value 1. For tuning the hyperparameters of DeepCoNN, we use Bayesian Optimization with ten steps. The search space comprises the number of neurons in the convolutional layer $n \in \{25, 50, 100\}$, the window size $o \in \{2, 3, 4\}$, and the latent factors $l \in \{25, 50, 100\}$. We manually set the learning rate to 0.0001, the dropout to 0.1, and the batch size to 100. The model achieves the best accuracy on the validation dataset with $n = 50$, $o = 2$, and $l = 100$ with two epochs of training. For HyCoNN, we reuse the hyperparameters of DeepCoNN, and the user embeddings learned with node2vec. Furthermore, we initialize the weights and biases of the CNNs with the ones from the trained DeepCoNN model and keep the learning rate, batch size, and dropout. We tune only the number of neurons $r \in \{25, 50, 100, 150\}$ in the fully connected layer, which corresponds to the user and section embedding size. The model achieves the best accuracy after three epochs with $r = 100$.

We follow the same tuning approach on the DAILY MAIL dataset. For node2vec, Bayesian Optimization leads to the same settings as on the GUARDIAN dataset. For DeepCoNN, we also use the previous settings for learning rate, dropout, and batch size. Further, we use the same Bayesian Optimization experimentation settings as before for the DeepCoNN model on the GUARDIAN dataset. It leads to $n = 100$ neurons in the convolutional layer, a kernel size of $o = 3$, and $l = 50$. The best accuracy is achieved after training for one epoch. For HyCoNN, we reuse hyperparameters of DeepCoNN and embeddings from node2vec. The model achieves the best results with $r = 50$ after one epoch.

d) Results: Table II lists precision and recall at $k \in \{1, 3, 5, 10, 15\}$ for the recommendation task on the GUARDIAN dataset and Table III on the DAILY MAIL dataset. A baseline recommending random sections achieves a recall@1 of 0.020, a recall@3 of 0.059, etc. on that recommendation task because there are always 51 samples with one being relevant and 50 being irrelevant. The results show that combining DeepCoNN and node2vec in HyCoNN results in better recommendations than applying the methods individually. On the DAILY MAIL dataset, HyCoNN outperforms all other methods for every k. The training of layers in both CNNs in HyCoNN, in combination with the user embeddings, learned with node2vec, also yields better results than NDRF on the rankings of DeepCoNN and node2vec on both datasets.

However, the poor performance of node2vec for smaller k also results in CF outperforming HyCoNN for $k = 1$ and $k = 3$ on the GUARDIAN dataset. For larger k, node2vec learns competitive embeddings. It achieves even better results than CF for top-k recommendations with $k \geq 10$. A remarkable point is that the rank fusion strategy BRF, which combines TF-IDF with CF, results in worse recommendations on the GUARDIAN dataset than using CF alone. TF-IDF and CF generate very different rankings and their combination in BRF results in worse performance. However, BRF on the DAILY MAIL dataset yields better results than the baselines alone. The content-based methods DeepCoNN and TF-IDF perform on both datasets worse than the CF method.

TABLE II
PRECISION AND RECALL @k FOR $k \in \{1, 3, 5, 10, 15\}$ FOR THE RECOMMENDATION TASK ON THE GUARDIAN DATASET IN PERCENT.

	@1		@3		@5		@10		@15	
	P	R	P	R	P	R	P	R	P	R
CF	**35.1**	**35.1**	**16.0**	**48.0**	11.1	55.5	6.8	67.9	5.1	76.3
CF42	33.7	33.7	15.3	45.8	10.7	53.4	6.6	65.9	5.0	74.6
TF-IDF	13.1	13.1	8.2	24.5	6.5	32.4	4.7	46.6	3.9	58.0
BRF	25.3	25.3	13.7	41.1	10.2	51.0	6.7	66.9	5.1	76.8
node2vec	26.7	26.7	14.5	43.5	10.7	53.3	6.8	68.3	5.2	77.6
DeepCoNN	12.8	12.8	9.9	29.6	8.2	40.9	6.0	59.7	4.8	71.7
NDRF	24.3	24.3	14.6	43.7	11.0	55.0	7.2	72.2	5.5	82.0
HyCoNN	26.2	26.2	15.5	46.4	**11.5**	**57.5**	**7.4**	**73.9**	**5.6**	**83.5**

TABLE III
PRECISION AND RECALL @k FOR $k \in \{1, 3, 5, 10, 15\}$ FOR THE RECOMMENDATION TASK ON THE DAILY MAIL DATASET IN PERCENT.

	@1		@3		@5		@10		@15	
	P	R	P	R	P	R	P	R	P	R
CF	17.8	17.8	10.6	31.9	8.5	42.5	6.2	61.9	5.0	74.5
CF22	17.2	17.2	10.3	31.0	8.2	41.2	5.8	58.2	4.7	69.9
TF-IDF	10.2	10.2	7.7	23.2	6.6	33.2	5.1	51.4	4.3	64.5
BRF	18.4	18.4	12.0	35.9	9.5	47.5	6.6	65.8	5.1	76.9
node2vec	14.6	14.6	10.5	31.6	8.7	43.4	6.3	63.1	5.0	75.7
DeepCoNN	14.1	14.1	10.4	31.3	8.5	42.7	6.2	61.8	4.9	74.0
NDRF	19.0	19.0	13.0	39.0	10.4	51.8	7.1	71.3	5.5	82.0
HyCoNN	**22.1**	**22.1**	**15.2**	**45.5**	**11.9**	**59.4**	**7.8**	**78.1**	**5.8**	**86.9**

C. User Embedding Evaluation

We evaluate the user embeddings regarding whether they also depict the interests of users in categories that are defined by the news platforms. To this end, we describe how we adapt the pair-distance correlation [24] to evaluate embeddings quantitatively on the GUARDIAN dataset. We call our adapted method pair-similarity correlation (PSC) to distinguish it from the existing pair-distance correlation. Blandfort et al. [24] calculate the distance between two users in a user-space as the mean-squared difference of their ratings on movies. To make use of this idea, we construct implicit ratings of users for comment sections by calculating the number of times a user commented in a comment section. However, these representations are very sparse. To tackle that problem, we create *user category vectors* utilizing fine-grained categories of news articles, such as politics, sports, and environment. For every article category in the validation dataset, we count the number of times a user posted a comment on articles in that category. These user vectors have a length of 47 since there are 47 different article categories in the validation dataset. Similar to Blandfort et al. [24], we take the similarities of the embeddings in the user category space as the ground truth and compare them with the similarities of user representations learned by HyCoNN and DeepCoNN.

To this end, we use cosine similarity to calculate the similarity of users in the user category space and the similarity of users in the respective embedding space that we want to evaluate. To calculate PSC, we create a list of user pairs. We include user pairs that co-occur in at least five different

comment sections and user pairs that co-occur in less than five comment sections, equally. In this way, we ensure that user pairs with similar and dissimilar commenting history occur equally in the list. The result includes $510,334$ different user pairs. Following these choices, the pair-similarity correlation is computed as the Pearson correlation between the similarity scores of user pairs in the user category space (ground truth) and the respective embedding space to be evaluated. Hence, the best possible score is 1, and the worst is -1. A random embedding would achieve a score of 0.

With a PSC score of 73.8%, HyCoNN best preserves the similarities of users' interests in categories on GUARDIAN. There is no difference in the performance of node2vec (69.5%) and DeepCoNN (69.4%), which outperform CF (59.1%) and TF-IDF (37.6%). Note that this evaluation only compares user embedding approaches and therefore cannot include the rank fusion approaches from the recommendation task. Comparing the PSC results of CF with its recommendation results leads to the conclusion that good results in the recommendation task do not necessarily imply that the user embeddings preserve the similarities in the user category space. This finding is in line with Blandfort et al. [24]. The biggest difference to their method is that we compare similarities of embeddings in two different vector spaces. Further, we do not only evaluate vectors learned by neural networks but also vectors based on collaborative filtering and TF-IDF.

V. DISCUSSION

On the DAILY MAIL dataset, HyCoNN outperforms all other approaches for every k and on the GUARDIAN dataset for $k \in \{5, 10, 15\}$ with regard to precision@k and recall@k. To our surprise, for $k = 1$ and $k = 3$ on the GUARDIAN dataset, the CF baseline outperforms all other approaches. One reason might be our sampling strategy, which uses only implicit information to select negative samples. A user did not necessarily encounter every reader discussion that we selected as a negative sample. For instance, maybe the user did not encounter a discussion just because the corresponding article was not displayed on the main page of the news platform at the time the user was active. As a consequence, discussions that we flag as irrelevant could likely be relevant to the user, although he or she did not post a comment in it. Especially, as the models only perform worse for $k = 1$ and $k = 3$, it could likely be that some negative samples are in fact good recommendations. Therefore, although the models perform worse according to our evaluation metric, the recommendations might still be valuable.

Since the combination of CF and TF-IDF in BRF performs worse on the GUARDIAN dataset as compared to the DAILY MAIL dataset, a hybrid recommendation method for GUARDIAN is not necessarily the best strategy, which is also reflected in the results of HyCoNN. In contrast, on the DAILY MAIL dataset, BRF achieves better results than TF-IDF, and CF individually. We conclude that the hybrid recommendation methods, such as the model we propose, do not necessarily lead to much better results on every dataset.

The proportion of users in the test dataset that node2vec can represent with user embeddings can influence the recommendations of node2vec, HyCoNN, and NDRF. 90.4% of the comments in the test datasets of GUARDIAN were written by users that appear in the training dataset. Respectively, users in the DAILY MAIL training dataset wrote 89.3% of the comments in the corresponding test dataset. Since node2vec and CF can represent every user appearing in the training dataset, we can exclude that the proportion of users in the test dataset, for whom user embeddings exist, is influencing the results when comparing both evaluation datasets. The results show that the user embeddings learned with node2vec on the proposed bipartite graph are useful for recommendations and that their performance is on the same level as CF. However, memory-based CF strategies need a lot of runtime and memory, especially if the number of news articles is large. The approach we propose with node2vec overcomes these problems as it represents users in a 25-dimensional space, which is, compared to the CF baseline, low-dimensional.

We can conclude that jointly modeling the user and section representations in the DeepCoNN architecture yields better results than a naive content-based approach such as TF-IDF. Finally, HyCoNN consistently outperforms DeepCoNN, node2vec, and NDRF. This result means that learning from content while incorporating the user embeddings with HyCoNN outperforms not only the individual approaches but also their combination with a rank fusion strategy. The strong pair-similarity correlation (PSC) in our second experiment shows that the learned user embeddings are not only tailored to the prediction task. They also preserve the similarities of users who share an interest in specific article categories. As a limitation, note that the cosine similarity is affected by the curse of dimensionality. With increasing dimensionality, the calculated distance between different pairs of points becomes almost equal. In particular, the user embeddings generated with TF-IDF and CF could be vulnerable to this problem because of the high dimensionality of their embeddings. However, given the strong PSC for CF, we assume that it is not a problem. Therefore, we refrain from further optimizing TF-IDF, e.g., with the help of a dimensionality reduction method.

To give an impression of the embeddings, Figure 2 shows a two-dimensional UMAP [30] visualization of the user embeddings learned by HyCoNN on the GUARDIAN dataset. Users are colored by their favorite article category, which is the category where they posted the largest number of their comments. The visualization is limited to users whose favorite is one of the ten most popular categories. It shows that users with similar interests form clusters in the embedding space. For example, there is a separate cluster of users who are most interested in news about Australia. An analysis of the mean publication time of the comments also suggests that some users are in non-British time zones, such as Australian or U.S. time zones. Further, the clusters of users who are most interested in politics or business are close to each other and they overlap. Similarly, sports in general and football in particular are the favorite categories of users with similar embeddings.

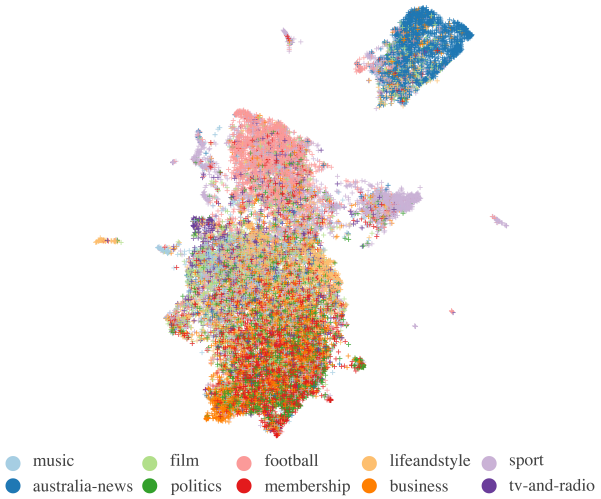

Fig. 2. UMAP projection of the user embeddings by HyCoNN. The colors correspond to a user's preferred article category.

VI. Conclusions and Future Work

In this work, we learned user representations on the news platforms Daily Mail and Guardian to recommend discussions to users. We introduced HyCoNN, a deep neural network for predicting if users post a comment given a specific reader discussion. This network jointly models users and discussions. It is trained to solve a binary classification problem, where we carefully selected positive and negative training samples of user-relevant discussions. Further, it incorporates user embeddings that we created using the node2vec framework on a graph of user co-occurrences within reader discussions.

Experimental results show that HyCoNN outperforms our baselines and state-of-the-art approaches. Text-only-based methods perform worse even compared to a collaborative filtering baseline or recommendations solely based on user embeddings learned with node2vec. In a quantitative evaluation of the embeddings, we find that HyCoNN preserves the similarity of user interests best compared to other methods. Future work could enrich HyCoNN's recommendations with explanations, e.g., based on attention mechanisms.

References

[1] N. J. Stroud, E. Van Duyn, and C. Peacock, "News commenters and news comment readers," 2016. [Online]. Available: https://mediaengagement.org/research/survey-of-commenters-and-comment-readers/

[2] L. Zheng, V. Noroozi, and P. S. Yu, "Joint deep modeling of users and items using reviews for recommendation," in *Proc. Int. Conf. on Web Search and Data Mining*, 2017, pp. 425–434.

[3] T. Yoneda, S. Kozawa, K. Osone, Y. Koide, Y. Abe, and Y. Seki, "Algorithms and system architecture for immediate personalized news recommendations," in *Proc. Int. Conf. on Web Intelligence*, 2019, pp. 124–131.

[4] J. Gao, X. Xin, J. Liu, R. Wang, J. Lu, B. Li, X. Fan, and P. Guo, "Fine-grained deep knowledge-aware network for news recommendation with self-attention," in *Proc. Int. Conf. on Web Intelligence*, 2018, pp. 81–88.

[5] E. V. Epure, B. Kille, J. E. Ingvaldsen, R. Deneckere, C. Salinesi, and S. Albayrak, "Recommending personalized news in short user sessions," in *Proc. Conf. on Recommender Systems*, 2017, pp. 121–129.

[6] H. Rahmatizadeh Zagheli, H. Zamani, and A. Shakery, "A semantic-aware profile updating model for text recommendation," in *Proc. Conf. on Recommender Systems*, 2017, pp. 316–320.

[7] D. Agarwal, B.-C. Chen, and B. Pang, "Personalized recommendation of user comments via factor models," in *Proc. Conf. on Empirical Methods in Natural Language Processing*, 2011, pp. 571–582.

[8] C.-F. Hsu, E. Khabiri, and J. Caverlee, "Ranking comments on the social web," in *Proc. Int. Conf. on Computational Science and Engineering*, 2009, pp. 90–97.

[9] J. Risch and R. Krestel, "Top comment or flop comment? predicting and explaining user engagement in online news discussions," in *Proc. Int. Conf. on Web and Social Media*, 2020, pp. 579–589.

[10] J. Risch and R. Krestel, "A dataset of journalists' interactions with their readership: When should article authors reply to reader comments?" in *Proc. Conf. on Information and Knowledge Management*, 2020, pp. 3117–3124.

[11] P. Aragón, V. Gómez, and A. Kaltenbrunner, "To thread or not to thread: The impact of conversation threading on online discussion," in *Proc. Int. Conf. on Web and Social Media*, 2017, pp. 12–21.

[12] T. Bansal, M. Das, and C. Bhattacharyya, "Content driven user profiling for comment-worthy recommendations of news and blog articles," in *Proc. Conf. on Recommender Systems*, 2015, pp. 195–202.

[13] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *Proc. Int. Conf. on Knowledge Discovery and Data Mining*, 2011, pp. 448–456.

[14] Q. Li, J. Wang, Y. P. Chen, and Z. Lin, "User comments for news recommendation in forum-based social media," *Information Sciences*, vol. 180, no. 24, pp. 4929–4939, 2010.

[15] N. X. Bach, N. D. Hai, and T. M. Phuong, "Personalized recommendation of stories for commenting in forum-based social media," *Information Sciences*, vol. 352-353, pp. 48–60, 2016.

[16] E. Shmueli, A. Kagian, Y. Koren, and R. Lempel, "Care to comment? recommendations for commenting on news stories," in *Proc. Int. Conf. on World Wide Web*, 2012, pp. 429–438.

[17] M. Aharon, A. Kagian, R. Lempel, and Y. Koren, "Dynamic personalized recommendation of comment-eliciting stories," in *Proc. Conf. on Recommender Systems*, 2012, pp. 209–212.

[18] S. Rendle, "Factorization machines," in *Proc. Int. Conf. on Data Mining*, 2010, pp. 995–1000.

[19] S. Seo, J. Huang, H. Yang, and Y. Liu, "Interpretable convolutional neural networks with dual local and global attention for review rating prediction," in *Proc. Conf. on Recommender Systems*, 2017, pp. 297–305.

[20] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[21] D. Liang, J. Altosaar, L. Charlin, and D. M. Blei, "Factorization meets the item embedding: Regularizing matrix factorization with item co-occurrence," in *Proc. Conf. on Recommender Systems*, 2016, pp. 59–66.

[22] D. Kim, C. Park, J. Oh, S. Lee, and H. Yu, "Convolutional matrix factorization for document context-aware recommendation," in *Proc. Conf. on Recommender Systems*, 2016, pp. 233–240.

[23] S. Amir, B. C. Wallace, H. Lyu, P. Carvalho, and M. J. Silva, "Modelling context with user embeddings for sarcasm detection in social media," in *Proc. Conf. on Computational Natural Language Learning*, 2016, pp. 167–177.

[24] P. Blandfort, T. Karayil, F. Raue, J. Hees, and A. Dengel, "Fusion strategies for learning user embeddings with neural networks," in *Proc. Int. Joint Conf. on Neural Networks*, 2019, pp. 1–8.

[25] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proc. Int. Conf. on Knowledge Discovery and Data Mining*, 2016, pp. 855–864.

[26] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions ACL*, vol. 5, pp. 135–146, 2017.

[27] M. Gao, L. Chen, X. He, and A. Zhou, "Bine: Bipartite network embedding," in *Proc. Int. Conf. on Research and Development in Information Retrieval*, 2018, pp. 715–724.

[28] R. Catherine and W. Cohen, "TransNets: Learning to transform for recommendation," in *Proc. Conf. on Recommender Systems*, 2017, pp. 288–296.

[29] G. V. Cormack, C. L. Clarke, and S. Buettcher, "Reciprocal rank fusion outperforms condorcet and individual rank learning methods," in *Proc. Int. Conf. on Research and Development in Information Retrieval*, 2009, pp. 758–759.

[30] L. McInnes, J. Healy, and J. Melville, "UMAP: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.