# The Hallucination Problem: Predicting When Large Language Models Get It Wrong

Akanksha Anand Raut[1], Priyadarshini Ashok Kumar[2], Riya Suresh Dubey[3], Tanushree Santosh Adsule[4]
[1,2,3,4]*Mahatma Education Societys Pillai College of Arts Commerce and Science Empowered Autonomous*

***Abstract*—Large Language Models (LLMs) can generate fluent and convincing responses that contain incorrect, unsupported, or fabricated information. These outputs, commonly called hallucinations, can reduce trust in AI-assisted systems. This study investigates a machine learning approach for distinguishing correct answers from hallucinated answers using the HaluEval QA dataset. A Python notebook prepares labelled examples, applies Term Frequency–Inverse Document Frequency (TF-IDF), and trains Linear Support Vector Machine (SVM), Logistic Regression, and Multinomial Naive Bayes classifiers. The prepared data contains 20,000 examples, balanced between correct-answer and hallucinated-answer classes. A group-based 80:20 split is used to keep examples derived from the same question together. Linear SVM achieves the highest accuracy at 61.35%, followed by Logistic Regression at 60.75% and Multinomial Naive Bayes at 57.35%. These results suggest that TF-IDF features capture some useful patterns, but performance remains limited. The study concludes that lexical features alone are insufficient for dependable hallucination detection and recommends semantic representations, evidence verification, and contextual reasoning.**

***Index Terms*—Large Language Models, hallucination detection, HaluEval, TF-IDF, Linear SVM, Logistic Regression, Naive Bayes.**

## I. INTRODUCTION

Large Language Models are used for question answering, summarisation, writing assistance, and information retrieval. Their responses can be fluent and coherent while containing false or unsupported claims. This problem is commonly called hallucination. Li et al. introduced HaluEval as a benchmark for studying hallucination recognition in LLM outputs and reported that recognising hallucinated content is challenging [1].

Hallucinations include factual errors, fabricated details, contradictions with supplied context, and responses that do not follow instructions. Huang et al. survey hallucination definitions, causes, detection approaches, and mitigation strategies, distinguishing factuality-related errors from faithfulness-related errors [2].

Research has explored methods beyond simple text classification. SelfCheckGPT, for example, uses consistency across multiple sampled responses as a signal for potentially non-factual content [3]. This study instead examines a conventional supervised approach using TF-IDF features and three classifiers. Its aim is to establish a baseline comparison, not to claim that a classifier can verify every factual statement.

## II. PROBLEM STATEMENT

LLMs may produce answers that appear relevant and fluent but contain hallucinated information.

Manually checking every answer can be time-consuming. A classifier could help flag suspicious answers for further review, but it is uncertain how well traditional text-classification methods can distinguish correct from hallucinated answers using lexical features alone. This study evaluates three classifiers on HaluEval QA and compares their performance using standard metrics.

## III. BJECTIVES OF THE STUDY

The main objective is to develop and evaluate a machine learning-based approach for predicting hallucinations in LLM answers using HaluEval QA. The study prepares labelled examples, applies TF-IDF for text feature extraction, and compares Linear SVM, Logistic Regression, and Multinomial Naive Bayes. Accuracy, precision, recall, F1-score, and a confusion matrix are used to assess performance and identify limitations.

## IV. LITERATURE REVIEW

### 4.1. HaluEval Benchmark

Li et al. introduced HaluEval, a benchmark of generated and human-annotated hallucinated samples for evaluating hallucination recognition [1]. Their work indicates that LLMs can struggle to recognise hallucinations and that external knowledge or reasoning can improve recognition. This study uses the QA subset as its only dataset.

### 4.2. Taxonomy and Challenges

Huang et al. review the principles, taxonomy, challenges, and mitigation of hallucinations in LLMs [2]. Their survey discusses factuality hallucination and faithfulness hallucination. The distinction matters because an answer may conflict with real-world facts or with the supplied context. The present experiment has a narrower scope: it predicts the dataset label for an answer.

### 4.3. Self-Consistency Detection

Manakul, Liusie, and Gales proposed SelfCheckGPT, a black-box approach that uses consistency across sampled responses to help assess factuality [3]. Unlike the current supervised
TF-IDF experiment, it examines agreement between multiple responses. This research demonstrates that hallucination detection can use signals beyond word frequencies.

### 4.4. Research Gap

Existing research presents benchmarks and a range of detection strategies, including self-consistency and knowledge-supported approaches [1]–[3]. This study provides a baseline comparison of three traditional classifiers on HaluEval QA and examines the limitations of relying on lexical features.

| Reference | Main focus | Relevance |
|---|---|---|
| Li et al. (2023) [1] | HaluEval benchmark | Dataset and task context |
| Huang et al. (2025) [2] | Survey of causes, detection, and mitigation | Conceptual background |
| Manakul et al. (2023) [3] | Self-consistency detection | Contrasting detection approach |

## V. DATASET AND MODEL IMPLEMENTATION

### 5.1. Dataset Description

This study uses only the HaluEval QA dataset, available through Hugging Face. The QA subset contains 10,000 question records with fields including knowledge, question, right_answer, and hallucinated_answer. Each question is transformed into two labelled examples: one with the correct answer and one with the hallucinated answer. The prepared data therefore contains 20,000 examples, 10,000 in each class. This balanced distribution results from the experiment's preparation process and should not be interpreted as the real-world prevalence of hallucinations.

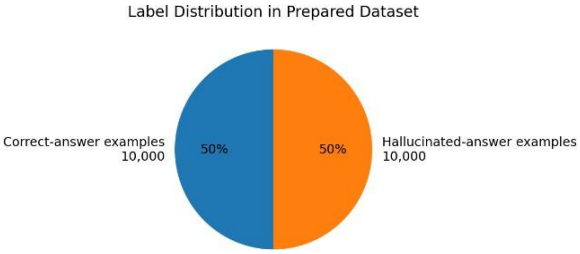



Figure 1. Label distribution in the prepared dataset.

### 5.2. Python Notebook and Classifiers

The file RESEARCH_PAPER_CLEANED (1).zip contains the Python notebook used to run the experiment. It is the implementation and experimental code, not a second dataset. The notebook loads and processes HaluEval QA, extracts features, trains classifiers, and calculates evaluation metrics.

| Component | Description |
|---|---|
| Dataset | HaluEval QA — the only dataset used |
| Prepared examples | 20,000 labelled examples:10,000 correct and 10,000 hallucinated |
| Feature extraction | TF-IDF;max_features=10,000; English stop-word removal; unigram and bigram features |
| Data split | Group-based 80:20 train/test split |
| Classifiers | Linear SVM, Logistic Regression, Multinomial Naive Bayes |
| Evaluation | Accuracy, precision, recall, F1-score, confusion matrix |

## VI. RESEARCH METHODOLOGY

The study follows a quantitative experimental design. The classifiers are trained on labelled text examples and evaluated on a held-out test set using
the same preparation and feature-extraction process.

### 6.1. Experimental Workflow




Figure 2. Workflow from loading HaluEval QA to evaluating the classifiers.

### 6.2. Data Preparation and Splitting

The data preparation process creates two labelled examples from each question. A group-based split keeps examples derived from the same original question together in either training or testing. This reduces leakage that could occur if the correct and hallucinated versions of one question were placed in different sets. The 80:20 split corresponds to 16,000 training examples and 4,000 test examples.

### 6.3. TF-IDF Feature Extraction

TF-IDF converts text into numerical features based on term frequency and inverse document frequency. The implementation uses up to 10,000 features, removes English stop words, and includes unigrams and bigrams. These features represent individual words and short word sequences, but do not directly verify claims against evidence.

### 6.4. Classification Algorithms

Linear SVM learns a decision boundary in the feature space. Logistic Regression estimates class probabilities using a linear decision function.
Multinomial Naive Bayes applies Bayes' rule with a conditional-independence assumption and is commonly used for text classification.

### 6.5. Evaluation Metrics

Accuracy is the proportion of predictions that are correct. Precision is the proportion of predicted hallucinations that are actually labelled hallucinations. Recall is the proportion of actual hallucinated examples identified by the model. F1-score is the harmonic mean of precision and recall.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$F1 = 2\frac{Precision \times Recall}{Precision + Recall}$$

TP = hallucination correctly predicted as hallucination; TN = correct answer correctly predicted as not hallucinated; FP = correct answer incorrectly flagged; FN = hallucination incorrectly missed.

## VII. DATA ANALYSIS AND RESULTS

### 7.1. Model Performance Comparison

Table 2 shows the test-set results reported by the Python notebook. Linear SVM achieves the highest accuracy and F1-score among the three tested classifiers.

| Classifier | Accuracy | Precision | Recall | F1-score |
|---|---|---|---|---|
| Linear SVM | 61.35% | 0.6261 | 0.5635 | 0.5932 |
| Logistic Regression | 60.75% | 0.6213 | 0.5505 | 0.5838 |
| Multinomial Naive Bayes | 57.35% | 0.5780 | 0.5445 | 0.5608 |

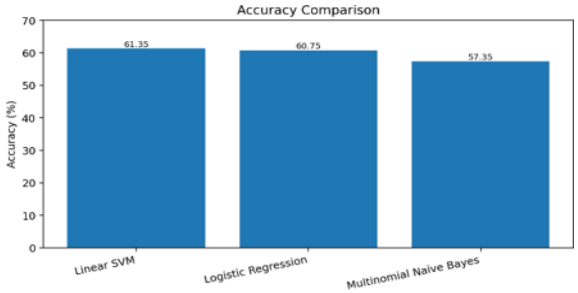



Figure 3. Accuracy comparison for the three classifiers

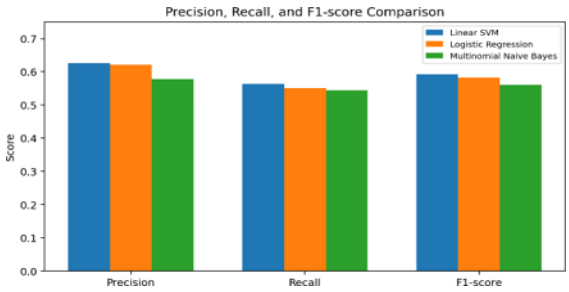



Figure 4. Precision, recall, and F1-score comparison.

Linear SVM performs best in this experiment. Its accuracy is 0.60 percentage points higher than Logistic Regression and 4.00 percentage points higher than Multinomial Naive Bayes. Because the prepared test set is balanced, an accuracy of 61.35% is only moderately above a 50% chance baseline. The results indicate that model choice affects performance, but all three classifiers have room for improvement.

### 7.2. Confusion Matrix for Linear SVM

The confusion matrix below uses hallucination as the positive class.

| Actual class | Predicted: No hallucination | Predicted: Hallucination |
|---|---|---|
| No hallucination | 1,327 | 673 |
| Hallucination | 873 | 1,127 |

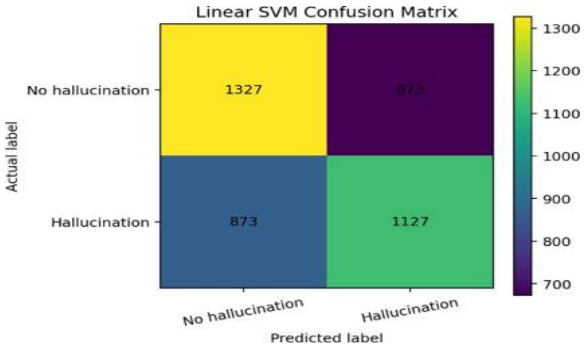



Figure 5. Confusion matrix for Linear SVM on the 4,000-example test set.

The model correctly identifies 1,127 hallucinated examples and 1,327 correct-answer examples. It misses 873 hallucinated examples and incorrectly flags 673 correct answers. These errors show that the model should be used as a screening aid, not as a definitive factuality verifier.

### 7.3. Discussion

Correct and hallucinated answers can share vocabulary and sentence structure. A hallucinated answer may sound plausible and differ from a correct answer only in a key fact. TF-IDF captures word and short-phrase frequency, but does not directly compare an answer with evidence or reason about factual consistency. The results therefore support a cautious interpretation: lexical classification can help, but alone it is not sufficient for dependable hallucination detection.

The findings fit the broader research context in which hallucination detection remains challenging and knowledge or consistency signals have been investigated [1]– [3]. This study does not directly compare its classifiers with Spellcheck or retrieval-based detectors, so it cannot establish that one method is superior to another.

## VIII. FINDINGS

HaluEval QA was the only dataset used; the Python notebook contains the experiment implementation.

The preparation procedure created 20,000 labelled examples from 10,000 QA records.

TF-IDF with unigram and bigram features was used to represent text.

Linear SVM achieved the highest accuracy (61.35%) and F1-score (0.5932).

Logistic Regression performed slightly below Linear SVM; Multinomial Naive Bayes had the lowest scores.

The confusion matrix shows both false positives and false negatives, including 873 hallucinated examples missed by Linear SVM.

Lexical features alone are not sufficient for reliable hallucination detection.

## IX. SUGGESTIONS AND RECOMMENDATIONS

Combine lexical features with semantic embeddings to represent meaning beyond word frequency.

Use evidence retrieval or knowledge-based checks to compare claims with supporting information.

Evaluate sentence-level and claim-level detection because an answer can contain both correct and incorrect statements.

Test on additional benchmarks and domains to assess generalisation.

Report precision and recall alongside accuracy, particularly when missed hallucinations and false alarms have different costs.

Keep human review for high-impact decisions; a classifier score is not proof that a claim is true or false.

Analyse false positives and false negatives to identify recurring error patterns.

## X. CONCLUSION

This study evaluated traditional machine learning classifiers for predicting hallucinated answers in LLM outputs using HaluEval QA. The Python notebook

prepared labelled examples, applied TF-IDF, and trained Linear SVM, Logistic Regression, and Multinomial Naive Bayes. Linear SVM achieved the best result, with 61.35% accuracy and an F1-score of 0.5932. However, the results remain modest and show that lexical features alone have substantial limitations. The experiment establishes a baseline comparison under a consistent evaluation setup. Future work should examine semantic representations, evidence-based verification, and additional
datasets. The findings apply to the selected HaluEval QA subset and should not be generalised to every LLM, domain, or type of hallucination.

## XI. LIMITATIONS OF THE STUDY

Only the HaluEval QA subset is used; other tasks and domains are not evaluated.
The prepared classes are balanced by construction and do not represent real-world hallucination prevalence.
TF-IDF captures lexical patterns but does not directly verify claims against external evidence.
Only three traditional classifiers are compared; neural, embedding-based, and retrieval-based detectors are outside the experiment.
Results depend on preprocessing, feature settings, the train/test split, and dataset labels.
The experiment does not explain why a specific answer is hallucinated or guarantee performance on unseen models and domains.

## REFERENCES

[1] J. Li, X. Cheng, W. X. Zhao, J.-Y. Nie, and J.-R. Wen, "HaluEval: A large-scale hallucination evaluation benchmark for large language models," in *Proc. 2023 Conf. Empirical Methods in Natural Language Processing (EMNLP)*, 2023, pp. 6449–6464, doi: 10.18653/v1/2023.emnlp-main.397. [Online]. Available: https://aclanthology.org/2023.emnlp-main.397/

[2] L. Huang *et al*., "A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions," *ACM Transactions on Information Systems*, vol. 43, no. 2, Art. no. 42, 2025, doi: 10.1145/3703155.

[3] P. Manakul, A. Liusie, and M. Gales, "SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models," in *Proc. 2023 Conf. Empirical Methods in Natural Language Processing (EMNLP)*, 2023, pp. 9004–9017, doi: 10.18653/v1/2023.emnlp-main.557. [Online]. Available: https://aclanthology.org/2023.emnlp-main.557/

[4] "HaluEval dataset," *Hugging Face Datasets*. [Online]. Available: https://huggingface.co/datasets/pminervini/HaluEval