Unveiling the Hidden: Movie Genre and User Bias in Spoiler Detection

Haokai Zhang[1] *, Shengtao Zhang[1], Zijian Cai[2], Heng Wang[3], Ruixuan Zhu[1], Zinan Zeng[1], and Minnan Luo[3,4](✉)

[1] Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University, Xi'an 710049, China
{zhanghaokai,zhangst,1760865856,2194214554}@stu.xjtu.edu.cn
[2] Institute of Automation, Chinese Academy of Sciences, Beijing, 100190, China
caizijian2024@ia.ac.cn
[3] School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an 710049, China
wh2213210554@stu.xjtu.edu.cn
[4] Shaanxi Province Key Laboratory of Big Data Knowledge Engineering, Xi'an Jiaotong University, Xi'an 710049, China
minnluo@xjtu.edu.cn

Abstract. Spoilers in movie reviews are important on platforms like IMDb and Rotten Tomatoes, offering benefits and drawbacks. They can guide some viewers' choices but also affect those who prefer no plot details in advance, making effective spoiler detection essential. Existing spoiler detection methods mainly analyze review text, often overlooking the impact of movie genres and user bias, limiting their effectiveness. To address this, we analyze movie review data, finding genre-specific variations in spoiler rates and identifying that certain users are more likely to post spoilers. Based on these findings, we introduce a new spoiler detection framework called `GUSD` (**G**enre-aware and **U**ser-specific **S**poiler **D**etection), which incorporates genre-specific data and user behavior bias. User bias is calculated through dynamic graph modeling of review history. Additionally, the R2GFormer module combines RetGAT (Retentive Graph Attention Network) for graph information and GenreFormer for genre-specific aggregation. The GMoE (Genre-Aware Mixture of Experts) model further assigns reviews to specialized experts based on genre. Extensive testing on benchmark datasets shows that GUSD achieves state-of-the-art results. This approach advances spoiler detection by addressing genre and user-specific patterns, enhancing user experience on movie review platforms. Our source code is available at https://github.com/AI-explorer-123/GUSD

Keywords: Spoiler Detection · Movie Genre · User Bias · Mixture-of-Experts

* H. Zhang and S. Zhang contributed equally to this work.

1 Introduction

Spoilers in movie reviews have become an important component of the movie-viewing experience on popular platforms like IMDb and Rotten Tomatoes [5]. For those who hope to learn the plot of the movie in advance to judge whether they like it or not, the spoilers are helping, while for those who prefer to experience a movie without prior knowledge, the spoilers can severely diminish the enjoyment by revealing crucial plot points, undermining suspense, and eliciting negative emotions among viewers [21]. Thus, effective spoiler detection methods are crucial for maintaining a positive user experience.

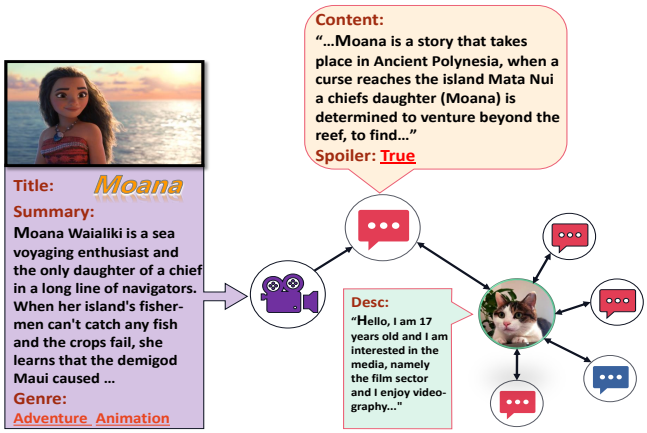

Fig. 1. An illustrative example of the data used in our spoiler detection study. The image shows a review of the movie *Moana*, including the movie's genres (Adventure, Animation), summary, the review's content, and user-specific details. All reviews from the user are color-coded: blue indicates non-spoiler content, while red indicates spoiler content.

Existing spoiler detection methods primarily focus on the textual content of reviews. For example, DNSD [6] integrates review sentences and movie genres, while SpoilerNet [36] utilizes a Hierarchical Attention Network and incorporates the item-specificity information. More recent approaches, such as MVSD [38], incorporate advanced techniques like syntax-aware graph neural networks and external movie knowledge to improve detection performance. Nevertheless, these methods still exhibit notable limitations. Solely relying on textual content proves insufficient for robust spoiler detection [38]. Moreover, spoilers are often genre-specific, with varying characteristics depending on the movie's genre — for instance, suspense films focus on plot details, whereas action films emphasize fight scenes. As a result, two significant challenges in spoiler detection remain unaddressed:

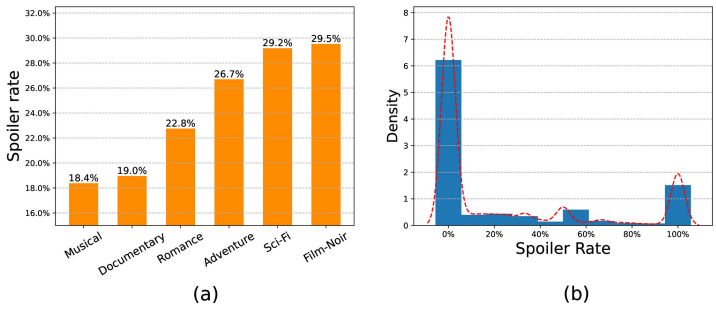

Fig. 2. (a) Spoiler rate across different movie genres (partial) in LCS dataset. (b) Kernel density estimation plot and distribution histogram of spoiler across different users.

- **Diverse Genres**. Previous works have largely ignored the impact of movie genres on the spoiler rate. Our analysis of the dataset indicates substantial differences in spoiler rate across genres, with specific categories defined according to IMDb standards. As shown in Figure 2(a), movies that heavily rely on plot twists and suspense, such as Film-Noir and Adventure, are more prone to having spoilers in reviews compared to genres like Musical or Documentary. This is understandable since plot-driven movies tend to have more critical plot points that can be spoiled. This variation in spoiler rate demonstrates the importance of considering genre-specific characteristics when developing spoiler detection models. By incorporating genre information, we could better capture these differences and improve the performance of spoiler detection.
- **User-specific Behavior Bias**. User behavior varies significantly, with some users being more prone to posting spoilers than others. Our statistical analysis shows a clear trend where certain users tend to post spoiler reviews more frequently. As illustrated in Figure 2(b), the graph of user spoiler rate distribution shows that a large proportion of users post very few spoilers, while a smaller, yet significant, group of users frequently post spoilers. This distribution indicates a noticeable user bias, highlighting that certain users are more likely to post spoilers than others. Leveraging these user-specific behavior bias can improve the detection performance by allowing models to adapt to user behavior bias.

To address the challenges of genre-specific spoiler tendencies and user bias in spoiler detection, we propose a comprehensive framework named `GUSD` (**G**enre-aware and **U**ser-specific **S**poiler **D**etection). This framework integrates genre information, user behavior bias, and global perception GNN. Our method begins with preprocessing movie, user, and review data. Then user bias is captured from review history through dynamic graph modeling. After that, the core component, R2GFormer (RetGAT and GenreFormer), which consists of RetGAT (Retentive Graph Attention Network) and GenreFormer, processes the graph

information. RetGAT aggregates all the features globally, and GenreFormer enhances representation by integrating genre features, allowing for comprehensive genre-specific and cross-genre interactions. And an Aggregator combines these features, and then the GMoE (Genre-Aware Mixture of Experts) model assigns reviews to different experts based on their corresponding movie genres, improving traditional MoE model performance. Finally, a classifier performs spoiler detection using the aggregated features.

Extensive experiments show that `GUSD` achieves state-of-the-art performance. We also conduct robustness studies, ablation studies, and specific experiments on GMoE and user bias to validate our proposed modules' effectiveness.

Our main contributions are summarized as follows:

- We are the first to model the complex interactions between genre-specific information and long-term user review behaviors for spoiler detection, providing a nuanced approach by understanding genre-specific spoiler characteristics and leveraging user behavior bias.
- We propose the `GUSD` framework, a novel spoiler detection system that integrates several key components: the GenreFormer to capture genre-specific spoiler tendencies, the GMoE model to dynamically assign reviews based on genres, and dynamic graph modeling to capture user bias. This cohesive integration enhances overall accuracy and robustness.
- Our method `GUSD` achieves state-of-the-art performance in spoiler detection. Extensive experiments on two benchmark datasets demonstrate its robustness and effectiveness, showing superior performance across various conditions.

2 Related Work

2.1 Spoiler Detection

The goal of automatic spoiler detection is to identify spoilers in reviews from domains like television [3], books [36], and movies [3]. Existing approaches to spoiler detection can be broadly classified into three categories: keyword matching methods, machine learning techniques, and deep learning models.

Keyword matching methods. These approaches rely on a set of predefined keywords to identify spoilers. Examples include keywords related to sports teams or events [25], or actors' names [12]. Although useful in specific scenarios, this method requires manual keyword definition and lacks generalizability across different application contexts.

Machine learning techniques. These methods often involve topic modeling or support vector machines using handcrafted features. For example, Guo et al. [13] applied a bag-of-words representation combined with an LDA-based model for spoiler detection. Jeon et al. [16] developed an SVM classifier incorporating four extracted features, while Boyd et al. [3] utilized lexical features and meta-data of review subjects (e.g., movies and books) in an SVM model.

Deep learning models. These models mainly leverage NLP techniques, employing RNNs, LSTMs, Transformer, and language models to process review texts and movie information through end-to-end training. Bao et al. [2] utilized LSTMs, BERT, and RoBERTa for sentence-level spoiler detection. DNSD [6] focused on incorporating external genre information using GRU and CNN. SpoilerNet [36] introduced item-specificity and bias with bi-RNN enhanced by GRU. SDGNN [7] leveraged dependency relations between context words in sentences with graph neural networks to capture semantics.

While some existing methods incorporate genre information and user bias [6, 40], they often rely on quite simple techniques such as concatenating or adding these additional features to the initial review features. Such approaches lack the sophistication needed for more effective and intricate modeling of genre features and user biases.

2.2 Mixture of Experts

The Mixture of Experts (MoE) approach, grounded in the Divide-and-Conquer principle, segments an input sample into sub-tasks and trains specialized experts for each sub-task. This method is extensively utilized in NLP to boost model capacity [33] and enhance reasoning capabilities [23]. Shazeer et al. [33] introduced a sparsely-gated Mixture-of-Experts layer, enabling conditional computing in large language models. Fedus et al. [10] developed simplified routing algorithms for MoE to enhance training stability and reduce computational costs. Furthermore, Soft-MoE [29] was introduced to mitigate issues like training instability and token dropping inherent in traditional MoE approaches.

Despite these advancements, traditional MoE methods assign tokens dynamically, which can cause incorrect token assignment, particularly when the dataset contains explicit category information such as movie genres and their associated reviews.

3 Methodology

Figure 3 shows the architecture of our proposed `GUSD` framework. This framework integrates genre-specific information, user behavior bias, and global receptive RetGAT for spoiler detection. Specifically, movie, user, and review data are first to be preprocessed, while user bias is extracted from review history using dynamic graph modeling. The R2GFormer component, consisting of RetGAT and GenreFormer, then processes graph features. RetGAT aggregates graph data, while GenreFormer handles genre-specific data. An Aggregator aggregates these features. The GMoE model assigns reviews to experts based on their related movie genres. Finally, a classifier utilizes the aggregated features to detect spoilers.

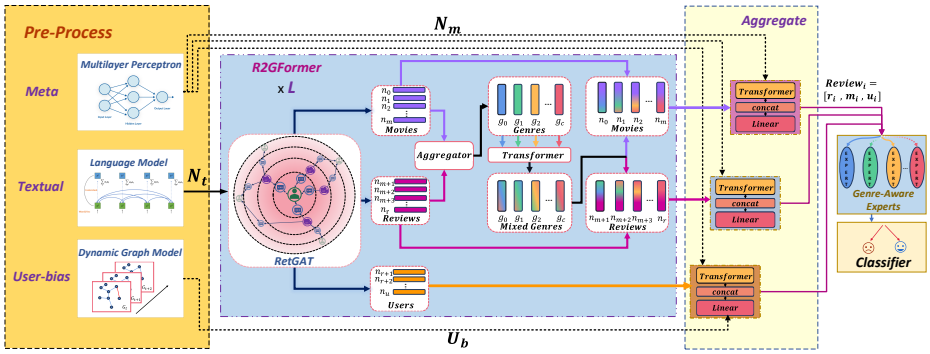

Fig. 3. Overview of our proposed GUSD framework, which integrates genre-specific information, user behavior bias, and global perceptive RetGAT for spoiler detection. It preprocesses movie, user, and review data with MLP and language models, and captures user bias via dynamic graph modeling. Then the data is processed by the R2GFormer component. An Aggregator merges these features, and then GMoE assigns reviews to experts based on genres. Finally, a classifier detects spoilers using the aggregated features.

3.1 Data preprocessing

Meta information. For review nodes, user nodes, and movie nodes, each type possesses metadata (details can be found in the supplementary material). After collecting the metadata for all three types of nodes, we pad them to the same length. A two-layer MLP is then employed as the meta encoder, producing the meta embeddings N_m.

Textual information. The textual content is fundamental for effective spoiler detection. To generate high-quality embeddings, we leverage an LM as our text encoder. Specifically, we augment the initial textual content with the textual descriptions of the node's metadata. This augmentation enriches the embeddings by providing additional contextual information about the node. Subsequently, we employ the LM to encode the nodes' textual information. The encoded embeddings are then transformed using a single-layer MLP, producing refined embeddings N_t.

User bias acquisition using dynamic-graph pre-training. To better capture the dynamic information attributes of users, we adopt the dynamic graph to handle users' review history flexibly. Thus, we need to convert the static dataset into a dynamic format. We utilize the given connections between different nodes and the time information of the reviews to form the dynamic event stream. The details about the formation of the graph will be displayed in Section 3.2. Considering the absence of additional information about the dynamic edges, we simply initialize the edge features as zero vectors. Then, we employ the robust DyGFormer [42] as our dynamic graph encoder to capture the interactions among

various nodes and utilize Link Prediction as the downstream task. We specifically obtain the features of user nodes as user bias U_b from the dynamic graph encoder.

3.2 R2GFormer

After acquiring the initial meta and textual features of the nodes, we use the textual features N_t, which include rich information about the nodes, as the initial embeddings $N_g^{(0)}$ for **R2GFormer**. This graph encoder not only models the complex relations and interactions between users, reviews, and movies but also handles features from different genres of the entire graph and captures remote dependencies. First, we will introduce how we construct the whole graph. Then, we decompose an R2GFormer layer into two parts: RetGAT and Genreformer, which will be introduced respectively.

Graph Construction. We first construct a directed graph consisting of three types of nodes: { *User*, *Review*, *Movie* } and the following three types of edges:

E1: *Movie-Review* We connect a review node to a movie node if the review is about the movie, but not vice versa. This setup allows movie information to influence the review while ensuring that the review information does not affect the movie.

E2: *Review-User* We connect a review node to a user node if the review is posted by the user.

E3: *User-Review* We connect a user node to a review node if the user posts the review.

RetGAT. Inspired by the work of [34, 9, 26], we propose RetGAT, which extends the RetNet framework by integrating a global perception into GAT, incorporating explicit exponential decay for nodes within k hops and truncation for nodes beyond k hops. This method ensures a broad receptive field while balancing performance and computational complexity by dynamically adjusting the influence of nodes based on their distance, with closer nodes having a higher impact on the final node features.

To achieve this global receptive field, we utilize k parallel GAT layers to separately aggregate information from k-hop neighbors. For each layer, node features are aggregated from different k-hop neighborhoods, applying decay factors to control the influence of information from various hops. Beyond k hops, the influence of nodes is truncated to maintain computational efficiency and focus on relevant information within the k-hop range. Note that we correct the algorithm that previous work [1, 37] used to compute k-hop neighbors, and the details are available in the supplementary material.

The decay factor δ_h for hop h ($h \leq k$) is defined as:

$$\delta_h = \exp(-\alpha \cdot h), \tag{1}$$

where α is a hyperparameter controlling the intensity of exponential decay. For the l-th layer, the contribution of the h-th hop neighbors is computed as:

$$N_{g,h}^{(l)} = \delta_h \cdot \mathrm{GAT}_h^{(l)}(\mathcal{A}_h, N_g^{(l-1)}), \tag{2}$$

where $\mathcal{A}_h$ represents the adjacency matrix at the h-th hop, and ${N_g}^{(l-1)}$ denotes the input node features of the $(l-1)$-th layer.

The final node features ${N_g}^{(l)}$ for layer l are aggregated from all the k-hop contributions using an Aggregator function:

$$N_g^{(l)} = \mathrm{AGGREGATOR}_r^{(l)}(N_{g,1}^{(l)}, N_{g,2}^{(l)}, \ldots, N_{g,h}^{(l)}, \ldots, N_{g,k}^{(l)}). \tag{3}$$

where $\mathrm{AGGREGATOR}_r^{(l)}$ is the aggregator for RetGAT at the l-th layer, which can be summation, concatenation, or a TransformerEncoder (TRM).

Genreformer. After message passing among different k-hop neighbors, Genreformer aims to pay attention to the entire graph to extract more comprehensive features of different genres. First, the global genre-specific representation is obtained by aggregating the features of all the review and movie nodes belonging to the same genre in the l-th layer, which is as follows:

$$g_j^{(l)} = \mathrm{AGGREGATOR}_g^{(l)}(\{n_{g,i}^{(l)} \mid i \in \mathcal{N}_j\}), \tag{4}$$

where $n_{g,i}^{(l)}$ is the feature of node i in the l-th layer, $\mathcal{N}_j$ denotes the set of nodes (both review and movie nodes) belonging to the j-th genre, and $\mathrm{AGGREGATOR}_g^{(l)}$ is the aggregator of Genreformer in the l-th layer, which can be summation, concatenation, or a TRM.

Next, we use a TRM to facilitate inter-genre information interaction, allowing each genre to acquire information from similar genres to further enrich its features:

$$[g_1^{(l)} g_2^{(l)} \cdots g_j^{(l)} \cdots g_c^{(l)}] = \mathrm{TRM}([g_1^{(l)} g_2^{(l)} \cdots g_j^{(l)} \cdots g_c^{(l)}]), \tag{5}$$

where c is the number of genres.

After interaction among genres, the genre features are fused with the movies and reviews nodes features. Then we incorporate the nodes with their genre features. Since some movies and reviews cover more than one genre, we take the average of the genre features a node covers. Subsequently, we concatenate the review or movie feature and its genre feature, then use an MLP to project it into the desired feature space, *i.e.*,

$$z_i^{(l)} = \mathrm{MEAN}(\{g_j^{(l)} \mid j \in G_i\}), \tag{6}$$

$$n_{g,i}^{(l)} = \mathrm{MLP}([n_{g,i}^{(l)} \parallel z_i^{(l)}]). \tag{7}$$

where G_i denotes the set of genres to which node i belongs, $n_{g,i}^{(l)}$ denotes the feature of node i in the l-th layer, $z_i^{(l)}$ is the aggregated genre feature for node i in the l-th layer, and $\|$ means concatenation. The MLP projects the concatenated feature into the desired feature space.

Overall interaction. One layer of our proposed R2GFormer layer, however, cannot enable the information interaction between all information sources. In order to further facilitate the interaction among the nodes, we employ $L\times$ R2GFormer layers for node representation learning. The representation of the nodes is updated after each layer, incorporating information from different sources. This process can be formulated as follows:

$$N_g^{(l)} = \text{R2GFormer}(N_g^{(l-1)}, \mathcal{E}, G). \tag{8}$$

where $\mathcal{E}$ denotes all the edges of the sampling graph, G denotes the genres of every movie and review node. After $L\times$ R2GFormer layers, we obtain the final graph representation $N_g^{(L)}$ of all nodes.

3.3 Multimodal Fusion

Through the data above processed by the R2GFormer, we have obtained the graph structural information and meta information for all nodes, including user, review and movie nodes. The next step is to fuse the multimodal information.

For each type of node, we utilize a type-wise TRM to facilitate inter-modal information interaction, then concatenate features of different modals followed by an MLP to get the final representation for each node, *i.e.*,

$$U_g, U_m, U_b = \text{TRM}([U_g\ U_m\ U_b], U = \text{MLP}([U_g \parallel U_m \parallel U_b])), \tag{9}$$

$$R_g, R_m = \text{TRM}([R_g\ R_m], R = \text{MLP}([R_g \parallel R_m])), \tag{10}$$

$$M_g, M_m = \text{TRM}([M_g\ M_m], M = \text{MLP}([M_g \parallel M_m]), \tag{11}$$

where U_g, R_g and M_g are derived from N_g, U_m, R_m and M_m are dervied from N_m.

After obtaining the comprehensive representation of each type of node, we then concatenate each review feature r_i with its corresponding movie feature m_i and user feature u_i:

$$r_i = [\ r_i \parallel m_i \parallel u_i\ ]. \tag{12}$$

3.4 GMoE

Inspired by the successful applications of Mixture-of-Experts in NLP and bot detection, and its capability to handle the small subsets of the whole dataset, we adopt MoE to handle different genres of reviews. However, distinct from the latent subsets of the dataset in the classic MoE application scenario, our datasets

already have the genre information of movies, as well as their related reviews. So we improve MoE to the proposed GMoE.

Specifically, instead of using the gating mechanism in the traditional MoE structure, we assign tokens to experts simply according to their genres: which genre it belongs to, which expert will deal with it; how many genres it belongs to, how many experts will deal with it.

$$r_i = \text{AGGREGATOR}_m(\{\text{Expert}_j(r_i) \mid \forall j \in G_i\}). \tag{13}$$

where G_i denotes the set of genres to which node i belongs; AGGREGATOR$_m$ can be summation, concatenation, or a TRM; each Expert is a MLP for simplicity.

3.5 Learning and Optimization

After using GMoE to process genre-specific information, we acquire the final representation r_i for the i-th review. Then we apply a linear transformation to r_i to obtain spoiler detection result $\hat{y}_i$. To train `GUSD`, We optimize the network by cross-entropy loss with L_2 regularization. The total loss function is as follows:

$$Loss = -\sum_{i \in \mathcal{R}} y_i \log \hat{y}_i + \lambda \sum_{\theta \in \Theta} \theta^2. \tag{14}$$

where $\hat{y}_i$ and y_i are the prediction for the i-th review and its corresponding ground truth, respectively. $\mathcal{R}$ encompasses all the reviews in the training set, while Θ denotes all trainable model parameters in `GUSD`, and λ is a hyper-parameter that maintains the balance between the two parts.

4 Experiment

4.1 Experiment Settings

Dataset. To evaluate our `GUSD` framework along with 14 other representative baselines on two widely recognized datasets: **LCS** [38] and **Kaggle** [24]:

- **LCS** is a comprehensive dataset for automatic spoiler detection, comprising 1,860,715 reviews, 259,705 users, and 147,191 movies. And about 24.59% (457,500) of the reviews are spoilers.
- **Kaggle**, introduced in 2019, consists of 573,913 valid reviews, 263,407 users, and 1,572 movies. And about 25.87% (150,924) of the reviews are spoilers.

Note that both datasets include the genre information of all movies, with specific categories defined according to IMDb standards, which facilitates the operation of our `GUSD` framework. Following MVSD [38], we randomly split the reviews into training, validation, and test sets with a ratio of 7:2:1.

Baselines. To achieve a comprehensive evaluation, we compare GUSD with pre-trained language models, GNN-based models, and task-specific baselines. For the pre-trained language models, the procedure involves feeding the review text into the model, averaging all token embeddings, and then applying two fully connected layers to perform spoiler detection. Regarding the GNN-based models, the graph neural network takes the output of RoBERTa [19] as the initial node features. Below, we provide a concise overview of each baseline method.

- **BERT** [8] is a language model pre-trained on extensive natural language data, using masked language modeling and next sentence prediction tasks.
- **RoBERTa** [19] improves upon BERT by eliminating the next sentence prediction task and enhancing masking techniques.
- **BART** [18] is a pre-trained language model that advances traditional autoregressive models through bidirectional encoding and denoising objectives.
- **DeBERTa** [14] refines BERT by implementing disentangled attention and an improved mask decoder, making it a more advanced language model.
- **Bge-Large** [41] is trained on a comprehensive training dataset C-MTP, combining vast unlabeled data and diverse labeled data.
- **GCN** [17] is a foundational graph neural network that performs convolutions on graph nodes and their neighbors, effectively propagating information.
- **R-GCN** [32] extends GCN to handle multi-relational graphs by incorporating relation-specific weights.
- **GAT** [35] is a graph neural network that applies attention mechanisms to dynamically assign importance to neighboring nodes.
- **SimpleHGN** [22] is tailored for heterogeneous graphs, integrating multiple types of nodes and edges with a shared embedding space and adaptive aggregation strategies.
- **GPS** [30] propose a recipe to build a general, powerful, scalable graph Transformer with linear complexity.
- **HGT** [15] design node- and edge-type dependent parameters to characterize the heterogeneous attention over each edge for modeling Web-scale heterogeneous graphs.
- **DNSD** [6] is a spoiler detection method that employs a CNN-based genre-aware attention mechanism.
- **SpoilerNet** [36] uses a hierarchical attention network and GRU alongside item and user bias terms for spoiler detection.
- **MVSD** [38] leverages external movie knowledge and user networks to detect spoilers.

4.2 Main Results

We evaluated our GUSD framework and 14 other baselines on two datasets. The results presented in Table 1 demonstrate the following:

- GUSD **consistently outperforms all baselines across both datasets**. Specifically, compared with the previous state-of-the-art method MVSD [38],

Table 1. Accuracy, AUC, and binary F1-score of GUSD and three types of baseline methods on two spoiler detection datasets. We run all experiments **five times** to ensure a consistent evaluation and report the average performance as well as standard deviation in parentheses. Bold indicates the best performance, underline the second best. GUSD consistently outperforms the three types of methods on both benchmarks.

Model	Kaggle			LCS		
	F1	AUC	Acc	F1	AUC	Acc
BERT	44.02 ±1.09	63.46 ±0.46	77.78 ±0.09	46.14 ±2.84	64.82 ±1.36	79.96 ±0.38
RoBERTa	50.93 ±0.76	66.94 ±0.40	79.12 ±0.10	47.72 ±0.44	65.55 ±0.22	80.16 ±0.03
BART	46.89 ±1.55	64.88 ±0.71	78.47 ±0.09	48.18 ±1.22	65.79 ±0.62	80.14 ±0.07
DeBERTa	49.94 ±1.13	66.42 ±0.59	79.08 ±0.09	47.38 ±2.22	65.42 ±1.08	80.13 ±0.08
Bge-Large	52.51 ±0.58	67.74 ±0.37	77.44 ±0.21	52.68 ±0.36	68.46 ±0.23	79.24 ±0.11
GCN	59.22 ±1.18	71.61 ±0.74	82.08 ±0.26	62.12 ±1.18	73.72 ±0.89	83.92 ±0.23
R-GCN	63.07 ±0.81	74.09 ±0.60	82.96 ±0.16	62.99 ±0.89	76.18 ±0.72	85.19 ±0.21
GAT	60.98 ±0.09	72.72 ±0.60	82.43 ±0.01	65.73 ±0.12	75.92 ±0.13	85.18 ±0.02
SimpleHGN	60.12 ±1.04	71.60 ±0.88	82.08 ±0.26	63.79 ±0.88	74.64 ±0.64	84.66 ±1.61
HGT	63.99 ±0.25	75.61 ±0.25	81.66 ±0.23	60.89 ±0.46	73.96 ±0.53	81.86 ±0.16
GPS	61.04 ±0.84	73.50 ±0.53	81.25 ±0.55	64.21 ±0.30	75.60 ±0.92	82.40 ±0.91
DNSD	46.33 ±2.37	64.50 ±1.11	78.44 ±0.14	44.69 ±1.64	64.10 ±0.74	79.76 ±0.08
SpoilerNet	57.19 ±0.69	70.64 ±0.44	79.85 ±0.10	62.86 ±0.38	74.62 ±0.69	83.23 ±0.23
MVSD	65.08 ±0.69	75.42 ±0.56	83.59 ±0.11	69.22 ±0.61	78.26 ±0.63	86.37 ±0.08
Ours	**80.24** ±0.73	**87.00** ±0.37	**89.65** ±0.36	**75.37** ±0.10	**83.71** ±0.27	**88.32** ±0.08

GUSD achieves **6.1%** higher Binary-F1, **5.5%** higher AUC, and **2.0%** higher accuracy on the LCS dataset, as well as **15.2%** higher Binary-F1, **11.6%** higher AUC, and **6.1%** higher accuracy on the Kaggle dataset. These improvements are statistically significant.

- In both datasets, **graph-based models generally outperform other types of baselines**, reaffirming the importance of analyzing the graph structure of reviews and their corresponding users and movies.
- Compared with DNSD [6], which also focuses on genre features of the reviews, GUSD surpasses DNSD across all three metrics in both datasets, further proving the effectiveness and robustness of our global-aware genreformer and GMoE methods.
- Both SpoilerNet [36] and GUSD utilize user bias, but GUSD outperforms SpoilerNet in all three metrics across both datasets. This demonstrates that our dynamic graph pretraining can better identify the latent behavior pattern of whether a user is likely to post spoilers.

4.3 Ablation Study

As GUSD outperforms all the baselines and has reached state-of-the-art (SOTA) performance across the two datasets, we conducted ablation study to further explore the impact of each part of GUSD on the final performance with the Kaggle Dataset. The results are shown in Table 2.

- To assess the importance of user bias information, we removed the user bias component U_b. The results in Table 2 show an obvious decrease in

Table 2. Ablation study of GUSD on Kaggle Dataset. Bold indicates the best performance, underline the second best.

Catagory	Ablation Settings	F1	AUC	Acc
User bias	-w/o U_b	78.85	85.78	88.58
RetGAT	approximate way	78.40	85.39	88.39
	normal GAT	78.08	85.22	88.29
GMoE	MLP	78.52	86.09	88.25
	MoE [33]	78.75	86.36	88.54
	Soft MoE [29]	79.10	86.10	88.78
Genreformer	-w/o genreformer	77.73	84.68	88.13
	sum pooling	<u>79.84</u>	<u>86.56</u>	<u>89.28</u>
	max pooling	78.77	85.52	89.13
Ours	GUSD	**80.24**	**87.00**	**89.65**

performance, confirming that user bias information is critical for effective spoiler detection.

- For the RetGAT component, we evaluated two variations: using an approximate method to compute k-hop neighbors [1, 37] and replacing our RetGAT with a standard GAT. Both variations lead to a drop in performance, indicating the necessity of our RetGAT design for capturing appropriate graph structures.
- To investigate the impact of GMoE, we replace it with a simple MLP, a traditional MoE [33], and a Soft-MoE [29]. Note that we set the number of experts of the traditional MoE and Soft-MoE to be the same as in GMoE, *i.e.*, the number of genres. From the results shown in Table 2, the full model with GMoE achieves the best performance, highlighting the effectiveness of our GMoE design in handling explicit genre information. Moreover, replacing the GMoE with an MLP performs the worst, proving the rationality of using genre information.
- We also evaluate the Genreformer component by removing it and replacing the mean AGGREGATOR with sum pooling and max pooling. The results in Table 2 show that the full Genreformer with sum AGGREGATOR outperforms these ablated versions, validating the necessity of the Genreformer.

4.4 GMoE Study

In this section, we conduct further experiments on the GMoE to better understand its effectiveness compared to other previous MoE methods such as traditional MoE [33] and Soft-MoE [29].

Different numbers of Experts. We conduct multiple experiments by varying the number of experts in traditional MoE and Soft-MoE, recording the Binary-

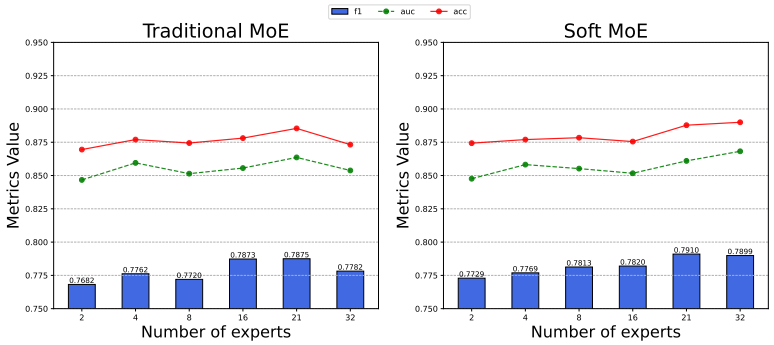

Fig. 4. Performance comparison of different numbers of experts in traditional MoE and Soft-MoE. Note that 21 is the number of genres. The results indicate that GMoE outperforms other variants irrespective of the number of experts.

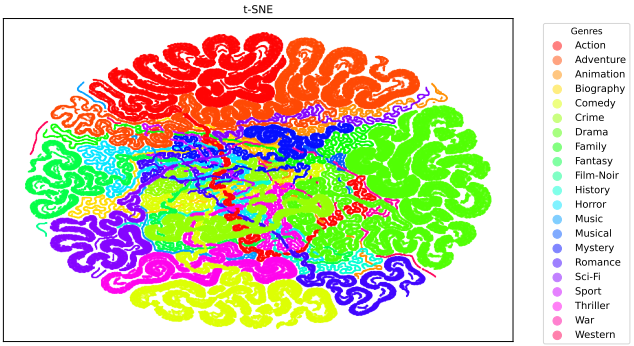

Fig. 5. T-SNE visualization of the features processed by GMoE. Different colors represent different genres, indicating distinct clustering of features according to genres.

F1, AUC, and Accuracy for each configuration. The results are summarized in Figure 4.

From the results shown in Figure 4, it is evident that the GMoE consistently outperforms both traditional MoE and Soft-MoE regardless of the number of experts used. Specifically, Soft-MoE performs better than traditional MoE. That is because GMoE uses explicit genre information, eliminating inaccurate dispatch that can occur in traditional MoE and Soft-MoE.

T-SNE Visualization of Features. To further validate the effectiveness of the GMoE, we perform dimensionality reduction on the features of all reviews processed by the GMoE layer. Specifically, we first reduce the dimensionality of the features to 50 dimensions using PCA, and then further reduce them to 2 dimensions using T-SNE. The result plot, shown in Figure 5, demonstrates that

reviews of different genres exhibit distinct features after being processed by the GMoE.

The T-SNE visualization in Figure 5 indicates that reviews are clustered according to their genres, providing compelling evidence of the effectiveness of our GMoE design. These distinct clusters suggest that GMoE successfully captures and utilizes genre-specific features to enhance its performance. In the visualization, each distinct color represents a different genre. For instance, genres like 'Action' (red), 'Drama' (green), and 'Sci-Fi' (purple) form well-defined clusters, indicating that the features of reviews from these genres are significantly different from each other. The presence of these distinct clusters reaffirms the model's capability to differentiate and leverage genre-specific information effectively.

5 Conclusion

In this paper, we introduce `GUSD`, a novel spoiler detection framework that integrates Genreformer and GMoE to effectively model diverse genre features. Additionally, `GUSD` incorporates dynamic graph pretraining to capture user bias related to spoiler posting. Extensive experiments reveal that `GUSD` significantly outperforms the state-of-the-art models on two major spoiler detection benchmarks. Further analysis validates the effectiveness of our proposed techniques, demonstrating `GUSD`'s superior capability in capturing intricate genre features and modeling user bias for spoiler detection.

Acknowledgments. This work was supported by the National Nature Science Foundation of China (No. 62272374, No. 62192781), the Natural Science Foundation of Shaanxi Province (2024JC-JCQN-62), the National Nature Science Foundation of China (No. 62202367, No. 62250009), the Key Research and Development Project in Shaanxi Province No. 2023GXLH-024, Project of China Knowledge Center for Engineering Science and Technology, and Project of Chinese academy of engineering "The Online and Offline Mixed Educational Service System for 'The Belt and Road' Training in MOOC China", and the K. C. Wong Education Foundation. Lastly, we would like to thank all LUD lab members for fostering a collaborative research environment.

References

1. Atwood, J., Towsley, D.: Diffusion-convolutional neural networks. Advances in neural information processing systems **29** (2016)
2. Bao, A., Ho, M., Sangamnerkar, S.: Spoiler alert: Using natural language processing to detect spoilers in book reviews. arXiv preprint arXiv:2102.03882 (2021)
3. Boyd-Graber, J., Glasgow, K., Zajac, J.S.: Spoiler alert: Machine learning approaches to detect social media posts with revelatory information. Proceedings of the American Society for Information Science and Technology **50**(1), 1–9 (2013)
4. Cai, Z., Tan, Z., Lei, Z., Zhu, Z., Wang, H., Zheng, Q., Luo, M.: Lmbot: distilling graph knowledge into language model for graph-less deployment in twitter bot detection. In: Proceedings of the 17th ACM International Conference on Web Search and Data Mining. pp. 57–66 (2024)

5. Cao, Y., Wang, X., He, X., Hu, Z., Chua, T.S.: Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences. In: The world wide web conference. pp. 151–161 (2019)
6. Chang, B., Kim, H., Kim, R., Kim, D., Kang, J.: A deep neural spoiler detection model using a genre-aware attention mechanism. In: Advances in Knowledge Discovery and Data Mining: 22nd Pacific-Asia Conference, PAKDD 2018, Melbourne, VIC, Australia, June 3-6, 2018, Proceedings, Part I 22. pp. 183–195. Springer (2018)
7. Chang, B., Lee, I., Kim, H., Kang, J.: " killing me" is not a spoiler: Spoiler detection model using graph neural networks with dependency relation-aware attention mechanism. arXiv preprint arXiv:2101.05972 (2021)
8. Devlin, J.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
9. Fan, Q., Huang, H., Chen, M., Liu, H., He, R.: Rmt: Retentive networks meet vision transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5641–5651 (2024)
10. Fedus, W., Zoph, B., Shazeer, N.: Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. Journal of Machine Learning Research **23**(120), 1–39 (2022)
11. Fey, M., Lenssen, J.E.: Fast graph representation learning with pytorch geometric. arXiv preprint arXiv:1903.02428 (2019)
12. Golbeck, J.: The twitter mute button: a web filtering challenge. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp. 2755–2758 (2012)
13. Guo, S., Ramakrishnan, N.: Finding the storyteller: automatic spoiler tagging using linguistic cues. In: Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010). pp. 412–420 (2010)
14. He, P., Gao, J., Chen, W.: Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. arXiv preprint arXiv:2111.09543 (2021)
15. Hu, Z., Dong, Y., Wang, K., Sun, Y.: Heterogeneous graph transformer. In: Proceedings of the web conference 2020. pp. 2704–2710 (2020)
16. Jeon, S., Kim, S., Yu, H.: Don't be spoiled by your friends: Spoiler detection in tv program tweets. In: Proceedings of the International AAAI Conference on Web and Social Media. vol. 7, pp. 681–684 (2013)
17. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
18. Lewis, M.: Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. arXiv preprint arXiv:1910.13461 (2019)
19. Liu, Y.: Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692 **364** (2019)
20. Liu, Y., Tan, Z., Wang, H., Feng, S., Zheng, Q., Luo, M.: Botmoe: Twitter bot detection with community-aware mixtures of modal-specific experts. In: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 485–495 (2023)
21. Loewenstein, G.: The psychology of curiosity: A review and reinterpretation. Psychological bulletin **116**(1), 75 (1994)
22. Lv, Q., Ding, M., Liu, Q., Chen, Y., Feng, W., He, S., Zhou, C., Jiang, J., Dong, Y., Tang, J.: Are we really making much progress? revisiting, benchmarking and

refining heterogeneous graph neural networks. In: Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining. pp. 1150–1160 (2021)
23. Madaan, A., Tandon, N., Rajagopal, D., Clark, P., Yang, Y., Hovy, E.: Think about it! improving defeasible reasoning by first modeling the question scenario. arXiv preprint arXiv:2110.12349 (2021)
24. Misra, R.: IMDB Spoiler Dataset. Available at https://www.kaggle.com/datasets/rmisra/imdb-spoiler-dataset (2019)
25. Nakamura, S., Tanaka, K.: Temporal filtering system to reduce the risk of spoiling a user's enjoyment. In: Proceedings of the 12th international conference on Intelligent user interfaces. pp. 345–348 (2007)
26. Nikolentzos, G., Dasoulas, G., Vazirgiannis, M.: K-hop graph neural networks. Neural Networks **130**, 195–205 (2020)
27. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems **32** (2019)
28. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.: Scikit-learn: Machine learning in python. the Journal of machine Learning research **12**, 2825–2830 (2011)
29. Puigcerver, J., Riquelme, C., Mustafa, B., Houlsby, N.: From sparse to soft mixtures of experts. arXiv preprint arXiv:2308.00951 (2023)
30. Rampášek, L., Galkin, M., Dwivedi, V.P., Luu, A.T., Wolf, G., Beaini, D.: Recipe for a general, powerful, scalable graph transformer. Advances in Neural Information Processing Systems **35**, 14501–14515 (2022)
31. Rau, D.: Sparsely-gated mixture-of-experts pytorch implementation (2019)
32. Schlichtkrull, M., Kipf, T.N., Bloem, P., Van Den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: The semantic web: 15th international conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, proceedings 15. pp. 593–607. Springer (2018)
33. Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., Dean, J.: Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. arXiv preprint arXiv:1701.06538 (2017)
34. Sun, Y., Dong, L., Huang, S., Ma, S., Xia, Y., Xue, J., Wang, J., Wei, F.: Retentive network: A successor to transformer for large language models. arXiv preprint arXiv:2307.08621 (2023)
35. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
36. Wan, M., Misra, R., Nakashole, N., McAuley, J.: Fine-grained spoiler detection from large-scale review corpora. arXiv preprint arXiv:1905.13416 (2019)
37. Wang, G., Ying, R., Huang, J., Leskovec, J.: Multi-hop attention graph neural network. arXiv preprint arXiv:2009.14332 (2020)
38. Wang, H., Zhang, W., Bai, Y., Tan, Z., Feng, S., Zheng, Q., Luo, M.: Detecting spoilers in movie reviews with external movie knowledge and user networks. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. pp. 16035–16050 (2023)
39. Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al.: Transformers: State-of-the-art natural language processing. In: Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations. pp. 38–45 (2020)
40. Wróblewska, A., Rzepiński, P., Sysko-Romańczuk, S.: Spoiler in a textstack: How much can transformers help? arXiv preprint arXiv:2112.12913 (2021)

41. Xiao, S., Liu, Z., Zhang, P., Muennighof, N.: C-pack: packaged resources to advance general chinese embedding. 2023. arXiv preprint arXiv:2309.07597 (2023)
42. Yu, L., Sun, L., Du, B., Lv, W.: Towards better dynamic graph learning: New architecture and unified library. Advances in Neural Information Processing Systems **36**, 67686–67700 (2023)