MMoE: Robust Spoiler Detection with Multi-modal Information and Domain-aware Mixture-of-Experts

Anonymous ACL submission

Abstract

Online movie review websites are valuable for information and discussion about movies. However, the massive spoiler reviews detract from the movie-watching experience, making spoiler detection an important task. Previous methods simply focus on reviews' text content, ignoring the heterogeneity of information in the platform. For instance, the metadata and the corresponding user's information of a review could be helpful. Besides, the spoiler language of movie reviews tends to be genre-specific, thus posing a domain generalization challenge for existing methods. To this end, we propose MMoE, a multi-modal network that utilizes information from multiple modalities to facilitate robust spoiler detection and adopts Mixture-of-Experts to enhance domain generalization. MMoE first extracts graph, text, and meta feature from the user-movie network, the review's textual content, and the review's metadata respectively. To handle genre-specific spoilers, we then adopt Mixture-of-Experts architecture to process information in three modalities to promote robustness. Finally, we use an expert fusion layer to integrate the features from different perspectives and make predictions based on the fused embedding. Experiments demonstrate that MMoE achieves state-of-the-art performance on two widely-used spoiler detection datasets, surpassing previous SOTA methods by 2.56% and 8.41% in terms of accuracy and F1-score. Further experiments also demonstrate MMoE's superiority in robustness and generalization.

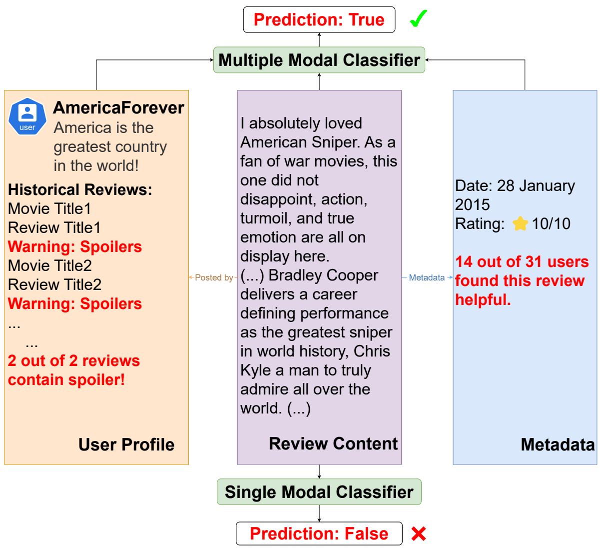


Figure 1: The information of a spoiler review from multiple sources. The text-based detection method struggles to identify whether this review is a spoiler. However, we can identify the review to be a spoiler by jointly considering the reviewer's historical preference and the review's metadata. The red font indicates the information which helps determine whether the review contains spoilers.

1 Introduction

Movie websites such as IMDb and Rotten Tomato have served as popular social platforms facilitating commentary, discussion, and recommendation about movies (Cao et al., 2019). However, there are a substantial amount of reviews that reveal the critical plot in advance on these websites, known as *spoilers*. Spoilers diminish the suspense and surprise of the movie and may evoke negative emotions in the users (Loewenstein, 1994). Therefore, it is necessary to propose an effective spoiler detection method to protect users' experience.

Existing spoiler detection methods mainly focus on the textual content. Chang et al. (2018) encode review sentences and movie genres together to detect spoilers. Wan et al. (2019) incorporate

Hierarchical Attention Network (Yang et al., 2016) and introduce user bias and item bias. Chang et al. (2021) exploit syntax-aware graph neural networks to model dependency relations in context words. Wang et al. (2023) take into account external movie knowledge and user interactions to promote effective spoiler detection.

However, there are still some limitations in the proposed approaches so far. Firstly, solely relying on the textual content is inadequate for robust spoiler detection (Wang et al., 2023). We argue that integrating multiple information sources (metadata, user profile, movie synopsis et al.) is necessary for reliable spoiler detection. For instance, as shown in Figure 1, it is challenging to discern whether this review contains spoilers solely based on its textual content. However, this reviewer can be correctly identified as a spoiler through the analysis of historical reviews and the establishment of a user profile for this reviewer. In addition, the vote count in metadata also suggests that the review is a potential spoiler. Secondly, the spoiler language tends to be genre-specific as people's focus varies depending on the genre of movies, resulting in distinct characteristics in their reviews. Specifically, for science fiction films, individuals tend to focus on the quality of special effects. In the case of action movies, the fight scenes become the primary highlight. On the other hand, for suspense movies, the plot takes precedence. Consequently, there is a significant variation of the spoilers in reviews across different domains. Existing methods fail to differentiate these reviews with varying styles, posing challenges in adapting to the increasingly diverse landscape of spoiler reviews.

To address these challenges, we propose MMoE (Multi-modal Mixture-of-Experts), which leverages multi-modal information and domain-aware Mixture-of-Experts. Specifically, we start training multiple encoders for different types of information by using a series of pretext tasks. Next, we use these models to obtain the features of reviews from graph view, text view, and meta view. We then adopt Mixture-of-Experts (MoE) to assign the information from different aspects to certain domains. Finally, we use a transformer encoder to combine the information from all three perspectives. Experiments demonstrate that MMoE achieves state-of-the-art performance on two widely-used spoiler detection datasets, surpassing previous SOTA methods by 2.56% and 8.41% in terms of accuracy and F1-score. Further extensive experiments also validate our design choices.

2 Related Work

Spoiler detection aims to automatically detect spoiler reviews in television (Boyd-Graber et al., 2013), books (Wan et al., 2019), and movies (Wang et al., 2023), thereby protecting users' experiences. Earlier methods usually design handcrafted features and apply a traditional classifier. Guo and Ramakrishnan (2010) use bag-of-words embeddings and LDA model (Blei et al., 2003) to detect spoilers in movie comments. Boyd-Graber et al. (2013) combine lexical features with metadata features and use an SVM model (Cortes and Vapnik, 1995) as the classifier. Recently, deep learning based detection methods have dominated. Chang et al. (2018) propose a model with a genre-aware attention mechanism. However, they don't take into account fine-grained movie text information. Wan et al. (2019) develop SpoilerNet which uses HAN (Hierarchical Attention Network) (Yang et al., 2016) to learn sentence embeddings and then applies GRU (Cho et al., 2014) on top of it. SpoilerNet also considers user bias and item bias. However, they simply model them as learnable vectors. Chang et al. (2021) use bi-directional LSTM (Hochreiter and Schmidhuber, 1997) to extract word features and feed the embedding into graph neural network to pass and aggregate messages on the dependency graph. However, it is worth noting that the authors only incorporate the movie's genre information at the final pooling stage. These methods basically use RNN-based networks (such as LSTM and GRU) as text encoders, and review contents are the primary or even the only reference information. Wang et al. (2023) first introduce user network and external movie knowledge into spoiler detection task and validate its effectiveness. However, their approach falls short of adequately leveraging user information and adopts a simplistic encoding strategy for the text, relying solely on average pooling.

Given the limitations of the above work, we develop a comprehensive framework which leverages multi-modal information and the domain-aware Mixture-of-Experts for robust and generalizable spoiler detection. Our method MMoE establishes a new state-of-the-art in spoiler detection.

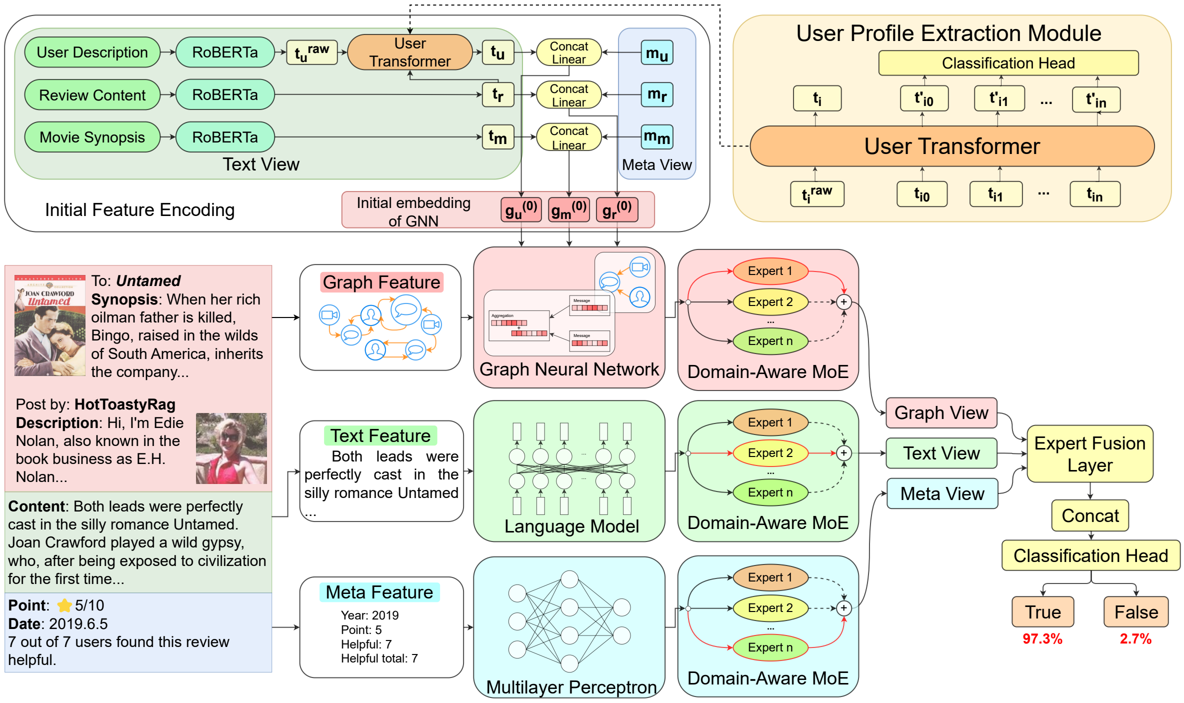


Figure 2: MMoE: a multi-modal mixture-of-experts framework that jointly leverages the review's metadata, text, and graph features for robust and generalizable spoiler detection. Metadata, text, and graph information are first processed by modal-specific encoders and then fed into Mixture-of-Experts layers. The user profile extraction module is employed to analyze the reviewer's historical preference and learn an embedding for the user. Finally, an expert fusion layer is adopted to integrate the three information sources and classify spoilers.

3 Methodology

The overall architecture of MMoE is illustrated in Figure 2. Specifically, we first encode the review's meta, text, and graph information to obtain comprehensive representations from three perspectives. We also propose a user profile extraction module which learns from the reviewer's historical reviews and analyzes the reviewer's preference. To deal with genre-specific spoilers, we then adopt Mixture-of-Expert (MoE) architecture (Jacobs et al., 1991; Shazeer et al., 2017) to process features in different modalities. MoE is able to assign reviews with different characteristics to different experts for robust classification (Liu et al., 2023). To facilitate information interaction, we finally use an expert fusion layer to integrate the information from the three perspectives and classify whether the review is a spoiler.

3.1 Modal-specific Feature Encoder

Metadata Encoder. The metadata associated with spoiler reviews tends to differ from that of regular reviews. Consequently, we gather the review metadata as auxiliary information for classification. Details of metadata are illustrated in Appendix A. Once this numerical information is collected, we employ a two-layer MLP as the meta encoder.

Text Encoder. The textual content plays a crucial role in spoiler detection. To obtain high-quality embeddings, we employ RoBERTa (Liu et al., 2019) as our text encoder. Initially, we fine-tune RoBERTa through a binary classification task using the textual content of reviews, which ensures that the model is specifically tailored for our spoiler detection task. Subsequently, we utilize the fine-tuned RoBERTa to encode the review content and transform the encoded embedding with a single-layer MLP.

Graph Encoder. To model the complex relations and interactions between user, review, and movie, we employ graph neural network to update the review feature through the corresponding user feature and movie feature. We first construct a directed graph consisting of the following three types of nodes and three types of edges:

N0: *User*.

N1: *Movie*.

N2: *Reviews*.

E1: *Movie-Review* We connect a review node with

a movie node if the review is about the movie.
E2: *User-Review* We connect a review node with a user node if the user posts the review.
E3: *Review-User* We use this type of edge to enable message passing between reviews.

For movie and review nodes, we encode their synopsis and review content respectively by the fine-tuned RoBERTa as the input feature. For user nodes, we design a user profile extraction module (Section 3.2) to extract their profiles as the initial feature. Initial node features are transformed by a linear layer followed by a ReLU activation, *i.e.*,

$$\mathbf{g}_i^{(0)} = \max(\mathbf{W}_{in} \cdot [\mathbf{t}_i, \mathbf{m}_i] + \mathbf{b}_{in}, 0),$$

where $\mathbf{m}_i$, $\mathbf{t}_i$ and $\mathbf{g}_i^{(0)}$ denote metadata features, text features and the initial embedding in the graph of node i. $[\cdot, \cdot]$ denotes the concatenation operation. $\mathbf{W}_{in}$ and $\mathbf{b}_{in}$ are parameters of the linear layer. We then use Graph Attention Network (GAT) (Velickovic et al., 2017) as the graph encoder to obtain the embedding of reviews from the graph modality, *i.e.*,

$$\mathbf{g}_i^{(l+1)} = \alpha_{i,i}\mathbf{\Theta}_s\mathbf{g}_i^{(l)} + \sum_{j \in N(i)} \alpha_{i,j}\mathbf{\Theta}_t\mathbf{g}_j^{(l)},$$

$$\alpha_{i,j} = \frac{\exp\left(f(\mathbf{a}_s^T\mathbf{\Theta}_s\mathbf{g}_i^{(l)} + \mathbf{a}_t^T\mathbf{\Theta}_t\mathbf{g}_j^{(l)})\right)}{\sum_{k \in N(i) \cup i} \exp\left(f(\mathbf{a}_s^T\mathbf{\Theta}_s\mathbf{g}_i^{(l)} + \mathbf{a}_t^T\mathbf{\Theta}_t\mathbf{g}_j^{(l)})\right)}$$

where f denotes the Leaky ReLU activation function. $\mathbf{g}_i^{(l)}$ is the embedding of node i in layer l. $N(i)$ is the neighbors of node i. In the directed graph, $N(i)$ denotes all nodes which point to node i. $\alpha_{i,j}$ is the attention score between node i and node j. $\mathbf{\Theta}_s \in \mathbb{R}^{d_{in} \times d_{out}}$, $\mathbf{\Theta}_t \in \mathbb{R}^{d_{in} \times d_{out}}$, $\mathbf{a}_s \in \mathbb{R}^{d_{in}}$, $\mathbf{a}_t \in \mathbb{R}^{d_{in}}$ are learnable parameters. d_{in} and d_{out} are the dimension of input vector and output vector, respectively.

We add a ReLU activation function between every GAT layer. After L layers of GAT, we obtain the review embeddings from the graph view.

3.2 User Profile Extraction Module

Since users normally have their preferences, they either infrequently or frequently post spoiler reviews. The specific proportion of spoiler reviews per user can be found in Appendix A, which illustrates this bias in detail. Therefore, capturing user preferences through their profiles can significantly aid in spoiler detection. While using users' self-descriptions is a direct approach to obtain their profiles, unluckily most users do not provide descriptions on film websites. Therefore, the initial information of user nodes is often missing in the graph. In light of this challenge, we model this kind of user preference by obtaining a learned user profile embedding through a user profile extraction module which takes the user's historical reviews as input and outputs a summarizing embedding indicating the user's preference.

To be specific, we concatenate the raw semantic features of users and the semantic features of their reviews into a sequence, *i.e.*,

$$\mathbf{s}_i = [\mathbf{t}_i^{raw}, \mathbf{t}_{i_1}, \mathbf{t}_{i_2}, \cdots, \mathbf{t}_{i_n}]$$

where $\mathbf{t}_i^{raw}$ is the raw text feature of the i-th user's description encoded by RoBERTa, $\mathbf{t}_{i_1}, \mathbf{t}_{i_2}, \cdots, \mathbf{t}_{i_n}$ are the text feature of the first, second, $\cdots$ and the last review of user i. $\mathbf{s}_i$ is the input sequence of the module. Since the number of reviews per user can vary, we employ the "maximum length" strategy. Sequences shorter than the maximum length are padded with zero vectors, while sequences longer than the maximum length are truncated to ensure uniform length.

After obtaining the input sequence, we use a transformer encoder (Vaswani et al., 2017) to get the output sequence. The encoder summarizes the user's historical reviews and utilizes self-attention mechanisms to learn a comprehensive profile embedding that reflects the user's preference. We pre-train the encoder by attaching a classification head after each review embedding, *i.e.*,

$$\mathbf{s}_i' = \mathrm{TRM}(\mathbf{s}_i),$$
$$\hat{\mathbf{p}}_i = \mathrm{softmax}(\mathbf{W}_u \cdot \mathbf{s}_i' + \mathbf{b}_u),$$

where $\mathbf{s}_i'$ is the output sequence; $\hat{\mathbf{p}}_i$ is the predicted output. We only compute the loss for the reviews within the training set.

After pre-training, we use the encoder to perform forward propagation on all sequences and extract the first embedding in the sequence (corresponding to the position of the user's raw profile feature in the input) as the user's profile feature, denoted as $\mathbf{t}_i$. The embedding will then be fixed in the model by

$$\mathbf{t}_i = \mathbf{s}_i'[0].$$

3.3 Domain-Aware MoE Layer

Inspired by the successful applications of Mixture-of-Experts in NLP and bot detection (Shazeer et al., 2017; Fedus et al., 2022; Liu et al., 2023), we adopt MoE to divide and conquer the information in the three modalities. Since spoiler reviews exhibit distinct characteristics across different genres of movies, we leverage the MoE framework, activating different experts to handle different reviews belonging to various domains. We calculate the weight G_j of each expert E_j as the same as Shazeer et al. (2017). Each expert E_j is a 2-layer MLP, *i.e.*,

$$\mathbf{z}_i^{mod} = \sum_{j=1}^{n} G_j(\mathbf{x}_i^{mod}) E_j(\mathbf{x}_i^{mod}),$$

where $\mathbf{x}_i^{mod}$ is the input embedding of review i, $\mathbf{z}^{mod}$ is the output feature, and $mod \in \{m, t, g\}$.

3.4 Expert Fusion Layer

After obtaining the review's representations processed by domain-aware experts in three modalities, we further combine the representations in three modalities by a multi-head transformer encoder to facilitate modality interaction, *i.e.*,

$$\mathbf{u}_i = [\mathbf{z}_i^m, \mathbf{z}_i^t, \mathbf{z}_i^g],$$
$$\mathbf{v}_i = \mathrm{TRM}(\mathbf{u}_i),$$

where $\mathbf{z}_i^m$, $\mathbf{z}_i^t$, $\mathbf{z}_i^g$ are features from the meta view, text view, and graph view respectively. $\mathbf{u}_i$ represents the concatenated sequence and $\mathbf{v}_i$ denotes the output sequence by the transformer encoder. We finally flatten $\mathbf{v}_i$ and apply a linear output layer to classify, *i.e.*,

$$\hat{\mathbf{y}}_i = \mathbf{W}_o \cdot \mathrm{flatten}(\mathbf{v}_i) + \mathbf{b}_o.$$

3.5 Learning and Optimization

We optimize the network by cross-entropy loss with L_2 regularization and balancing loss. The total loss function is as follows:

$$Loss = -\sum \mathbf{y}_i \log \hat{\mathbf{y}}_i + \lambda \sum \theta^2 + w \sum_{mod}^{m,t,g} BL(\mathbf{x}_i^{mod}),$$

where $\hat{\mathbf{y}}_i$ and $\mathbf{y}_i$ are the prediction for i-th review and its corresponding ground truth, respectively. θ denotes all trainable model parameters, and λ and w are hyperparameters which maintain the balance among the three parts. For balancing loss $BL(\mathbf{x}) = CV(\sum_i G(\mathbf{x}_i))^2$, where CV denotes the coefficient of variation, $G(\mathbf{x}_i)$ denotes the calculated weight of each expert, we refer to Shazeer et al. (2017) to encourage each expert to receive a balanced sample of reviews.

4 Experiment

4.1 Experiment Settings

Dataset. We evaluate our method `MMoE` on **LCS** dataset (Wang et al., 2023) and **Kaggle IMDB Spoiler** dataset (Misra, 2019). We follow the same dataset split method as Wang et al. (2023). Specific details of datasets can be found in Appendix A.

Baselines. We use the same baselines as in Wang et al. (2023). Specifically, we explore three kinds of approaches: PLM(Pre-trained Language Model)-based methods, GNN(Graph Neural Network)-based methods, and task-specific methods. For PLM-based methods, We evaluate BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), BART (Lewis et al., 2019) and DeBERTa (He et al., 2021). For GNN-based methods, we evaluate GCN (Kipf and Welling, 2016), R-GCN (Schlichtkrull et al., 2018), GAT (Velickovic et al., 2017), and Simple-HGN (Lv et al., 2021). For task-specific moethods, we evaluate DNSD (Chang et al., 2018), SpoilerNet (Wan et al., 2019), and MVSD (Wang et al., 2023). Specific details of baselines can be found in Appendix D.

Implementation Details. We use Pytorch (Paszke et al., 2019), Pytorch Geometric (Fey and Lenssen, 2019), scikit-learn (Pedregosa et al., 2011), and Transformers (Wolf et al., 2020) to implement `MMoE`. The hyperparameter settings and architecture parameters are shown in Appendix B. We conduct our experiments on a cluster with 4 Tesla V100 GPUs with 32 GB memory, 16 CPU cores, and 377GB CPU memory.

4.2 Overall Performances

We evaluate our proposed `MMoE` and other baseline methods on the two datasets. The results presented in Table 1 demonstrate that:

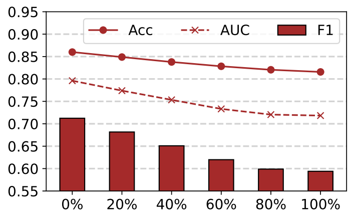
- `MMoE` achieves state-of-the-art on both datasets, outperforming all other methods by at least 8.41% in F1-score, 5.07% in AUC, and 2.56% in accuracy. This illustrates that `MMoE` is not only more accurate but also much more robust than former approaches.

- GNN-based methods significantly outperform other types of baselines. This confirms our view that using text information alone is not enough in spoiler detection. Social network information from movies and users is also very important.

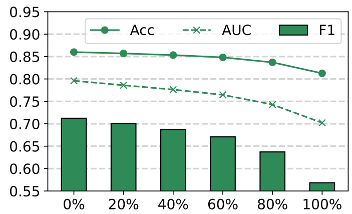
- For task-specific baselines, SpoilerNet (Wan et al., 2019) outperforms DNSD (Chang et al.,

Table 1: Accuracy, AUC, and binary F1-score of `MMoE` and other baselines on the two datasets. We repeat all experiments five times and report the average performance with standard deviation. **Bold** indicates the best performance, underline the second best. `MMoE` significantly outperforms the previous state-of-the-art method on two benchmarks on all metrics.

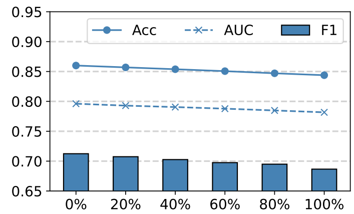
Model	**Kaggle**			**LCS**		
	F1	**AUC**	**Acc**	**F1**	**AUC**	**Acc**
BERT (Devlin et al., 2018)	44.02 ($\pm$1.09)	63.46 ($\pm$0.46)	77.78 ($\pm$0.09)	46.14 ($\pm$2.84)	65.55 ($\pm$1.36)	79.96 ($\pm$0.38)
RoBERTa (Liu et al., 2019)	50.93 ($\pm$0.76)	66.94 ($\pm$0.40)	79.12 ($\pm$0.10)	47.72 ($\pm$0.44)	65.55 ($\pm$0.22)	80.16 ($\pm$0.03)
BART (Lewis et al., 2019)	46.89 ($\pm$1.55)	64.88 ($\pm$0.71)	78.47 ($\pm$0.06)	48.18 ($\pm$1.22)	65.79 ($\pm$0.62)	80.14 ($\pm$0.07)
DeBERTa (He et al., 2021)	49.94 ($\pm$1.13)	66.42 ($\pm$0.59)	79.08 ($\pm$0.09)	47.38 ($\pm$2.22)	65.42 ($\pm$1.08)	80.13 ($\pm$0.08)
GCN (Kipf and Welling, 2016)	59.22 ($\pm$1.18)	71.61 ($\pm$0.74)	82.08 ($\pm$0.26)	62.12 ($\pm$1.18)	73.72 ($\pm$0.89)	83.92 ($\pm$0.23)
R-GCN (Schlichtkrull et al., 2018)	63.07 ($\pm$0.81)	74.09 ($\pm$0.60)	82.96 ($\pm$0.09)	66.00 ($\pm$0.99)	76.18 ($\pm$0.72)	85.19 ($\pm$0.21)
GAT (Velickovic et al., 2017)	60.98 ($\pm$0.09)	72.72 ($\pm$0.06)	82.43 ($\pm$0.01)	65.73 ($\pm$0.12)	75.92 ($\pm$0.13)	85.18 ($\pm$0.02)
SimpleHGN (Lv et al., 2021)	60.12 ($\pm$1.04)	71.61 ($\pm$0.74)	82.08 ($\pm$0.26)	63.79 ($\pm$0.88)	74.64 ($\pm$0.64)	84.66 ($\pm$1.61)
DNSD (Chang et al., 2018)	46.33 ($\pm$2.37)	64.50 ($\pm$1.11)	78.44 ($\pm$0.12)	44.69 ($\pm$1.63)	64.10 ($\pm$0.74)	79.76 ($\pm$0.08)
SpoilerNet (Wan et al., 2019)	57.19 ($\pm$0.66)	70.64 ($\pm$0.44)	79.85 ($\pm$0.12)	62.86 ($\pm$0.38)	74.62 ($\pm$0.09)	83.23 ($\pm$1.63)
MVSD (Wang et al., 2023)	65.08 ($\pm$0.69)	75.42 ($\pm$0.56)	83.59 ($\pm$0.11)	69.22 ($\pm$0.61)	78.26 ($\pm$0.63)	86.37 ($\pm$0.08)
`MMoE` (Ours)	**71.24** ($\pm$0.08)	**79.61** ($\pm$0.09)	**86.00** ($\pm$0.04)	**75.04** ($\pm$0.06)	**82.23** ($\pm$0.04)	**88.58** ($\pm$0.02)



(a) Removing rate of graph edges

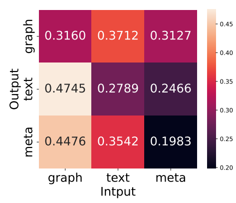


(b) Removing rate of text features

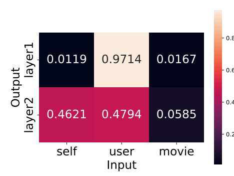


(c) Removing rate of meta features

Figure 3: `MMoE` performance when randomly removing edges in the graph, setting elements of text features to zero, and setting elements of meta features to zero. Performance slowly declines with the gradual ablations, indicating the robustness of our method.



(a) The attention score between modalities



(b) The attention score of edges in GNN

Figure 4: We first investigate the contribution of information from the three views (graph, text, and meta). We then delve into the graph neural network to find out which nodes the review nodes mainly receive information from.

2018) with user bias. MVSD (Wang et al., 2023), which introduces graph neural networks to handle user interactions, undoubtedly performs best. `MMoE` further reinforces user bias and thus achieves much better results.

4.3 Robustness Study

We verify the robustness of the model by randomly perturbing the input to simulate the absence of some information reviewed in the real situation. In specific, for graph view information, we randomly remove some of the edges in the graph; for text view and meta view information, we randomly set some of the elements to zero. The result in Figure 3 shows that, with the help of information from other modalities, even if some of the information is missing, our model still makes the correct prediction most of the time. This proves our view that multi-source information can not only improve the prediction accuracy of the model but also enhance the robustness of the model.

4.4 Multi-Modal Study

To further investigate the contribution of information from each modality, we calculate the attention score between features of different views. In specific, we extract the attention score of each layer

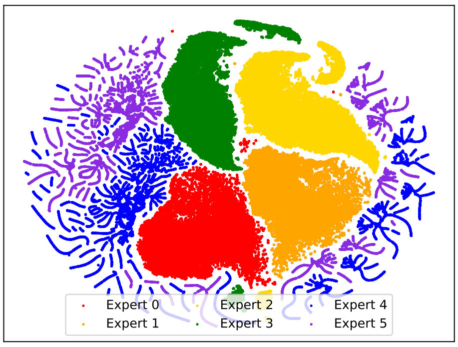


Figure 5: T-SNE visualization of reviews' graph, text, and meta features. Reviews of the same expert are represented in the same color. The reviews are clearly divided into domains based on their embedding.

in the final expert fusion transformer, and average the score of each layer. Then by averaging the values of each sample, we obtain the heat map as shown in Figure 4. Graph view features are without doubt the most contributed information, with an average attention score of 0.4127. For graph view features, we expect review nodes to receive sufficient information from user nodes and movie nodes. So we then extract the average attention scores corresponding to different types of edges in each GAT layer. "Self", "user", and "movie" represent the attention scores between review nodes and themselves, corresponding user nodes, and corresponding movie nodes in each layer respectively. It is clear that users' information is the most helpful, which also demonstrates the importance and effectiveness of our designed user profile extraction module.

4.5 Review Domain Study

We posit that due to significant variations in review styles across different types of movies, it is essential to categorize them into distinct domains and assign them to appropriate experts using the Mixture-of-Experts (MoE) approach. To validate our hypothesis, we employ T-SNE visualization (Van der Maaten and Hinton, 2008) to depict the domain assignments of reviews. We extract review representations from the MoE's output for the graph, text, and meta modalities and present them in Figure 5. The visualization clearly illustrates that reviews are distinctly segregated into different domains within each modality, which demonstrates the effectiveness of the MoE in categorizing reviews based on their representations.

Table 2: Ablation study concerning pretext task, user bias, multi-view data, MoE structure, and fusion methods.

Category	Setting	F1	AUC	Acc
Fine-tuning	w/o fine-tuning	67.45	76.99	84.48
User profile	w/o user profile	68.82	77.76	85.24
Multi-view	w/o graph view	58.29	71.09	81.55
	w/o text view	70.69	79.19	85.76
	w/o meta view	70.00	78.59	85.64
	replace GAT with R-GCN	70.34	79.03	85.51
MoE	w/o MoE	70.99	79.35	85.96
	replace MoE with MLP	71.09	79.43	85.97
	8 experts	70.96	79.40	85.84
	4 experts	70.93	79.29	85.94
Fusion	concatenate	70.02	78.48	85.82
	mean-pooling	69.84	78.31	85.82
	max-pooling	70.65	78.99	85.96
Ours	MMoE	**71.24**	**79.61**	**86.00**

4.6 Ablation Study

In order to investigate the effects of different parts of our model on performance, we conduct a series of ablation experiments on the Kaggle dataset. We report the binary F1-Score, AUC, and accuracy of the ablation study in Table 2.

Fine-tuning Strategy Study. We remove the fine-tuning step. As we can see, the performance of the model will be significantly reduced across the board. This indicates that the encoding quality of language models is very important for spoiler detection.

User Profile Study. We remove the additional user profile in our model to examine its contribution. The results show that all aspects of the model performance are reduced after removing the user profile, especially F1 and AUC.

Multi-view Study. We examine the contribution of information from different perspectives to the final result by removing information from each modality. The graph view information is the most important, which further demonstrates the significance of external information in spoiler detection. We also replace the GAT layer with other layers to observe the effects of different graph convolution operators. Interestingly, R-GCN, which is the best performer in GNN-based baselines, underperforms GAT when applied in our model. In addition, the removal of meta or text view information also has a considerable impact on the final performance, indicating the importance of the multi-view framework.

MoE Study. To investigate the contribution of

Table 3: Examples of the performance of two baselines and MMoE. Underlined parts indicate the plots. "Key Information" indicates the most helpful information from other sources when detecting spoilers.

Review Text	Key Information	Label	GAT	RoBERTa	MMoE
A loser called Brian is born on the same night as Jesus of Nazareth. He lives a parallel life with Jesus of Nazareth. He joins 'People's Front of Judea', a Jewish revolutionary party, against Romans and is confused as a messiah by (...)	**Movie Synopsis:** Brian Cohen is born in a stable a few doors down from the one in which Jesus is born, (...) His desire for Judith and hatred for the Romans lead him to join the People's Front of Judea (...)	True	False ✗	False ✗	True ✔
zzzzz. i fell asleep toward the end of this dull, lackluster Hollywood product, so i don't know if it's fair to review it but i don't feel like going back and watching the ending because i really don't care what happens to (...)	**User Profile:** Historical reviews: Reviews 1: **Warning: Spoilers** Reviews 2: **Warning: Spoilers**	True	False ✗	False ✗	True ✔
With all due respect to the original Star Wars (which is the greatest movie of all time), this is a spectacular movie, that long after you see it, you still find yourself wondering about details. (...)		False	True ✗	True ✗	False ✔
Hacksaw Ridge is an unflinching, violent assault on your senses with action sequences and people being blown apart, shot in the head, losing limbs etc etc etc which reminded me of the brutal opening scenes in (...)		False	True ✗	True ✗	False ✔

MoE, we analyze the performance changes of the model under the condition of removing the entire MoE layer, replacing MoE with MLP, and changing the number of experts. We can find from the results that the MoE layer enables the model to make a more accurate and robust prediction, which proves that it is helpful to divide reviews into different domains. We further change the number of experts to explore its impact. We use 2 experts as default, then increase the number of experts to 4 and 8. The model performance decreases in both settings, indicating that the number of experts needs to be appropriate.

Fusion Strategy Study. Finally, we study the effect of the information fusion method on performance. The results show that our self-attention-based transformer fusion method performs best in all aspects. In addition, the performance of the max-pooling method is significantly better than that of concatenation and mean-pooling.

4.7 Case Study

We conduct qualitative analysis to explore the effect of multiple source information. We select some representative cases as shown in Table 3. In the first case, the underlined part reveals the main plot of the movie. However, baseline models mainly focus on the review content itself and don't realize that it contains spoilers. With the help of information from the movie synopsis, MMoE is able to discriminate that the review is a spoiler. As for the second case, it is actually hard to identify whether the review contains spoilers. Yet through the user profile extraction module we designed, we find that the user often posts spoiler reviews. Therefore, a positive label is assigned to the sample.

5 Conclusion

We propose MMoE, a state-of-the-art spoiler detection framework which jointly leverages features from multiple modalities and adopts a domain-aware Mixture-of-Experts to handle genre-specific spoiler languages. Extensive experiments illustrate that MMoE achieves the best result among existing methods, highlighting the advantages of multi-modal information, domain-aware MoE, and user profile modeling.

Limitations and Future Work

We have considered using large language models (LLMs) to profile users based on their historical comments by generating more interpretive text features of users. However, due to the large number of users in the dataset, either calling the LLM through the API or running the open-source LLM locally takes a long time, which is one of the most difficult problems. In addition, the user descriptions generated by the LLM are not necessarily appropriate for our task. However, we still believe that there is considerable potential for using LLM for data augmentation. We can also look beyond user descriptions. Many movies lack plot synopsis. Using LLM to generate synopsis for these movies is also promising. The application of LLMs may be a key factor in subsequent breakthroughs.

Ethical Statements

Although MMoE has achieved excellent results, it still needs to be carefully applied in practice.

Firstly, there is still room for improvement in the performance of MMoE. We think it's better suited as a pre-screening tool that needs to be combined with human experts to make final decisions. Secondly, the language model encodes social bias and offensive language in the dataset (Li et al., 2022; Nadeem et al., 2020). In addition, the user profile extraction module we introduced may exacerbate this bias. We look forward to further work to detect and mitigate social bias in the spoiler detection task.

References

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Jordan Boyd-Graber, Kimberly Glasgow, and Jackie Sauter Zajac. 2013. 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.

Yixin Cao, Xiang Wang, Xiangnan He, Zikun Hu, and Tat-Seng Chua. 2019. Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences. In *The world wide web conference*, pages 151–161.

Buru Chang, Hyunjae Kim, Raehyun Kim, Deahan Kim, and Jaewoo Kang. 2018. 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*, pages 183–195. Springer.

Buru Chang, Inggeol Lee, Hyunjae Kim, and Jaewoo Kang. 2021. "killing me" is not a spoiler: Spoiler detection model using graph neural networks with dependency relation-aware attention mechanism. *arXiv preprint arXiv:2101.05972*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20:273–297.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *The Journal of Machine Learning Research*, 23(1):5232–5270.

Matthias Fey and Jan Eric Lenssen. 2019. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*.

Sheng Guo and Naren Ramakrishnan. 2010. Finding the storyteller: automatic spoiler tagging using linguistic cues. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 412–420.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Yizhi Li, Ge Zhang, Bohao Yang, Chenghua Lin, Shi Wang, Anton Ragni, and Jie Fu. 2022. Herb: Measuring hierarchical regional bias in pre-trained language models. *arXiv preprint arXiv:2211.02882*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Yuhan Liu, Zhaoxuan Tan, Heng Wang, Shangbin Feng, Qinghua Zheng, and Minnan Luo. 2023. Botmoe: Twitter bot detection with community-aware mixtures of modal-specific experts. *arXiv preprint arXiv:2304.06280*.

George Loewenstein. 1994. The psychology of curiosity: A review and reinterpretation. *Psychological bulletin*, 116(1):75.

Qingsong Lv, Ming Ding, Qiang Liu, Yuxiang Chen, Wenzheng Feng, Siming He, Chang Zhou, Jianguo Jiang, Yuxiao Dong, and Jie Tang. 2021. 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*, pages 1150–1160.

Rishabh Misra. 2019. Imdb spoiler dataset.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2020. Stereoset: Measuring stereotypical bias in pretrained language models. *arXiv preprint arXiv:2004.09456*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. 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*, pages 593–607. Springer.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, et al. 2017. Graph attention networks. *stat*, 1050(20):10–48550.

Mengting Wan, Rishabh Misra, Ndapa Nakashole, and Julian McAuley. 2019. Fine-grained spoiler detection from large-scale review corpora. *arXiv preprint arXiv:1905.13416*.

Heng Wang, Wenqian Zhang, Yuyang Bai, Zhaoxuan Tan, Shangbin Feng, Qinghua Zheng, and Minnan Luo. 2023. Detecting spoilers in movie reviews with external movie knowledge and user networks. *arXiv preprint arXiv:2304.11411*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489.

A Data Details

Table 4 and Table 5 show the metadata details of LCS and Kaggle datasets, respectively.

We further investigate the correlation between spoilers and review ratings, publication time, and length, as depicted in Figure 6. Notable patterns emerge from our investigation:

- Spoiler reviews are often poorly rated. Highly rated reviews often reveal little about the plot.
- Spoiler reviews proportion in the early and recent years is low. A large number of reviews from around 2009-2016 are filled with spoilers.
- Longer reviews are more likely to include spoilers, suggesting that the presence of spoilers increases as the length of the review expands.

We also show the proportion of spoiler reviews per user in the 2 datasets in Figure 7. It is obvious that most users are concentrated on both ends, that is, they either barely publish spoiler reviews or publish them frequently, thus have a clear tendency.

Table 4: Details of metadata contained in Kaggle.

Entity Name	Metadata
User	badge count, review count, description length
Movie	year, isAdult, runtime, rating, vote count, synopsis length
Review	time, helpful vote count, total vote count, point, content length

Table 5: Details of metadata contained in LCS.

Entity Name	Metadata
Movie	year, runtime, rating, synopsis length
Review	time, point, content length

Table 6: Hyperparameter settings of MMoE.

Hyperparameter	Language Model	User Transformer	Backbone Network
optimizer	AdamW	AdamW	AdamW
learning rate	1e-5	1e-5	1e-4
lr scheduler	WarmUpLinear	Exponential	Exponential
warm up/gamma	0.1	0.9	0.95
weight decay	1e-3	1e-5	1e-4
epochs	1	20	60
dropout	0.1	0.1	0.2
w	\	\	1e-2

B Hyperparameters

Table 6 illustrates the hyperparameter settings in the experiments. Table 7 and Table 8 demonstrate detailed model architecture parameters for easy reproduction.

Table 7: Model architecture parameters of MMoE on LCS dataset.

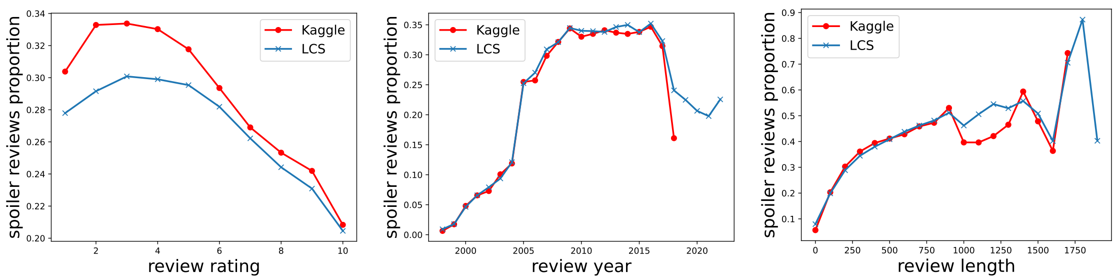
Parameters	Value
language model MLP hidden dim	3072
language model MLP out dim	768
User Transformer dim	768
User Transformer feedforward dim	3072
User Transformer number of heads	12
User Transformer layers	12
User Transformer max length	16
meta dim	6
meta MLP hidden dim	768
meta MLP out dim	256
text projection dim	256
GNN input dim	774
GNN hidden dim	512
GNN out dim	256
GNN layers	2
number of experts	2
k	1
MoE MLP hidden dim	1024
MoE MLP out dim	256
Fusion Transformer dim	256
Fusion Transformer feedforward dim	1024
Fusion Transformer number of heads	4
Fusion Transformer layers	4

C Experiment Details

- We use Neighbor Loader in Pytorch Geometric library to sample review nodes in the graph. We set the maximum number of neighbors to 200 and sample the 2-hop subgraph.
- We pad the metadata to the same dimension with -1.
- The Kaggle dataset doesn't provide the description of users. This situation further highlights the value of our user profile extraction module because it extracts user profiles from reviews. For GNN-based methods, we use zero vectors as the user's initial embedding. For our method MMoE, we set the first token of the sequence as learnable parameters, which is similar to the CLS token of BERT (Devlin et al., 2018).

D Baseline Details

We compare MMoE with PLM-based methods, GNN-based methods, and task-specific methods to ensure a holistic evaluation. For pre-trained language models, we pass the review text to the model, average all tokens, and adopt two linear projection layers to classify. For GNN-based methods, we pass the review text to RoBERTa, averaging all tokens to



(a) (b) (c)

Figure 6: (a) The spoiler proportion of reviews with different ratings; (b) The spoiler proportion of reviews posted in different years; (c) The spoiler proportion of reviews in different lengths;

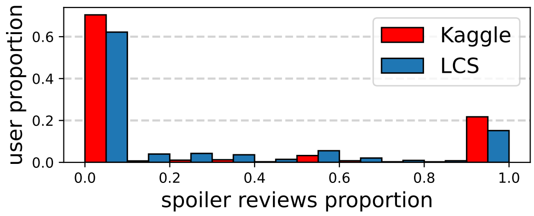


Figure 7: The proportion of spoiler reviews per user in 2 datasets, LCS and Kaggle. Spoiler review percentage intervals are divided every 10 percent.

Table 8: Model architecture parameters of MMoE on Kaggle dataset.

Parameters	**Value**
language model MLP hidden dim	3072
language model MLP out dim	768
User Transformer dim	768
User Transformer feedforward dim	3072
User Transformer number of heads	12
User Transformer layers	12
User Transformer max length	4
meta dim	4
meta MLP hidden dim	768
meta MLP out dim	256
text projection dim	256
GNN input dim	772
GNN hidden dim	512
GNN out dim	256
GNN layers	2
number of experts	4
k	1
MoE MLP hidden dim	1024
MoE MLP out dim	256
Fusion Transformer dim	256
Fusion Transformer feedforward dim	1024
Fusion Transformer number of heads	4
Fusion Transformer layers	4

get the initial node feature. We provide a brief description of each of the baseline methods, in the following.

- **BERT** (Devlin et al., 2018) is a pre-trained language model which uses masked language model and next sentence prediction tasks to train on a large amount of natural language corpus.
- **RoBERTa** (Liu et al., 2019) is an improvement model based on BERT which removes the next sentence prediction task and improves the masking strategies.
- **BART** (Lewis et al., 2019) is a pre-trained language model that improves upon traditional autoregressive models by incorporating bidirectional encoding and denoising objectives.
- **DeBERTa** (He et al., 2021) is an advanced language model that enhances BERT by introducing disentangled attention and enhanced mask decoder.
- **GCN** (Kipf and Welling, 2016) is a basic graph neural network that effectively captures and propagates information across graph-structured data by performing convolutions on the graph's nodes and their neighboring nodes.
- **R-GCN** (Schlichtkrull et al., 2018) is an extension of GCN that specifically handles multi-relational graphs by incorporating relation-specific weights.
- **GAT** (Velickovic et al., 2017) is a graph neural network that utilizes attention mechanisms to assign importance weights to neighboring nodes dynamically.
- **Simple-HGN** (Lv et al., 2021) is a graph neural network model designed for heterogeneous graphs, which effectively integrates multiple types of nodes and edges by employing a shared

embedding space and adaptive aggregation strategies.

- **DNSD** (Chang et al., 2018) is a spoiler detection method using a CNN-based genre-aware attention mechanism.

- **SpoilerNet** (Wan et al., 2019) incorporates the hierarchical attention network (HAN) (Yang et al., 2016) and the gated recurrent unit (GRU) (Cho et al., 2014) with item and user bias terms for spoiler detection.

- **MVSD** (Wang et al., 2023) utilizes external movie knowledge and user networks to detect spoilers.