Pertanika J. Sci. & Technol. 34 (4): 1981 - 2002 (2026)

PERTANIKA JOURNALS

SCIENCE & TECHNOLOGY

Journal homepage: http://www.pertanika.upm.edu.my/

Smart Comment Manager: Streamlining Comment Analysis and Viewer Query Clustering Using NLP Processing

N Venkata Sailaja, Karnam Akhil*, Sai Shashidhar Reddy Katanguri, Maringanti Gopi Krishnna, Alladi Shiva Vinay, and Devireddy Siddartha Reddy

Department of Computer Science and Engineering, VNR Vignana Jyothi Institute of Engineering and Technology, Vignana Jyothi Nagar, Bachupally, 500118 Hyderabad, Telangana, India

ABSTRACT

In Today's world, YouTube and other content creators are frequently barraged by a deluge of spam, unrelated remarks, and even hate messages, which makes it challenging for them to have a healthy and active online community. The more their audience, the more time-consuming are the traditional moderation practices and the less insight the audience opinion provides. This disconnection renders the creators incapable of knowing and redressing the problems of their community. Adding to the problem are the fuzzy logic limitations in identifying subtle topics and emotive cues, where misinterpretation can diminish the strength of content analysis. To fill these gaps, the NLP-driven Smart Comment Manager is set to simplify and enhance comment analysis. The platform is made user-friendly and removes hate speech, spam, and clusters duplicate questions, allowing creators to respond better and build a better and more engaging community. This system is built from three foundation modules: Hate Speech and Spam Detection, with RoBERTA to detect toxic content quickly and efficiently; Similar Question Clustering, with stsb-distilbert-base, to cluster repetitive questions for best responses; and Topic Discovery, with BERTopic, to detect trending topics in both comment streams and live chat. Smart Comment Manager converts comment streams into active feedback engines for building audience engagement and community, not passive feedback pipelines. Beyond content creation, the platform is also beneficial to customer support teams and online communities where handling massive amounts of user feedback is critical.

Keywords: Comment analysis, content moderation, duplicate question detection, hate speech detection, Natural Language Processing (NLP), semantic clustering, spam detection, topic modelling

ARTICLE INFO

Article history:
Received: 07 June 2025
Accepted: 02 June 2026
Published: 13 August 2026

DOI: https://doi.org/10.47836/pjst.34.4.02

E-mail addresses:
sailaja_nv@vnrvjiet.in (N Venkata Sailaja)
akhilresearch18@gmail.com (Karnam Akhil)
shashidhar7682@gmail.com (Sai Shashidhar Reddy Katanguri)
gopeekrishna36@gmail.com (Maringanti Gopi Krishnna)
shivavinayalladi@gmail.com (Alladi Shiva Vinay)
siddarthareddy.d02@gmail.com (Devireddy Siddartha Reddy)
* Corresponding author

e-ISSN: 2231-8526

INTRODUCTION

In today's online world, sites like YouTube are the hub of content creation, community, and audience interaction. With millions of comments made daily, these sites are breeding grounds for feedback, questions, and discussion. But it is a large task to handle this stream of user-generated content for a moderator.

The most surprising thing in such online forums is the presence of spam. Conventional spam filters, too often based on rigid keyword lists or blacklisting, can't hope to match spammers' constantly evolving tactics. Spam today is crafted specifically to evade such filters by using such deceptions as obfuscation, synonyms, or contextual mimicry. So, there is a need for a more sophisticated solution, one based on natural language understanding rather than shallow keyword scanning.

One of the main concerns in comment moderation is the spread of hate speech on the internet. From abusive comments to toxic content is becoming more sophisticated and difficult to identify. Today's rule-based moderation tools tend to miss the mark, either censoring innocent conversations or allowing toxic posts to pass through. But Sophisticated NLP models such as RoBERTa have demonstrated a high degree of contextual understanding. These models can make informed judgments regarding whether content is safe or not.

Another growing need in comment moderation is determining when users are asking questions. Audience engagement is a measure of content success, and creators rely on questions to gauge interest and offer more relevant content. These questions, however, become buried among general comments, and many are not in the traditional question form. Transformer models like BERT or GPT-classifiers understand sentence intent. and can identify questions even when they are informally phrased, so no meaningful interaction is lost.

Aside from identifying questions, there is also the problem of redundancy. Dozens of users will pose the same question but in slightly different wording. NLP models such as STSB-DISTIL-BERT-base are engineered specifically to overcome this. Through the conversion of sentences into vector embeddings, the system can group semantically similar questions - even ones that are worded differently.

Knowing what's going on in the larger conversation is just as valuable as responding to individual comments. Old-school topic modelling techniques such as LDA are flawed; they create unclear or confounding topics that don't reflect audience sentiment. Models such as BERTopic that use neural networks provide a closer answer by employing contextual embeddings and dynamic clustering to categorise comments by high-priority topics, enabling easy insights for more informed content decisions.

Merging all these capabilities is the Smart Comment Manager, an NLP-powered system that has the potential to transform comment management at scale. It integrates best-in-class models like RoBERTa for hate speech detection, stsb-distilbert-base for semantic clustering,

and BERTopic for trend detection. The system moderates' comments in real-time, filtering offending content automatically, clustering duplicate questions, and identifying trending topics. This solution brings order, insight, and productivity for creators.

The Smart Comment Manager enables creators and organisations to build better, safer communities. It's not merely a technical breakthrough, but a natural development of how we have conversations at scale with a correct balance between automation and comprehension of human communication.

The structure of this paper is the following: Section 2 presents an overview of related work and highlights the weaknesses of current techniques. Section 3 describes the methodology, i.e., design and implementation of the Smart Comment Manager. Section 4 presents experimental results and evaluations, and Section 5 concludes with a discussion of the main findings and future work.

Main Contributions

1. A unified transformer-based framework for real-time comment moderation and semantic audience interaction.
2. Integration of contextual spam filtering, semantic duplicate clustering, and topic discovery into a single pipeline.
3. Lightweight deployment-oriented architecture using DistilBERT and DistilRoBERTa for scalable moderation.
4. Comparative evaluation against classical ML baselines, including SVM, Logistic Regression, MNB, and ProdLDA.
5. Real-time topic summarisation for large-scale YouTube comment streams.

RELATED WORK

Ghanem and Erbay (2023) were focusing on enhancing spam detection accuracy on social media platforms from deep contextualised word representations. Their study utilised advanced word embeddings and Deep learning models, with a focus on BERT and its variants. Their results suggested BERT variants perform better than earlier state-of-the-art methods, especially in unstructured social media status message context comprehension. Nonetheless, they were computationally intensive.

Sharma et al. (2018) tried automatic classification and tagging of abusive social media content to prevent the propagation of hate messages. This research used feature extraction methods like k-skip-n-grams with Support Vector Machines (SVM), Random Forest, and Naive Bayes. Results indicated that this ensemble approach produced better accuracy but had lower accuracy on large datasets and was overwhelmed by the application of feature engineering, which did not consider the original intent or sarcasm. Building on such efforts, Song et al. (2024) extended offensive language detection to open chatbot platforms, reducing harmful interactions and improving moderation effectiveness.

Wei et al. (2021) explored transfer learning and deep learning methods in hate speech and offensive language detection on social media. They trained models using a fine-tuning approach for better accuracy on current hate speech corpora. The outcomes indicated that transfer learning, especially with DistilBERT and GPT-2, greatly improved performance. The research, although biased towards contextual meaning, still performs poorly in real-world scenarios.

Badjatiya et al. (2017) proposed a deep learning approach to hate speech detection from semantic embeddings and classification frameworks. They enhanced detection accuracy using model selection and feature engineering with pretrained word embeddings and experimented with LSTMs, GRUs, and CNNs for classification, along with Hyperparameter tuning. The results showed the strength of deep learning in extracting contextual meaning.

Zavrak and Yilmaz (2022) proposed a deep learning-based spam filtering model that is a combination of CNNs, GRUs, and attention mechanisms. The proposed model demonstrated strong capability to capture textual richness with attention layers and improved spam detection. The results showed that their hybrid approach and hierarchical attention mechanisms were more interpretable and contextually aware.

In natural language processing, question/statement classification of text is at the core of many applications. The authors Wang and Jia (2023) introduced a question type classification model with a self-attention mechanism added to a BiLSTM-AlexNet model to better capture contextual perception and improve accuracy. Their model outperformed conventional approaches in performance.

Wasim et al. (2019) addressed the problem of multi-label classification for biomedical QA systems, i.e., factoid vs. list-type question identification. Their system performed better than classical classifiers in precision and recall. Deep learning-based feature extraction assisted in enhancing classification accuracy.

Pratiwi and Syukriyah (2019) investigated how machine learning methodologies can be used to classify questions in e-learning systems. In their comparative study, they found that deep learning-based classification models outperformed algorithms like decision trees and SVMs in all environments. While the model proposed improved accuracy in question retrieval and classification, it heavily depended on the quality of the data and consumed vast computational power.

Sahu et al. (2019) tackled the issue of automatic user-posted Q&A question tagging using multi-label classification methods. The system exhibited notable improvements in tagging accuracy. Nevertheless, struggled with ambiguous or overlapping topics, and dependency on prior tagging history compromised flexibility to new topics.

Tahtah et al. (2023) created a machine learning-based question classification system that was used in the Moroccan legal context to aid in the more precise retrieval of legal information. Transformer models outperformed others in the classification process. However, Intrinsic legal language complexity and limited data became hindrances in scaling the system to more environments.

Deep Learning-Based Methods

A Critical objective of our system is identifying and collating redundant user questions, which not only improves interaction quality but also assists content creators in better answering questions.

Initial solutions to question similarity problems were based on statistical natural language processing techniques. Shah et al. (2017) proposed a deep learning model that combined LSTM networks with convolutional layers to identify redundant question pairs with boosted accuracy. They employed sentence embeddings and feature-based similarity with much superior performance.

According to the previous approach, Gupta and Joshi (2017) proposed a hybrid architecture, integrating Convolutional RNN's and CNNs to extract both syntactic and semantic properties of question similarity. And they introduced a multi-layer attention mechanism to have more precise sentence embeddings.

Likewise, Mueller and Schütze (2016) have attempted to use Siamese LSTM networks on question duplicate detection, specifically an equal-weight model that can be trained to detect more precise similarity representations with a great performance boost in paraphrased query detection and is therefore very well-suited for scaling. Building on this line of research, Korade et al. (2024) have explored new ways of NLP techniques for duplicate Question detection for maximising responses on Q&A websites.

Sentence Embedding and Transformer-Based Models for YouTube Comment Clustering

In today's world, with the growing popularity of Deep learning, Wang et al. (2017) built upon earlier deep learning techniques by employing deep sentence embeddings in combination with attention and fine-tuning on the Quora duplicate question dataset (Quora, 2017), with a focus on both similarity techniques and deeper contextualization.

Additional enhancement was suggested by Reimers and Gurevych (2019), who introduced Sentence-BERT (SBERT), specifically fine-tuned for sentence similarity tasks. It uses Siamese and triplet network architectures to generate contextualised sentence embeddings. This makes it scalable, and because of its performance, it is a go-to benchmark.

Gerlach et al. (2017) introduced a bimodal network-based model to model topic evolution over long timescales. The model was used for years of economic texts and applied to monitor the evolution of economic concepts through the analysis of the co-occurrence of topics in documents. It captured structural changes in themes over time more effectively.

Reisenbichler and Reutterer (2019) explored the potential of topic modelling to be used in consumer behaviour research and market segmentation. They showed how meaning could be uniquely extracted. Using models like Latent Dirichlet Allocation (LDA) and Non-Negative Matrix Factorisation (NMF), their work showed that high topic coherence relies on good preprocessing.

A systematic review of literature by Kherwa and Bansal (2019) analysed about 300 topic modelling studies, comparing probabilistic models like LDA with recent neural networks. The authors put a high emphasis on coherence measures to quantify results and interpretability of resulting topic structures.

Dieng et al. (2019) solved this with the Embedded Topic Model (ETM), a combination of word embeddings and traditional topic modelling to increase coherence and predictiveness. They achieved better semantic similarity on a wide range of datasets depending on pre-trained embeddings. Recent advances in Topic Modelling are carried out by Kapoor et al. (2024) by introducing QualIT, an LLM -enhanced Topic Modelling framework that provides qualitative insights.

Qiang et al. (2019) focused on Short-text topic modelling on data sources like tweets, chat logs, and product reviews. This survey addressed models like Dirichlet Multinomial Mixture (DMM), Bi-term Topic Model (BTM) on sparse textual data, which addressed the loss of topic coherence. More recently, Pham et al. (2024) came up with Topic-GPT, which leverages the power of large language models for topic discovery by using prompts, which offers great adaptability.

Grounding our work on this research basis, our research seeks to further enhance topic modelling in coherence, interpretability, and scalability. Our solution involves a blend of transformer-based models, embedding-augmented models and domain-specific optimisation techniques for short-text input, that are in coordination with the Quora question pairs dataset (Quora, 2017). With the use of deep learning methods, our solution tackles the issues of real-time deployment and domain-specific modelling.

METHODOLOGY

System Architecture

The system presented is an NLP system that is intended to moderate and analyse YouTube comments based on both machine learning and deep learning. The system is organised around four basic modules: a Comment Filtering Module, a Question Detection Module, Duplicate Question Clustering, and a Topic Modelling Module.

The process adheres to a systematic pipeline that assists in sorting and interpreting YouTube comments in an efficient manner. It all begins with real-time comment gathering, either on a regular video or live stream, with the help of the YouTube API, which enables the system to remain synchronised with the live aspect of user interaction.

Once data collection is completed, the comments are pre-processed. This phase is dedicated to text data cleaning. After the text is cleaned, tokenisation is achieved with the DistilBERT-base-uncased tokeniser. Tokenisation is a critical step in enabling more efficient processing.

Table 1
Splits of data based on train, validation and test

Module	Train	Validation	Test
Hate Speech	70	10	20
Question Detection	70	10	20
Clustering	70	10	20

Following tokenisation, the system proceeds to detect hate speech and spam for the data specified as shown in Table 1. The task of classifying is achieved by employing RoBERTa and DistilRoBERTa, which classify all comments and tag them as Normal, Hate Speech, Spam, or their combination. And they were automatically removed if they were Hate/Spam Speech.

The filtered comments are then processed by the Question Detection Module. Here, the DistilBERT-base-uncased model is used to determine whether a comment is a question or not. If a comment is determined to be a question, then it is passed on to the Duplicate Question Clustering Module for processing. And the remaining questions were sent to the Topic Modelling phase.

To efficiently process redundant questions, the system employs the STSB-BERT model, which was trained on the Quora Duplicate Question Pairs dataset (Quora, 2017). By clustering redundant & semantically similar questions in a single cluster, the system prevents repetitive responses and provides concise and complete responses.

Simultaneously, non-question comments are sent to the Topic Modelling Module. This module uses BERTopic, a powerful topic modelling platform that can pull out new topics dynamically. The result is a real-time summary of the most important topics being discussed by viewers. These can help viewers to remain attuned to the most important conversational topics.

At its last phase, the system generates three main outputs. The first is the Trending Topics. Second is the Consolidated Questions feature that groups. Moderation System, the third feature, is one that guarantees the integrity of the community is maintained and builds a more open and interactive space by eliminating hate speech as well as spam.

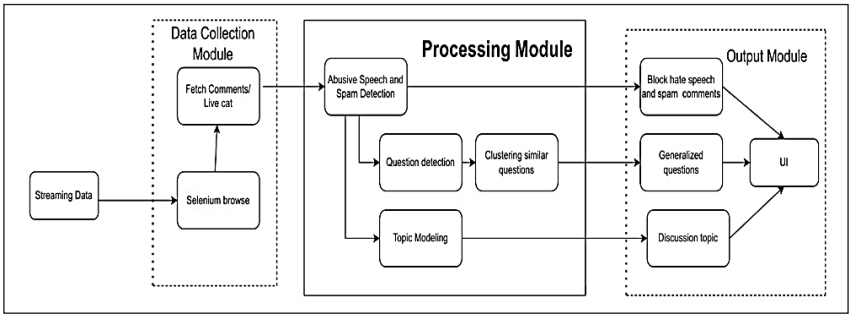

Figure 1. Generic framework for comment manager

As shown in Figure 1, the architecture of our proposed system. In general, this well-designed process keeps the pages of YouTube comments in order, informative, and convenient. It allows creators to handle audience interaction in a manner that is effective, as well as allowing for full, pertinent discussion.

ALGORITHMS

Comment Filtering Module

The Comment Filtering Module has an important function in ensuring the quality of the conversation space is kept in check by properly recognising and removing the hate and spam speeches. Since the module classifies comments by their nature, the module aids in keeping all users within a respectful, open, and interactive environment.

Models

The architecture employs DistilRoBERTa, which is a lighter yet effective variant of RoBERTa that has been optimised on hate speech datasets. Tuned for efficiency and speed, DistilRoBERTa also has good accuracy for classification, capable of detecting and eliminating harmful or toxic material to allow quality and positive user interactions.

Spam is detected by RoBERTa-base, which is a highly capable language model that has been trained on extensive text data and has been fine-tuned. It performs binary classification, labelling all comments as either Spam or Non-Spam.

Algorithm

Input. The system takes an input of comments along with pre-trained models Distil-RoBERTa (hate speech detection) and RoBERTa-base (spam detection).

Output. It generated a classification label for each comment as Normal, Hate Speech, Spam, or both.

Step 1: Load Dataset. The tweets_hate_speech_detection dataset is loaded first and split into two sets (80:20).

Step 2: Preprocessing. The tweets are tokenised with the corresponding tokeniser of each task: DistilRoBERTa for hate speech and RoBERTa for spam. Padding and truncation are applied to clean the inputs.

Step 3: Fine-tuning. Train the DistilRoBERTa model for classifying comments as hate speech or not, alongside train RoBERTa-base model to classify comments into spam or not.

Step 4: Evaluation. Compare the two models based on parameters like mean Average Precision (mAP) to evaluate their performance.

Step 5: Model Saving. Once training is complete, save the fine-tuned models and their tokenisers for later usage.

Step 6: Prediction. For each comment, run predictions through both models:

- The hate speech model returns 1 for Hate Speech or 0 if not.
- The spam model returns 1 for Spam or 0 if not.

Step 7: Final Classification.

- If both outputs are 0, then classify it as Normal Comment.
- If only the hate speech model returns 1, then classify it as Hate Speech.
- If only the spam model returns 1, then classify it as Spam.
- If both return 1, then classify as Hate Speech and Spam.

Step 8: Output. The system then returns this final classification label for the input comment.

Question Detection Module

This module is important in terms of organising discussion since it determines whether a comment from a user is indeed a question or not.

Model

The system is based on the DistilBERT-base model, a smaller and faster version of BERT, but still strong enough to provide good accuracy with rapid response times. The model is trained to distinguish actual questions and general commentary. Questions are then sent to the Duplicate Question Clustering Module, and in parallel, all comments are sent to the Topic Modelling module.

Algorithm

Input. The system takes an input of comments along with a pre-trained DistilBERT model.

Output. The model predicts and labels each comment whether it is a "Question" or "Statement".

Step 1: Dataset Loading.

- Import a labelled dataset SQuAD, which includes both questions and non-questions.
- Split the dataset into 80% - 20% ratio for training and testing.

Step 2: Preprocessing.

- A tokeniser like distil-bert-base-uncased is used for tokenising the comments.
- Truncate the inputs to ensure all are of uniform length.

Step 3: Fine-tuning. Train the DistilBERT-base model to perform binary classification—classify questions from non-questions.

Step 4: Evaluation. Test the performance of the model using standard measures like accuracy, precision, recall, and F1-score.

Step 5: Model Saving. Once training is complete, save the fine-tuned model and its tokeniser.

Step 6: Prediction.

- Tokenise any fresh comment and run it through the trained model.
- Get the output logits and use them through a softmax layer to produce probability scores.
- If the question likelihood is greater than 0.6, then it is a "Question", otherwise a "Non-Question".

Step 7: Output. The system then returns a classification whether it's a question or not.

Step 8: Routing. If a comment is labelled as a question, it's passed to the Duplicate Question Clustering Module for additional processing and in parallel, all the inputs are forwarded to the topic modelling phase.

Duplicate Question Clustering

This module is meant to group similar questions into a single category, removing redundancy. This is not only time-saving but also groups the interaction and makes it more efficient for users.

Model

The system uses STSB-BERT, a benchmark for semantic textual similarity. Unlike basic keyword matching, this model captures the underlying semantics of sentences by generating

rich, high-dimensional embeddings. It computes the similarity between two questions using cosine similarity. And if the score is greater than a certain threshold, the questions are labelled as duplicates and clustered together.

Algorithm

Input. The system begins with a list of user-submitted questions.

Output. The result is a trimmed-down collection of unique questions, where similar or repeated ones are grouped.

Step 1: Dataset Loading.

- To train the model, we used the dataset Quora Duplicate Question Pairs (Quora, 2017).
- With each question pair labelled as either a duplicate (1) or non-duplicate (0). And it is split (70:10:20) ratio for training and testing purposes.

Step 2: Data Preprocessing. All questions are tokenised to prepare them for model input.

Step 3: Model Fine-tuning. Train the STSB-BERT model to determine questions with the same intent on the Quora Dataset (Quora, 2017).

Step 4: Model Evaluation. Once training is complete, the model's accuracy is measured using metrics like mean average precision (mAP).

Step 5: Model Saving. After training, the model and its tokeniser are saved for later use.

Step 6: Duplicate Question Detection.

- Sentence embeddings are generated for all of them with the stsb-bert-base model.
- Pairwise cosine similarity values are then computed to determine how much each question is like the others.
- These are then translated into a binary similarity matrix; if two questions have a score greater than some threshold (0.75), they're labelled as duplicates and grouped. Finally, one representative question is selected in each group.

Step 7: Output. The system returns a set of separate questions.

- By effectively clustering similar questions, this module enhances user interaction, eliminates redundant answers, and maintains discussions brief and efficient.

Topic Modelling Module

This module is intended to reveal and forecast popular topics from YouTube live chats. By examining the stream of incoming comments, it determines the breaking topics in the discussion, allowing creators, moderators, and viewers to remain in sync without having to scroll through hundreds of chat messages.

Model

The system uses BERTopic, a powerful pre-trained topic modelling technique as shown in Table 2. It works by grouping similar comments into clean and understandable topics using TF-IDF and transformer-based embeddings. In word matching at the surface

Table 2
Ablation for the proposed methodology

Configuration	Accuracy
Only Spam Detection	88
Spam + Hate	90
Spam + Hate + Clustering	91
Proposed System	92

level, BERTopic identifies the underlying context and patterns in the co-occurrence of words and is therefore highly effective in dynamic environments like live chats as illustrated in Table 5. This intelligent clustering of semantically related comments provides a concise summary of the most popular topics discussed in the conversation.To ensure the fairness, Table 3 illustrates the settings of hyperparameters in detail.

Table 3
Hyperparameter settings

Model	Hyperparameters
SVM	RBF kernel, C=10, gamma=scale
Logistic Regression	C=1.0, max_iter=1000
MNB	alpha=0.5
MLP	Hidden layers = (128,64), Adam optimiser
DistilRoBERTa	lr=2e-5, batch=32, epochs=3
RoBERTa	lr=2e-5, batch=32, epochs=3
BERT-base	lr=2e-5, batch=16, epochs=4
STSB-BERT	lr=2e-5, batch=16, epochs=3
BERTopic	STSB-BERT embeddings, UMAP (15 neighbours, 5 components), HDBSCAN (cluster size=10), top 10 words/topic

Table 4
Cross-validation table

Model	Accuracy (%)	Std Dev
DistilRoBERTa	91.0	±0.4
SVM	89.1	±0.8
RoBERTa	92.2	±0.3
MNB	86.8	±1.2
MLP	88.6	±0.7
BERT-base	90.5	±0.5
STSB-BERT	94.3	±0.2
Logistic Regression	87.3	±0.9

Table 5
BERTopic stability analysis

Fold	Topic Coherence	Topic Diversity
1	0.81	0.84
2	0.83	0.86
3	0.82	0.87
4	0.84	0.85
5	0.84	0.86
Mean	0.824	0.856

Algorithm

Input. The input to this system is a list of hate and spam-filtered comments.

Output. It produces a clean, summarised list highlighting the main topics people are actively discussing.

Step 1: Loading the Pre-Trained Model. A pre-trained BERTopic model is used, which is trained on BERTopic_Wikipedia.

Step 2: Extracting Topic Information. The system gathers topic metadata using the get_topic_info method.

Step 3: Processing Input Data. Live chat comments are collected and pre-processed to prepare them for analysis.

Step 4: Applying BERTopic for Topic Modelling.

- The filtered comments are passed to the BERTopic model.
- It identifies and extracts the most relevant and frequently discussed topics.
- From there, the top two trending topics are selected based on frequency and relevance.

Step 5: Generating a Summarised Output.

- A LLM model like Gemini-Pro is used to polish the raw topics into smooth, well-phrased descriptions.
- For example, the output might say: "People in the chat are discussing Artificial Intelligence and Machine Learning, as well as Climate Change and Global Warming."

Step 6: Output. The final list of trending topics is displayed in real-time, allowing for quick understanding of the ongoing conversation.

By automating topic modelling, this module helps moderators and content creators easily follow the flow of discussion without needing to manually scroll through hundreds of live comments. Table 4 shows various model analysis on the data.

RESULTS AND DISCUSSIONS

Quora Question Pairs Dataset

Quora Question Pairs (QQP) dataset is an open-source benchmark dataset created by Quora and released in 2017 for detecting semantic similarity and duplicate questions. The dataset includes over 400,000 question pairs asked on Quora, labelled with a binary label whether the question pair is semantically similar (duplicate) or not.

Each data point contains:

- qid1, qid2: Unique identifiers for each question.
- question1, question2: What are the two questions about?
- is_duplicate: Binary indicator (1 if questions are duplicates, else 0).

In this research, we use the QQP dataset to fine-tune a sentence embedding model (STSB-BERT) for semantically similar or the same question detection from YouTube comments. Our trained model measures cosine similarity among detected questions for classifying them into sets in terms of repeated questions.

In Figure 2, the performance of DistilRoBERTa and SVM (gamma) on TF-IDF is compared for varying sample sizes for the comment filtering. The performance of both models improves with increasing sample size. DistilRoBERTa is marginally better at the beginning and consistent with all datasets, but SVM on TF-IDF takes over at larger sample sizes, more dependent on the type of task.

Figure 3 compares the accuracy of RoBERTa, Multinomial Naïve Bayes (MNB), and Multi-Layer Perceptron (MLP) when used to filter comments. RoBERTa performs best, always increasing as the samples are increased to around 92% accuracy. This is a sign that RoBERTa performs better to process comment trends.

Figure 4 shows the comparison of BERT-base uncased and Logistic Regression with TF-IDF in terms of comment analysis accuracy. BERT is always higher, and accuracy approximates nearly 91.5% as the sample size is larger. And BERT can interpret the meaning and context of the comments better, while Logistic Regression has difficulty with intricate patterns but gets better with more data.

Figure 5 interprets the performance of Logistic Regression

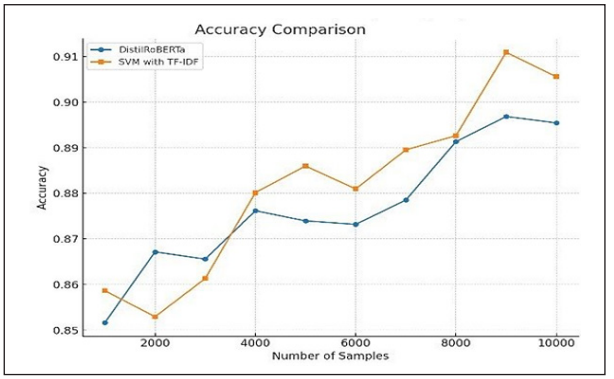

Figure 2. Accuracy Comparison of DistilRoBERTa with SVM TF-IDF

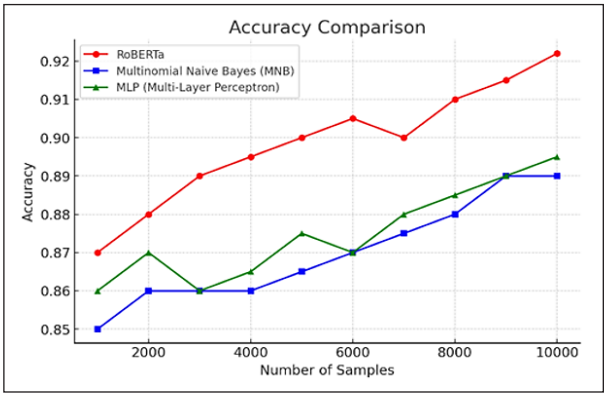

Figure 3. Accuracy Comparison between RoBERTa, MNB and MLP models

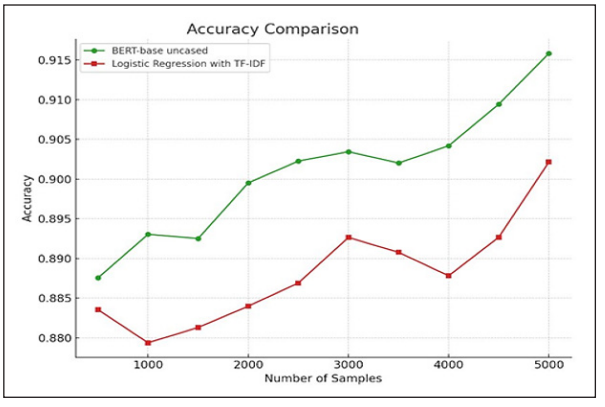

Figure 4. Accuracy comparison between BERT-base uncased and logistic regression with TF-IDF

with TF-IDF and compares it to STSB-BERT-Distil in question clustering. STSB-BERT-Distil is far ahead with a performance of approximately 94%, while Logistic Regression lags with approximately 87%.

The Intertopic Distance Map, as shown in Figure 6, displays the proximity of different topics in comments. Every bubble displays a topic, and its size displays the number of comments for that topic. This assists in comprehending the fundamental discussions in YouTube comments, clustering equivalent ones together and separating different topics from one another.

The bar chart, as shown in Figure 7, displays the most discussed subjects in YouTube comments. "Galactus" is the most discussed, followed by "Reed" and "Wonder". These subjects represent what individuals are most interested in, whether specific characters, concepts, or just general conversation.

Figure 8 shows the most relevant words in different subjects from YouTube comments. Each subject is a group of similar conversations, such as superheroes (Marvel, Superman), movie trailers, and fan reactions (love, videos).

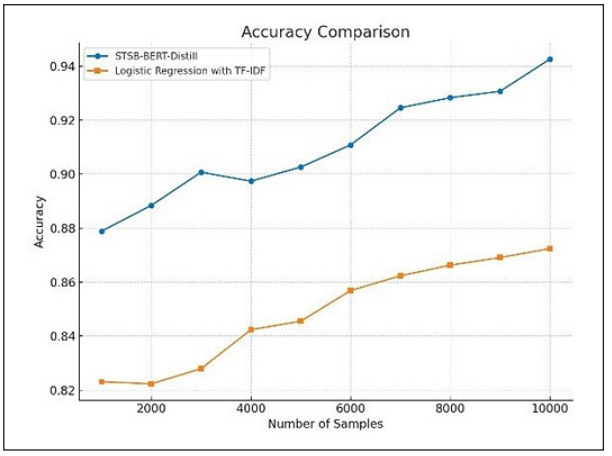

Figure 5. Accuracy comparison between STSB-BERT-distil and logistic regression with TF-IDF

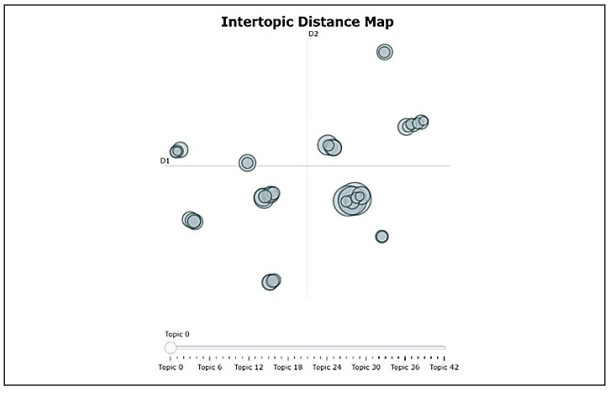

Figure 6. Intertopic distance map showing clusters of comments with similar topics

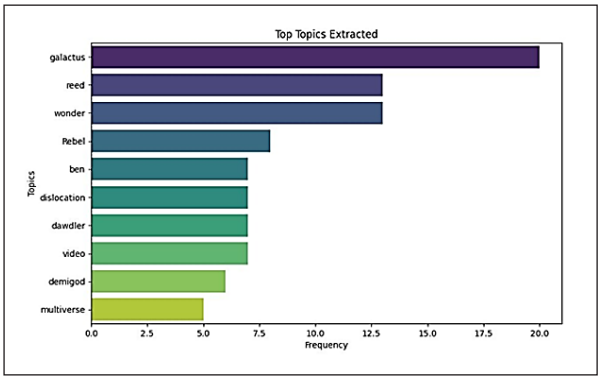

Figure 7. Bar chart displaying the most discussed topics

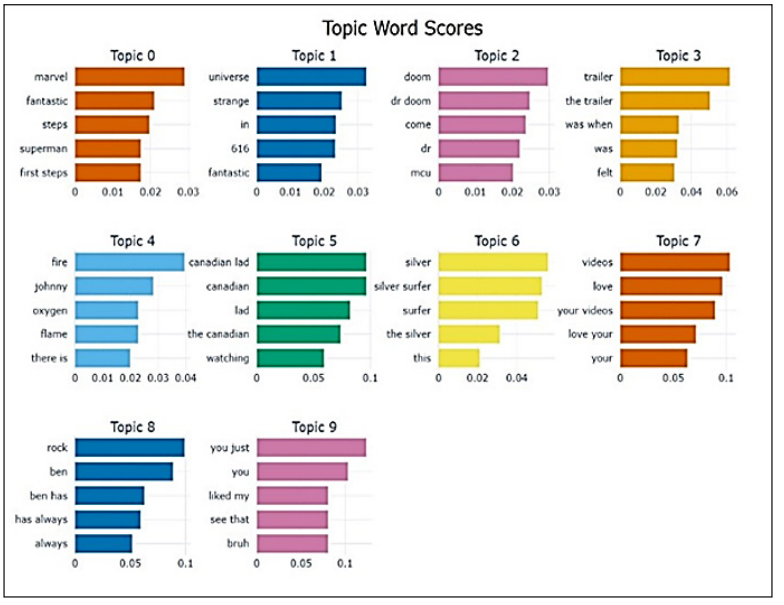

Figure 8. Most relevant words in different clusters of comments

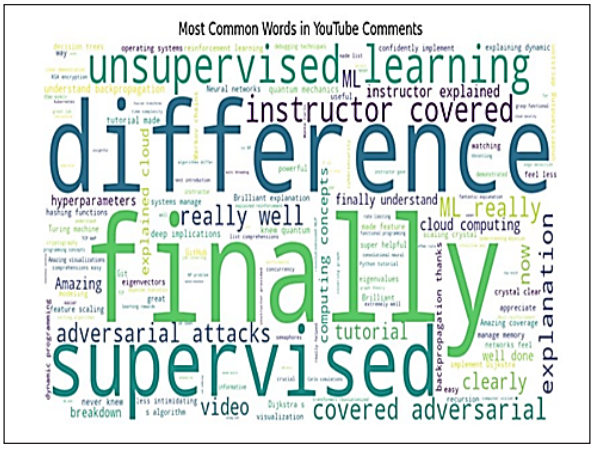

Figure 9. Word cloud of the most common words in comments

The word cloud, as shown in Figure 9, indicates the most common words used in YouTube comments, which are points of discussion. The words "difference," "supervised," "unsupervised," and "learning" are an indication that the users are interacting with learning content, especially with machine learning concepts. "Finally," and "thanks" are signs of viewers enjoying explanations.

The Perplexity score comparison is shown in Figure 10, which assists in measuring how well a model clusters YouTube comments into relevant topics. A lower perplexity

score means the model is better at detecting prevailing topics, and thus, similar comments are clustered better. BERTopic is at lower perplexity at each point in the plot compared to ProdLDA, so it is better at detecting relevant topics in real-time comment streams.

Figure 11 shows BERTopic and ProdLDA topic diversity scores against each other when they extract topics from YouTube comments. The greater the diversity score, the more types of discussion topics the model extracts, so more perspectives and subtopics are accounted for. BERTopic will have a greater diversity than ProdLDA, meaning more trending topics are extracted from real-time chat and user talks.

The Dashboard of Smart Comment Manager offers an interactive platform that acts as the main tool for analysing YouTube comments and measuring their engagement. According to Figure 12, after launching the application, the dashboard will be seen on the screen as the main point of entry, providing access to various features such as comment analysis, detecting spam and hate speech, clustering of questions, and topic modelling. The dashboard is segmented into different modules that facilitate comment analysis, identify popular topics, and cluster viewer questions. When the application initialises, these modules will be empty until a YouTube video link is provided.

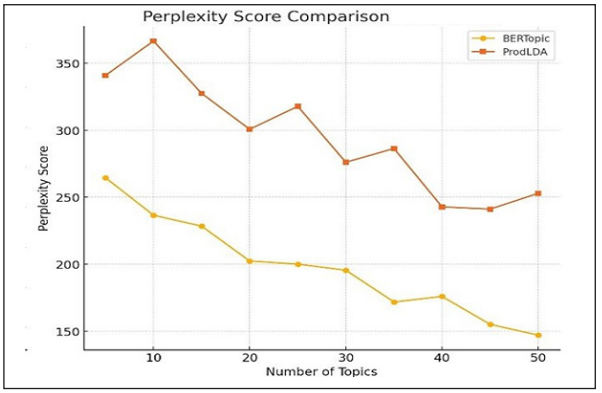

Figure 10. Perplexity score comparison between BERTopic and ProdLDA

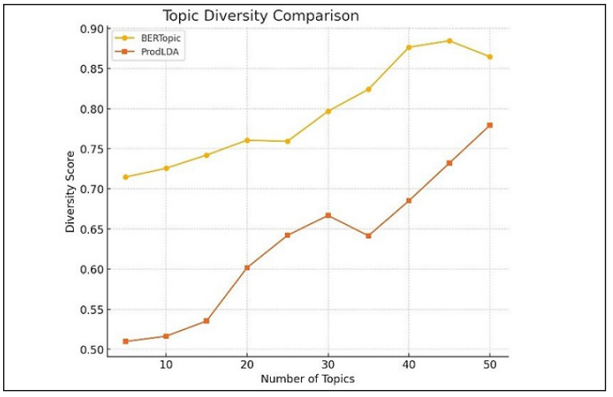

Figure 11. Topic diversity score comparison b/w BERTopic and ProdLDA

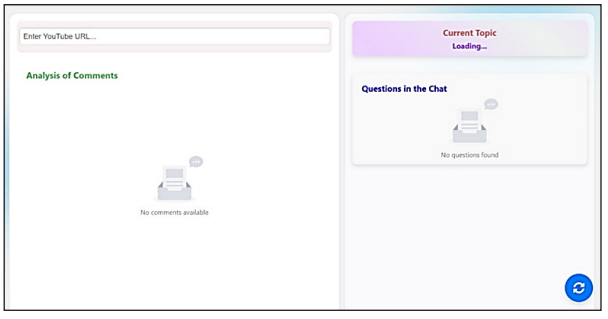

Figure 12. Initial interface when the application is launched

For an improved user experience, the Smart Comment Manager employs extensive validation procedures for the input data. Before the comments are obtained and analysed, the program validates whether the entered YouTube URL is well-formatted and supported by the system. The purpose of this process is to avoid excessive processing and the possibility that the user might enter an unsupported YouTube video URL. In case the URL is not supported, invalid, or malformed, as shown in Figure 13, the application informs the user about that using a validation message.

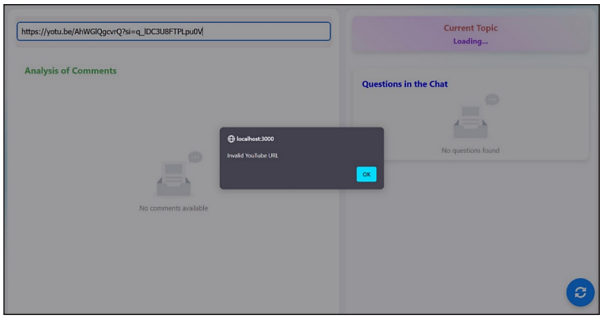

Figure 13. The system alerts the user when an invalid URL is provided

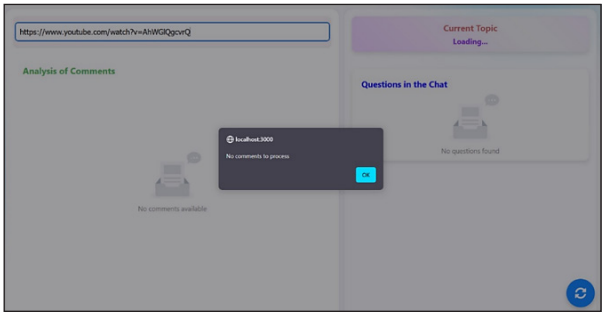

Figure 14. Application prompts when no comments are found in the processed URL

The other validation procedure involves checking whether the comments for a particular video are available or not. As shown in Figure 14, if the user enters a video that does not have any enabled comments, the program will display the validation message, thus preventing any further processes from being executed.

Upon successful validation of a YouTube video URL that has been entered into the system, shown in Figure 15, the system then analyses the comments and performs a series of actions. The dashboard provides the comments categorised, questions posted by viewers, and discussion topics raised, shown in Figures 16 and 17. Additionally, it also alerts users to comments that have anything to do with hate and spam. A side menu is provided that enables browsing of other comments for further analysis.

For the implementation of our hate speech detector model, we used Smart Comment Manager that uses the hate speech detection model based on the DistilRoBERTa transformer model. To get good classification results, it was necessary to find out the optimal model parameters. Rather than going for an automated search for hyperparameters, it was decided to go with a manual search.

The model was set up using the transformer architecture named DistilRoBERTa Base. The training was conducted on five epochs, and the batch size used was sixteen.

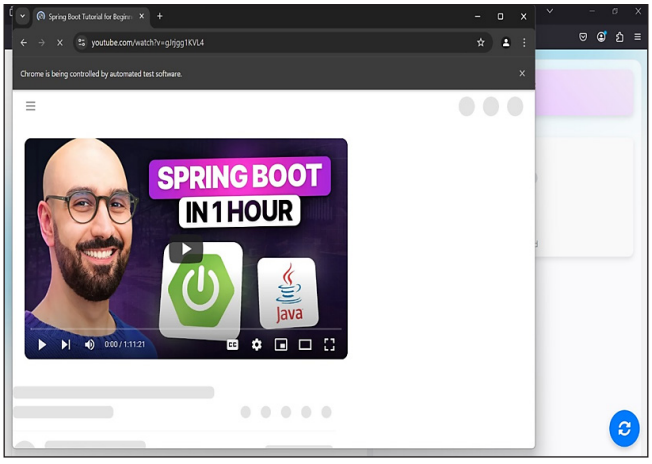

Figure 15. Comments scraping starts when a valid URL is provided

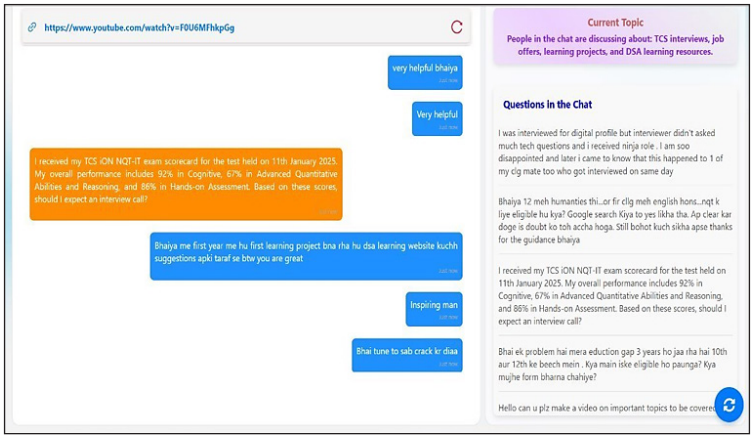

Figure 16. Displays extracted comments, trending topics, and questions in chat

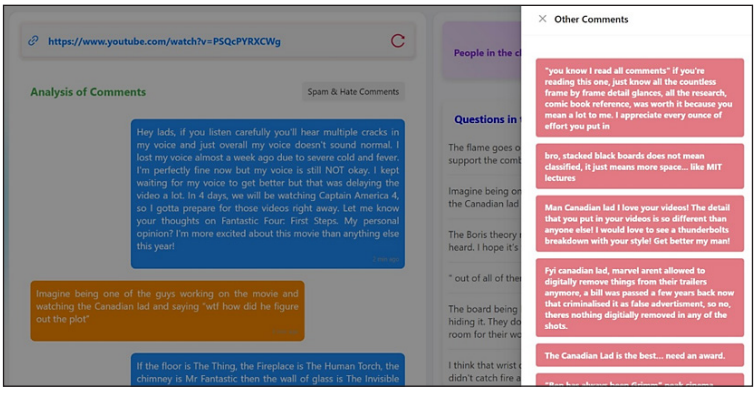

Figure 17. Displaying spam and hate comments in the side panel section for review

The classification model had two possible outputs: either Hat Speech or Non-Hate Speech. AdamW optimizer algorithm from Hugging Face Trainer was used to train the model by optimizing its weights. The default learning rate, which works well for transformers, was maintained since it demonstrated stability and reasonable performance during training.

Several combinations of training hyperparameters were tested during the experiments. It turned out that increasing the number of training epochs above five did not improve results significantly but prolonged the process considerably. Similarly, using a batch size of sixteen proved stable and allowed efficient use of available resources. Considering the achieved results, this combination of hyperparameters was found to be optimal for the given task.

The obtained model was evaluated according to standard evaluation metrics. The used hyperparameters provided a balance between achieving high accuracy and keeping the training process reasonably efficient. In view of this, they were adopted for the implementation of the Hate Speech Detector within the Smart Comment Manager system.

CONCLUSION

The Smart Comment Manager simplifies moderation of massive YouTube comment streams using advanced natural language processing techniques to detect spam and toxic comments, group similar questions, and signal trending discussion topics. It also offers insightful information about audience sentiment and interests, reducing manual moderation needs and transforming chaotic comment streams into coherent, engaging conversations. Adaptive BERT Training for improved spam and hate speech detection can be developed further. Adding multilingual NLP models and emotion detection refinement can further enhance inclusivity and audience insight, making Smart Comment Manager an all-around solution for efficient online conversation management.

Future Scope

There are a lot of interesting improvements and future work towards other applications we can implement, like Adaptive reinforcement learning for improved spam and hate speech detection, which can be developed further. Integration support for platforms like Twitter, Instagram, and Facebook, as well as API integration for social media software, can expand its applications. Adding multilingual NLP models and emotion detection refinement can further enhance inclusivity and audience insight, making Smart Comment Manager an all-around solution for efficient online conversation management.

As Natural Language Processing algorithms continue to improve, creating even more accurate question vs statement classification, similarity-based question clustering, and Topic detection should be possible, eventually resulting in clean and sophisticated content that a user can manage and respond to. We can also make a browser extension that can be enabled and utilised during a YouTube live session or a static YouTube video to provide

more accessibility and make it easier to manage. Moreover, extending the system to provide automated responses that contain brief explanations about a particular question or a topic to the users who have raised them in the comments all at once can help save time and be efficient in making more content.

ACKNOWLEDGEMENT

We sincerely acknowledge VNR Vignana Jyothi Institute of Engineering and Technology for providence of infrastructure.

LIST OF ABBREVIATIONS

BERT	:	Bidirectional encoder representations from transformers
BERTopic	:	Bidirectional encoder representations from transformers for topic modelling
CNN	:	Convolutional neural network
LDA	:	Latent Dirichlet Allocation
LLM	:	Large language model
LSTM	:	Long short-term memory
mAP	:	Mean average precision
ML	:	Machine learning
MLP	:	Multi-Layer Perceptron
MNB	:	Multinomial Naïve Bayes
NLP	:	Natural language processing
QA	:	Question answering
QQP	:	Quora question pairs
RoBERTa	:	Robustly optimised BERT pretraining approach
SBERT	:	Sentence-BERT
SQuAD	:	Stanford question answering dataset
STSB	:	Semantic textual similarity benchmark
STSB-BERT	:	Semantic textual similarity benchmark BERT
SVM	:	Support vector machine
TF-IDF	:	Term frequency–inverse document frequency
UI	:	User interface
URL	:	Uniform Resource Locator

REFERENCES

Badjatiya, P., Gupta, S., Gupta, M., & Varma, V. (2017). Deep learning for hate speech detection in tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion (WWW '17 Companion)* (pp. 759-760). International World Wide Web Conferences Steering Committee. https://doi.org/10.1145/3041021.3054223

Dieng, A. B., Ruiz, F. J. R., & Blei, D. M. (2019). Topic modelling in embedding spaces. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics* (pp. 1370-1379). PMLR. https://doi.org/10.48550/arXiv.1907.04907

Gerlach, M., Font-Clos, F., & Altmann, E. G. (2017). A bimodal network approach to model topic dynamics. *Journal of Complex Networks, 5*(6), 852-876. https://doi.org/10.1093/comnet/cnx029

Ghanem, R., & Erbay, H. (2023). Spam detection on social networks using deep contextualised word representation. *Multimedia Tools and Applications, 82*(3), 3697-3712. https://doi.org/10.1007/s11042-022-13397-8

Gupta, S., & Joshi, P. (2017). *Detecting duplicate questions with deep learning* [Technical report]. Stanford University.

Kapoor, S., Gil, A., Bhaduri, S., Mittal, A., & Mulkar, R. (2024). QualIT: LLM-enhanced topic modelling. *arXiv*. https://doi.org/10.48550/arXiv.2409.15626

Kherwa, P., & Bansal, P. (2019). Topic modelling: A comprehensive review. *EAI Endorsed Transactions on Scalable Information Systems, 7*(24), Article e2. https://doi.org/10.4108/eai.13-7-2018.159623

Korade, N. B., Salunke, M. B., Asalkar, G. G., Khedkar, R. G., Bhosale, A. U., Joshi, D. M., & Jadhav, A. C. (2024). Exploring NLP techniques for duplicate question detection to maximise responses on Q&A websites. *International Journal of Intelligent Systems and Applications in Engineering, 12*(3), 11-20.

Mueller, T., & Schütze, H. (2016). Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 2786-2797). Association for Computational Linguistics. https://doi.org/10.18653/v1/P16-1206

Pham, C. M., Hoyle, A., Sun, S., Resnik, P., & Iyyer, M. (2024). TopicGPT: A prompt-based topic modelling framework. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)* (pp. 2956-2984). Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.naacl-long.164

Pratiwi, O. N., & Syukriyah, Y. (2019). Question classification for e-learning using a machine learning approach. In the *2019 International Conference on ICT for Smart Society (ICISS)* (pp. 1-4). IEEE. https://doi.org/10.1109/ICISS48059.2019.8969811

Qiang, X., Qian, Q., & Li, Y. (2019). Short text topic modelling techniques, applications, and performance: A survey. *IEEE Transactions on Knowledge and Data Engineering, 31*(9), 1681-1706. https://doi.org/10.1109/TKDE.2018.2879325

Quora. (2017). *Quora question pairs*. Kaggle. https://www.kaggle.com/c/quora-question-pairs

Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (pp. 3982-3992). Association for Computational Linguistics. https://doi.org/10.18653/v1/D19-1410

Reisenbichler, M., & Reutterer, T. (2019). Topic modelling in marketing: Recent advances and research opportunities. *Journal of Business Economics, 89*(3), 327-356. https://doi.org/10.1007/s11573-018-0915-7

Sahu, T. P., Thummalapudi, R. S., & Nagwani, N. K. (2019). Automatic question tagging using multi-label classification in community question answering sites. In *2019 6th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud) and 2019 5th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom)* (pp. 63-68). IEEE. https://doi.org/10.1109/CSCloud/EdgeCom.2019.00017

Shah, A., Bansal, T., Venkataraman, S., & Goel, A. (2017). *Duplicate question pair detection with deep learning* (Technical Report). Stanford University.

Sharma, H. K., Kshitiz, K., & Shailendra. (2018). NLP and machine learning techniques for detecting insulting comments on social networking platforms. In the *2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)* (pp. 265-272). IEEE. https://doi.org/10.1109/ICACCE.2018.8441728

Song, H., Hong, J., Jung, C., Chin, H., Shin, M., Choi, Y., Choi, J., & Cha, M. (2024). Detecting offensive language in an open chatbot platform. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)* (pp. 4760-4771). ELRA and ICCL.

Tahtah, O., Akhiat, Y., Zinedine, A., & Fardousse, K. (2023). Towards a question-answering system in the Moroccan legal domain: Data preparation and question classification phase using ML approaches. In *2023 7th IEEE Congress on Information Science and Technology (CiSt)* (pp. 140-144). IEEE. https://doi.org/10.1109/CiSt56084.2023.10499644

Wang, S., Khabsa, M. J., & Ling, C. X. (2017). Semantic similarity for question matching in Quora. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)* (pp. 1692-1702). Association for Computational Linguistics. https://doi.org/10.18653/v1/D17-1181

Wang, Y., & Jia, Y. (2023). Question classification method based on a self-attention mechanism and BiLSTM-AlexNet model. In *2023 IEEE 29th International Conference on Parallel and Distributed Systems (ICPADS)* (pp. 1717-1724). IEEE. https://doi.org/10.1109/ICPADS60453.2023.00239

Wasim, M., Mahmood, W., Asim, M. N., & Khan, M. U. (2019). Multi-label question classification for factoid and list-type questions in biomedical question answering. *IEEE Access, 7*, 3882-3896. https://doi.org/10.1109/ACCESS.2018.2887165

Wei, Z., Ren, X., & Yu, H. (2021). Offensive language and hate speech detection with deep learning and transfer learning. *arXiv*. https://doi.org/10.48550/arXiv.2110.04627

Zavrak, S., & Yilmaz, S. (2022). Email spam detection using a hierarchical attention hybrid deep learning method. *Journal of Ambient Intelligence and Humanised Computing, 13*(6), 3467-3479. https://doi.org/10.1007/s12652-021-03549-8