# Embedding and Gradient Say Wrong: A White-Box Method for Hallucination Detection

**Xiaomeng Hu**[1], **Yiming Zhang**[1], **Ru Peng**[1], **Haozhe Zhang**[2], **Chenwei Wu**[2], **Gang Chen**[1], **Junbo Zhao**[1]
[1]ZheJiang University
[2]Huawei Technologies Co., Ltd, China
{xm.hu,rupeng,yimingz,cg,j.zhao}@zju.edu.cn}
zhanghaozhe7@huawei.com wucw14@mails.tsinghua.edu.cn

## Abstract

In recent years, large language models (LLMs) have achieved remarkable success in the field of natural language generation. Compared to previous small-scale models, they are capable of generating fluent output based on the provided prefix or prompt. However, one critical challenge — the *hallucination* problem — remains to be resolved. Generally, the community refers to the undetected hallucination scenario where the LLMs generate text unrelated to the input text or facts. In this study, we intend to model the distributional distance between the regular conditional output and the unconditional output, which is generated without a given input text. Based upon Taylor Expansion for this distance at the output probability space, our approach manages to leverage the embedding and first-order gradient information. The resulting approach is plug-and-play that can be easily adapted to any autoregressive LLM. On the hallucination benchmarks HADES and other datasets, our approach achieves state-of-the-art performance.

## 1 Introduction

In recent years, large language models, such as GPT-4 (OpenAI, 2023), LLaMa (Touvron et al., 2023) and PaLM (Chowdhery et al., 2022), have achieved tremendous successes. LLMs generate fluent output based on given texts and prompts. They have been applied in various real-world scenarios, such as dialogue systems and information retrieval systems (Mialon et al., 2023). However, despite their powerful generative capabilities, LLMs still suffer from the problem of hallucinations: Given an input, the model generates text unrelated to the source or even contradictory (Ji et al., 2023). Even the most advanced language model currently available, GPT-4, has been shown to experience hallucinations (Bang et al., 2023). These hallucination issues severely compromise the fairness and safety of LLMs, hindering their large-scale deployment.

Indeed, to solve this issue, the task of hallucination detection, aiming to determine whether hallucination occurs, has attracted more and more attention from the community. The existing methods for hallucination detection primarily focus on detecting at the token level in the output (Zhang et al., 2023). Some methods utilize uncertainty metrics at the token level to determine hallucination (Yuan et al., 2021; Fu et al., 2023), while some other methods design prompts to query the model multiple times to assess the hallucination (Mündler et al., 2023). **Despite the prior efforts, we believe that white-box hallucination detection is not much studied.** The white-box detection setting can both benefit the open-sourced LLM community — such as LLaMA or Mistral (Jiang et al., 2023) — and prompt the LLM owners to provide extra services upon the current forms of API. Contrary to the black-box setting, a white-box setup may utilize the full token probability matrices and the embedding layers in the transformer. In this work, we dive into this white-box setup and prove that this internal information may significantly benefit detecting hallucinations.

However, given the current parameter scale of the LLM, devising a white-box detector is non-trivial. This means we need to systematically determine what portion of the information in the LLM is discriminative to raise the hallucination case. The motivation for this study stems from a rather simple and straightforward idea, as follows. We translate the verbal definition of hallucination to be more formal. Take the QA task as an example, where $Q$ and $A$ represent the question and the model's answer, respectively. $P(A|Q)$ and $P(A)$ represent the probability distributions of the model outputting $A$ under conditional $Q$ and unconditional scenarios. The hallucination detection task can be considered as a task to determine how much the model answer $A$ depends on the question $Q$, that is, modelling the



*Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1950–1959

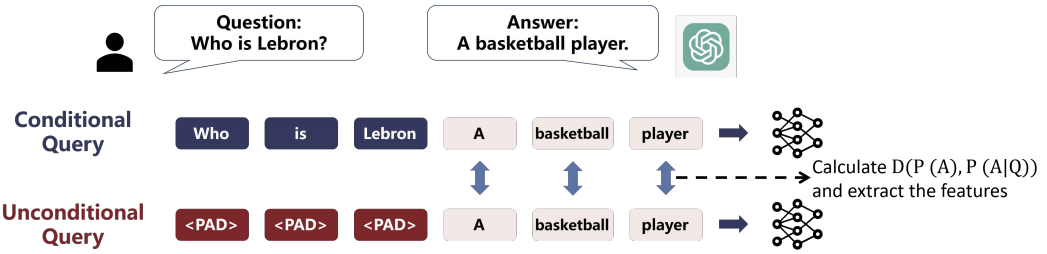



Figure 1: Two query forms of our method. Conditional query jointly inputs the question-answer text, while unconditional query only inputs the answer text. We use zero padding in the unconditional query to align the dimension.

difference between $P(A)$ and $P(A|Q)$ as denoted as $D(P(A), P(A|Q))$.

Indeed, one can directly use metrics like KL divergence, cross-entropy or others to calculate the difference between $P(A)$ and $P(A|Q)$. However, these measurements would only offer a single-dimensional output as the indicator, which we found quite imprecise towards the detection of hallucination. We argue that simply using the probability space for this problem overlooks the complexity of the hallucination issue due to the massive amount of information wasted and dropped. In our approach, by contrast, we first conduct *Taylor series* of $D(P(A), P(A|Q))$. We further show that the subitems from the resulting series can be modelled by the embeddings in the transformer combined sophistically with the first-order gradient information. This result paves the way to effectively detect hallucinations.

Figure 1 provides a visual example of our approach, named **EGH** (**E**mbedding and **G**radient-based **H**allucination detection method). More specifically, inspired by Filippova (2020), we construct a dual form, (i)-the conditional tuple jointing the source-output text versus (ii)-its unconditional counterpart with only the output text. After feeding both forms through the LLM, we yield the $D(P(A), P(A|Q))$ mentioned above. We cache the corresponding necessary information, including embedding and gradient, alongside the two feedforward passes. On top of them, we train a simple classifier to discriminate if the hallucination occurs.

Our method is a model-agnostic solution for white-box hallucination detection, which relies on model internal signals (embedding and gradient information) to perform. In order to make our method solid, we managed to experiment with multiple LLMs as base models. On several common hallucination detection datasets, including HaluEval (Li et al., 2023), SelfCheckGPT (Manakul et al., 2023), and HADES (Liu et al., 2022), EGH establishes a set of state-of-the-art performances.

## 2 Related Work

### 2.1 Hallucination in LLMs

In recent years, large language models, such as GPT-4 (OpenAI, 2023), LLaMa (Touvron et al., 2023), and PaLM (Chowdhery et al., 2022) have become the mainstream research direction in the field of natural language processing. According to the scaling law (Kaplan et al., 2020), as the number of parameters increases, the capabilities of these large language models (LLMs) also improve, allowing them to generate fluent output based on existing text. However, due to the uncertainty in the outputs, large language models (LLMs) also face hallucination issues, which significantly hinder the development of LLMs.

Ji et al. (2023) define hallucination as natural language generation models generating unfaithful or nonsensical text. In a more granular classification, we can categorize hallucinations into two types: intrinsic hallucination and extrinsic hallucination. Intrinsic hallucination refers to generated text that contradicts the input, while extrinsic hallucination refers to outputs that cannot be derived from the input, meaning unrelated to the input. While Zhang et al. (2023) categorize hallucinations into three types: Input-conflicting hallucination, where LLMs generate content that deviates from the source input provided by users; Context-conflicting hallucination, where LLMs generate content that conflicts with previously generated information by itself; Fact-conflicting hallucination, where LLMs generate content that is not faithful to established world knowledge.

Recent work indicates that hallucinations in

large language models are inevitable (Kalai and Vempala, 2023; Kang et al., 2024; Xu et al., 2024). Therefore, we need to detect hallucinations during the model's output process.

### 2.2 Hallucination Detect Method

One simple detection method utilizes numerical metrics such as ROUGE (Lin, 2004) and PARENT (Dhingra et al., 2019). Some researchers have also explored using models to detect hallucinations. FActScore (Min et al., 2023) retrieves external knowledge based on the question, and after obtaining this external knowledge, it utilizes a language model (such as LLaMa-65b) to assess the consistency between the answer and the external knowledge in order to determine the hallucination. Filippova (2020) proposed a language model-based detection method where they trained a conditional LM and an unconditional LM separately, using their respective losses to measure the presence of hallucination.

Besides, some methods allow LLMs to autonomously assess hallucinations by designing prompts to query the model multiple times. Mündler et al. (2023) design dedicated prompts to query an evaluator LLM (e.g.ChatGPT) whether the subjective LLM contradicts itself under the same context and report classification metrics, including precision, recall, and F1 score.

Existing hallucination detection methods mostly perform hallucination detection at the token level. However, when the model loses much of the internal information while outputting tokens, it undoubtedly significantly increases the difficulty of hallucination detection. Therefore, this paper designs a white-box hallucination detection method that utilizes the internal information of the model to detect hallucinations.

## 3 Method

### 3.1 Preliminary

To simplify representation, in this section, we take a question-answering task as an example for illustration. The input source text is the question in the QA task, denoted as $Q = \{Q_1, Q_2, \cdots, Q_m\}$, and the text generated by the model is the answer to the question, denoted as $A = \{A_1, A_2, \cdots, A_n\}$. Here, $m$ and $n$ represent the number of tokens of $Q$ and $A$, respectively. Our task is to classify $A$ and obtain its hallucination label $y_{hal}$, where $y_{hal} \in \{0, 1\}$.

### 3.2 Overview

Our method is based on the assumption that during the process of generating hallucinations, the model tends to incorporate less information from the source text, and the output is unrelated to the source text. Therefore, the extent to which the model accesses the source text can, to a certain degree, represent the level of hallucination. For a language model $LLM$, question $Q$, and output text $A$, we feed $Q$ and $A$ into the model in two different forms,where $[,]$ represent the concatenation operation:

- Concatenate $Q$ and $A$ together and feed into the model. This input method simulates the model's output in the presence of conditions $Q$.
- Only feed $A$ into the model. This input method corresponds to simulating the model's output without any conditions. To align with the dimension in the first case, We adopt the zero-padding method, which involves inserting $\mathbf{0}$ of length $m$ before $A$, where $\mathbf{0} = \{\overbrace{0, 0, \cdots, 0}^{m}\}$. Since the zero-padding parts are masked in the forward pass, this input method is equivalent to directly inputting A.

After completing the input in the two aforementioned forms, We have the following definition:

**Definition** ($D[Q, A]$). *the difference in output between the conditional input and the unconditional input.*

Consider the probability distributions of the answer part in two input modes, denoted as $P(A|Q)$ and $P(A|\mathbf{0})$, $D[Q, A]$ can be represented as the distribution difference between $P(A|Q)$ and $P(A|\mathbf{0})$:

$$D([Q, A]) = \text{Difference}(P(A|Q), P(A|\mathbf{0})) \quad (1)$$

where Difference() is the differentiable indicators representing differences in probability distributions, such as KL divergence, cross-entropy, and so on. During the generation of hallucinated outputs, we believe that the model receives less information from Q, resulting in a smaller relationship between the model's output and Q. The gap $D([Q, A])$ is smaller. Conversely, when the model outputs normally, it fully utilizes information from Q, leading to a larger gap $D([Q, A])$. Therefore, $D([Q, A])$ can be the extent to which the model accesses the

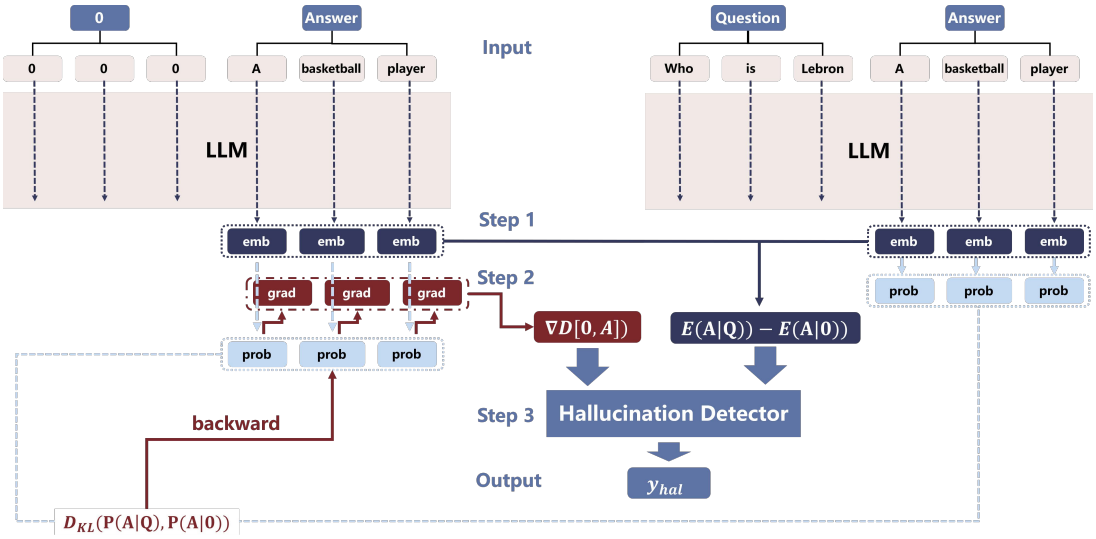



Figure 2: The algorithm schematic of the EGH. The step 1 part of the figure represents the extraction process of feature $E = E(A|Q) - E(A|\mathbf{0})$, while the step 2 part represents the extraction process of feature $G = \nabla D([\mathbf{0}, A])$. Only the step 3 part (Hallucination Detector) in the figure undergoes parameter updates during the training process.

source text and then represents the degree of hallucination further.

Since Difference() is differentiable by definition, together with the language model itself being differentiable, $D([Q, A])$ is a differentiable function. Consider the first-order Taylor polynomial of $D([Q, A])$ at point $[0, A] = \{\overbrace{0, 0, \cdots, 0}^{m}, A_1, A_2, \cdots, A_n\}$:

$$\begin{aligned} D([Q, A]) = & D([\mathbf{0}, A]) \\ & + [\nabla D([\mathbf{0}, A])]^T([Q, A] - [\mathbf{0}, A]) \\ & + R_1([Q, A]) \end{aligned} \quad (2)$$

It's clear that $D([\mathbf{0}, A])$ is a constant. $R_1([Q, A])$ is a term of higher order, such as a second derivative, with a high computational cost to model. Considering the high computational cost and difficulty of $R_1$, we ignore the use of $R_1$ in this article and only focus on whether hallucination detection can be performed based on the other terms. And the remaining two terms in Equation 2, $[Q, A] - [\mathbf{0}, A]$ and $\nabla D([\mathbf{0}, A])$, can be used as factors influecing $D([\mathbf{0}, A])$. EGH aims to identify which internal parts affect hallucination generation rather than fully modelling the entirety of $D([\mathbf{0}, A])$. Therefore, we only extract features to represent the two factors above. In the following two sections, we demonstrate the features used to represent both entities.

### 3.3 Embedding Information

For the factor $[Q, A] - [\mathbf{0}, A]$, as shown in step 1 of Figure 2, it means the difference between $[Q, A]$ and $[\mathbf{0}, A]$. When it comes to the token level, subtracting two tokens has no practical significance. Therefore, we consider using embedding layers as an alternative. $E(A|Q)$ means the embedding under condition $Q$, while $E(A|\mathbf{0})$ means the embedding under condition 0 (unconditional). The difference vectors of two embeddings can be used as representations of $[Q, A] - [0, A]$. Let $E(A|Q)$ and $E(A|\mathbf{0})$ be the embedding vectors corresponding to the answer part in two input modes, we use the differential embedding vector between $E(A|Q)$ and $E(A|\mathbf{0})$ to represent this feature, denoted as $E = E(A|Q) - E(A|\mathbf{0})$.

### 3.4 Gradient Information

For the factor $\nabla D([\mathbf{0}, A])$, as shown in step 2 of Figure 2, since we used the embedding layer above, we calculate the gradient of its corresponding layer as the representation. Consider the probability distributions of the answer part in two input modes, $D([Q, A])$ can be expressed as the sum of KL divergences corresponding to each token in the two probability distributions:

$$D([Q, A]) = \sum_{i=1}^{n} D_{KL}[P(A_i|Q)||P(A_i|\mathbf{0})] \quad (3)$$

After computing the KL divergence, we perform gradient backpropagation to obtain the gradients

**Algorithm 1** Algorithm of EGH

**Input**: The Question: $Q = \{Q_1, Q_2, \cdots, Q_m\}$, The Answer $A = \{A_1, A_2, \cdots, A_n\}$, The Function $e_{\text{LLM}}()$ is to get the final embedding of the LLM. The Function $p_{\text{LLM}}()$ is to get the Probability of the LLM's outputs. The Function $d_{\text{KL}}()$ means Calculate KL Divergence. The Weight Parameter $\lambda$.

**Output**: The hallucination label $y_{hal} \in \{0, 1\}$.

$[Q, A] \leftarrow \{Q_1, Q_2, \cdots, Q_m, A_1, A_2, \cdots, A_n\}$

$[\mathbf{0}, A] \leftarrow \{\overbrace{0, 0, \cdots, 0}^{m}, A_1, A_2, \cdots, A_n\}$

$P(A|Q) \leftarrow p_{\text{LLM}}([Q, A])$

$P(A|\mathbf{0}) \leftarrow p_{\text{LLM}}([\mathbf{0}, A])$

$D([Q, A]) \leftarrow \sum_{i=1}^{n} d_{\text{KL}}[P(A_i|Q)||P(A_i|\mathbf{0})]$

$G \leftarrow \nabla(D[\mathbf{0}, A])$

$E(A|Q) \leftarrow e_{\text{LLM}}([Q, A])$

$E(A|\mathbf{0}) \leftarrow e_{\text{LLM}}([\mathbf{0}, A])$

$E \leftarrow E(A|Q) - E(A|\mathbf{0})$

$y_{hal} \leftarrow f(\lambda E + (1 - \lambda)G)$

**return** $y_{hal}$

corresponding to the embedding layer, which is the feature that represents the factor $\nabla D([\mathbf{0}, A])$. We denote it as $G = \nabla D([\mathbf{0}, A])$

### 3.5 Embedding and Gradient-based Hallucination Detection Method

The overall algorithm is shown in Figure 2. Following the input methods described in Section 3.2, we input $Q$ and $A$ to the LLM in two different forms. For feature $E$, we take the last hidden layer as the embedding, extract them, and obtain the difference vector $E \in \mathbb{R}^{n \times h}$, where $h$ is the hidden size. For feature $G$, we first calculate the Kullback-Leibler divergence between the two output probability distributions and then perform gradient backpropagation to obtain $G \in \mathbb{R}^{n \times h}$. So far, our operations have been solely focused on obtaining the two features, and the parameters involved during this process are frozen.

After extracting the aforementioned two features, $E$ and $G$, the label of hallucination, $y_{hal} \in \{0, 1\}$, can be represented as:

$$y_{hal} = f(\alpha E + \beta G), \tag{4}$$

where $f()$ is a trainable function instantiated by a simple MLP as the end-point hallucination detector. $\alpha$, $\beta$ are hyper-parameters weighing $E$ and $G$. To simplify the representation, we introduce a variable $\lambda = \frac{\alpha}{\alpha+\beta}$ to rewrite the above formula into a more standardized form:

$$y_{hal} = f(\lambda E + (1 - \lambda)G) \tag{5}$$

In the actual implementation process, after fusing the $E$ and $G$, we take the average pooling of the outputs to obtain a representation vector $\mathbb{R}^h$ for the whole answer text. Finally, we simply use a three-layer MLP model as f to obtain the final hallucination label, $y_{\text{hal}} \in \{0, 1\}$. We use the training data to train the parameters of this part and ultimately obtain the hallucination detector.

## 4 Experiment

### 4.1 Dataset

**HaluEval (Li et al., 2023)** HaluEval is one of the benchmarks for hallucination detection. It consists of 5000 general examples and 30000 task-specific examples, covering three tasks: question answering, knowledge-grounded dialogue, and text summarization. For general examples, the questions come from the Alpaca (Taori et al., 2023) instruction fine-tuning dataset, where each data consists of ChatGPT's response and hallucination labels. For task-specific examples, each data includes a question, ChatGPT's generated correct answer, and a hallucinated answer.

**SelfcheckGPT (Manakul et al., 2023)** The dataset contains 1908 sentences from 298 articles generated by GPT-3. Each sentence is labelled with one of the three veracity labels: "Accurate", "Minor Inaccurate" and "Major Inaccurate". In the experiments, "Major Inaccurate" and "Minor Inaccurate" sentences are labelled as hallucination class, while sentences labelled as "Accurate" are considered non-hallucination class.

**HADES (Liu et al., 2022)**. HADES is a token-level hallucination detection dataset consisting of 8,754 training and 1,000 validation examples. The dataset annotates certain specific tokens in a sentence to determine whether these labelled tokens contain hallucinations.

### 4.2 Experiment Setup

**Selection of LLM** Ideally, EGH should detect during the hallucination generation process, directly utilizing a model that generates hallucinations. However, since we cannot reproduce the generation process of these datasets, we opt to use other language models to simulate the generation process. This language model must be open source

| Method | Model | QA | Dialogue | Summary | General |
|---|---|---|---|---|---|
| Baseline | GPT-3 | 49.71 | 50.42 | 51.03 | 77.47 |
| | text-davinci-002 | 59.56 | 61.23 | 47.77 | 48.23 |
| | text-davinci-003 | 49.65 | 68.37 | 48.10 | **87.05** |
| | ChatGPT | 63.03 | 67.82 | 61.74 | 86.01 |
| CoNLI (Lei et al., 2023) | GPT-3.5 | 85.00 | - | 64.10 | - |
| | GPT-4 | 86.20 | - | 72.50 | - |
| KnowHalu (Zhang et al., 2024) | GPT-3.5 | 80.70 | - | 68.50 | - |
| SAPLMA (Azaria and Mitchell, 2023) | LLaMa2-7B | 95.31 | 73.59 | 93.68 | 79.23 |
| **EGH** | LLaMa2-7B | **97.19** | **77.10** | **95.08** | 83.01 |
| | OPT-6.7B | 96.26 | 74.96 | 92.23 | 81.46 |

Table 1: Results in HaluEval Benchmark. The baseline method utilizes prompts to query the model, prompting it to determine whether the answer exhibits hallucinations. We reproduced the experiments in the baseline on our partitioned test set. EGH has achieved state-of-the-art results in three task-specific tasks.

| Method | Setup | PR-AUC |
|---|---|---|
| SelfCheckGPT | w/BERTScore | 81.96 |
| | w/QA | 84.26 |
| | w/Unigram(max) | 85.63 |
| | Combination | 87.33 |
| Wang et al. (2023) | trained on SelfCheckGPT | 86.45 |
| **EGH** | trained on HaluEval | 87.23 |
| | trained on SelfCheckGPT | **89.82** |

Table 2: Results on the SelfCheckGPT dataset. We train two models on two different datasets, 10% of the SelfCheckGPT and HaluEval. The latter results demonstrate that our method possesses a certain degree of generalization.

since we need to leverage white-box information. We chose LLaMa-2-7B and OPT-6.7B as base language models for the feature extraction step. For the HADES dataset, to align the settings in the baselines, we use BERT, RoBERTa and GPT-2 as the base model, which are the same as baselines.

**Training Process** For the HaluEval dataset, due to there being four tasks in HaluEval (for uniform representation, we also consider the general examples as a task), we trained a separate model for each task. For a specific task, We randomly select 10% of the data as the training set to train the hallucination detector and use the remaining 90% of the data as the testing set for testing. For the SelfCheckGPT dataset, we conducted separate tests with and without training: the untrained test utilized a model trained on HaluEval for evaluation, while the trained test involved training on 10% of the data from SelfCheckGPT and testing on the remaining 90% of the data. For the HADES dataset, we use the training data of HADES to obtain the hallucination detector model and test it on the validation set.

**Baselines** For the HaluEval dataset, we adopt the LLM test methods from the original paper as the baselines. The baseline experiments utilized the following prompts: *You should try your best to determine if the answer contains non-factual or hallucinated information according to the above hallucination types. The answer you give MUST be "Yes" or "No".* We evaluate four state-of-the-art LLMs as baselines: GPT-3(davinci), text-davinci-002, text-davinci-003 and ChatGPT. We reproduced the experiments in the baseline on our partitioned test set (as described above) as the final baseline result. We also compare the results in Zhang et al. (2024) and Lei et al. (2023). Additionally, since our method is a white-box method, we used the method from Azaria and Mitchell (2023) as a white-box baseline for comparison with our approach. For the SelfCheckGPT Dataset, we use the results from the original paper as the baseline. We also compare the results in Wang et al. (2023). For the HADES dataset, since the HADES dataset does not have a publicly available labelled test set, we replicated the benchmark experiments on the validation set, which serves as the baseline.

**Training Hyperparameters** Our hallucination detector is a three-layer MLP, where the first two layers scale the feature dimension to half of the preceding layer, and the final layer transforms it into probabilities. The ReLU (Glorot et al., 2011) function is chosen as the activation function. In Equation 5, $\lambda$ is set to 0.8, respectively. In Section

| Method | Model | Acc | G-Mean (↑) | BSS (↓) | AUC | Not Hallucination | | | Hallucination | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | Precision | Recall | F1 | Precision | Recall | F1 |
| Benchmark | GPT-2 | 70.60 | 69.66 | 19.68 | 75.86 | 68.16 | 80.11 | 73.66 | 74.31 | 60.57 | 66.74 |
| | BERT | 71.20 | 70.09 | 20.61 | 75.56 | 68.35 | 81.67 | 74.42 | 75.71 | 60.16 | 67.05 |
| | RoBERTa | 71.00 | 69.78 | 19.46 | 77.39 | 68.01 | **82.06** | 74.38 | **75.85** | 59.34 | 66.59 |
| **EGH** | GPT-2 | 71.30 | 70.16 | 19.34 | 75.87 | 70.35 | 75.44 | 72.81 | 71.49 | 65.88 | 68.56 |
| | BERT | **73.00** | **72.69** | **18.86** | **78.84** | **73.62** | 72.91 | 73.26 | 71.75 | **72.48** | **72.11** |
| | RoBERTa | 72.60 | 71.77 | 19.24 | 77.62 | 70.18 | 80.51 | **74.99** | 75.19 | 63.22 | 68.68 |

Table 3: We utilize the same base model to obtain gradient information and compare the results of our method with the benchmark. Our approach consistently outperforms the benchmark in terms of performance.

5.1, we provide a detailed discussion of the impact of $\lambda$ on the results.

### 4.3 Results in HaluEval

Table 1 presents the results of our experiments on HaluEval. EGH has achieved state-of-the-art (SOTA) results in three task-specific tasks. In the QA task, EGH achieved a 97% accuracy rate, which is 35% higher than the baseline and 10% higher than the CONLI method. In the Dialogue task, EGH achieved a 77.1% accuracy rate, which is 5% higher than the baseline. In the Summary task, EGH achieved a 95% accuracy rate, which is 35% higher than the baseline. In the general question-answering task, our approach achieved an accuracy of 82%, slightly lower than the baseline. Due to the imbalance of labels in the general question-answering task, with only 815 out of 5000 data instances being hallucination data, we consider this result to be a normal phenomenon. Compared with the white-box method SAPLMA, EGH has a 2-3% improvement on all four tasks.

### 4.4 Results in SelfCheckGPT

Table 2 presents the results of our experiments on SelfCheckGPT. The model trained on the HaluEval achieved performance comparable to the baseline and method Wang et al. (2023) on PR-AUC. After training on 10% of the data, our approach outperformed the baseline and method Wang et al. (2023) by 2-3% on PR-AUC. The results indicate that our method exhibits a certain degree of generalization and is applicable to various tasks and datasets.

### 4.5 Results in HADES

Table 3 presents the results of our experiments on HADES. EGH has shown improvements in accuracy compared to the benchmark, using the same base model. BERT achieved the highest accuracy, reaching 73%, and it also had the highest F1 score, surpassing the benchmark by 5%. RoBERTa achieved the highest F1 score on non-hallucination data. The experimental results indicate that EGH is also applicable to traditional language models such as BERT and GPT-2.

| $\lambda$ | 0.0 | 0.2 | 0.5 | 0.8 | 1.0 |
|---|---|---|---|---|---|
| Accuracy | 74.22 | 75.47 | 77.12 | **77.39** | 76.54 |
| F1 | 75.36 | 76.62 | 77.74 | **78.28** | 76.68 |

Table 4: Ablation Study on Weights on dialogue task

## 5 Discussion

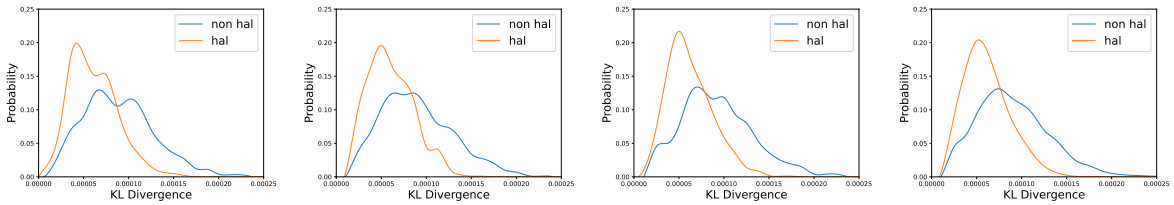
### 5.1 Weights for the embedding and Gradient

Recalling Equation 5, $y_{hal} = f(\lambda E + (1-\lambda)G)$. For the two extracted features, the difference embedding E and the gradient G, we respectively use weight $\lambda$ to perform weighted summation. To verify the effectiveness of the two features, we conduct ablation experiments using different $\lambda$, with a set of values for 0, 0.2, 0.5, 0.8, and 1. Since the accuracy on QA and summary tasks is higher, making the ablation effects less noticeable, we conducted experiments on the dialogue task instead. The experimental results are presented in the Table 4:

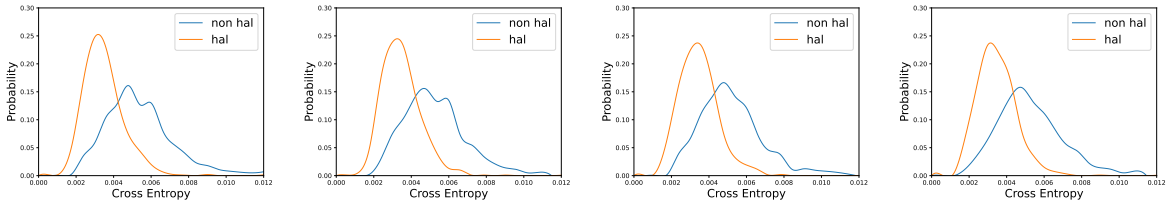
As shown in Table 4, when using only embedding or gradient ($\lambda = 0$ or $\lambda = 1$), both accuracy and F1 score are lower than when using both features simultaneously. Therefore, both features contribute to the classification process. When the value of $\lambda$ is relatively large, the classification accuracy is higher. Specifically, when $\lambda = 0.8$, the accuracy is the highest, indicating that embedding plays a primary role in classification.

### 5.2 Visualization: Difference$(P(A), P(A|Q))$

Our method is based on the assumption that during the process of generating hallucinations, the




(a) KL Divergence




(b) Cross Entropy

Figure 3: Probability Distribution for KL Divergence and Cross Entropy. There is a significant difference in the distribution of non-hallucination and hallucination samples, and the peaks also conform to our hypothesis.

| Difference Form | Metric | Hal | Non-Hal |
|---|---|---|---|
| KL Divergence | mean | $6.7 \times 10^{-5}$ | $9.6 \times 10^{-5}$ |
| | std | $2.6 \times 10^{-5}$ | $9.6 \times 10^{-5}$ |
| Cross entropy | mean | $3.7 \times 10^{-3}$ | $5.7 \times 10^{-3}$ |
| | std | $1.0 \times 10^{-3}$ | $1.8 \times 10^{-3}$ |

Table 5: Average(mean) and standard deviations(std) of KL divergence and cross entropy

| Method | QA | Summary |
|---|---|---|
| LR with KL and CE | 67.71 | 63.00 |
| EGH | 97.19 | 95.08 |

Table 6: The results of directly comparing KL divergence and cross-entropy using logistic regression and EGH indicate that applying these two features directly for hallucination detection is not advisable.

model tends to rely less on information from the input and engage in more "free expression". Due to these two input forms representing conditional and unconditional scenarios, respectively, the extent of the model's creativity, namely its access to source conditions, can be indicated by the difference between these two outputs. Therefore, we introduce Difference$(P(A), P(A|Q))$ to represent the difference between the output probability distributions under two input scenarios. Mathematical quantities such as KL divergence, cross-entropy, etc., can directly express the difference between two probability distributions, thus serving as a form of Difference$(P(A), P(A|Q))$. Consequently, we compute the values of these two statistical quantities in both hallucination and non-hallucination samples and plot the probability distribution figure.

As shown in Figure 3, we randomly sampled 1,000 examples for each task in HaluEval and obtained the probability distributions corresponding to hallucinated and non-hallucinated data. The distribution of KL divergence and Cross Entropy have certain differences between non-hallucination and hallucination samples. Since hallucinatory responses tend to receive less information from the questions, the disparity Difference$(P(A), P(A|Q))$ between the two output probability distributions will be relatively smaller. Therefore, the KL divergence and cross-entropy will be smaller for hallucination samples compared to non-hallucination samples. The mean values in the Table 5 above also confirm this observation.

However, as mentioned earlier, using a threshold value for the KL divergence or cross-entropy is imprecise to determine whether hallucinations occur. Besides, due to the loss of a significant amount of information during the process of outputting probabilities, directly utilizing features such as probabilities, KL divergence, cross-entropy, etc., cannot effectively model hallucinations. To demonstrate this, we designed a simple experiment: we used KL divergence and cross-entropy as two-dimensional features, with Logistic Regression as the classifier for detection. The results of the QA and Summary tasks in HaluEval are shown in Table 6. The experimental results support our viewpoint. Therefore, our approach extracts deeper features from these two input modes for hallucination detection.

## 6 Conclusion

This paper proposes a white-box hallucination detection method named **EGH** that utilizes the internal embedding and gradient of the model to determine hallucination. EGH is based on the following assumptions: during hallucination generation, the model tends to generate responses without considering the input question directly. The model's understanding of the input question represents the degree of hallucination. We designed both conditioned and unconditioned inputs and utilized the Taylor expansion method to demonstrate that embeddings and gradient features can represent this degree. EGH achieves state-of-the-art results on hallucination detection datasets such as HaluEval, SelfCheckGPT, and HADES, validating the effectiveness of our method. Our work provides valuable insights into detecting hallucinations in LLM.

## 7 Limitation

The main limitation of this method lies in the fact that, as our approach requires leveraging gradient information and necessitates gradient backpropagation, it undoubtedly increases the time and space complexity of the method. In terms of time, our method requires two inputs, although it can be alleviated through batch input. In terms of space, our method requires gradient calculation and additional space to store information. In the future, we will explore more lightweight methods to detect hallucinations.

## Acknowledgements

This work is supported by the Pioneer R&D Program of Zhejiang (No. 2024C01035), and the NSFC Grants (No. 62206247), and in part by the Fundamental Research Funds for the Central Universities 226-2024-00049. This work is also partially supported by ZJU Kunpeng&Ascend Center of Excellence.

## References

Amos Azaria and Tom Mitchell. 2023. The internal state of an LLM knows when it's lying. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 967–976, Singapore. Association for Computational Linguistics.

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, et al. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. *arXiv preprint arXiv:2302.04023*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Bhuwan Dhingra, Manaal Faruqui, Ankur Parikh, Ming-Wei Chang, Dipanjan Das, and William W Cohen. 2019. Handling divergent reference texts when evaluating table-to-text generation. *arXiv preprint arXiv:1906.01081*.

Katja Filippova. 2020. Controlled hallucinations: Learning to generate faithfully from noisy data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 864–870.

Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. 2023. Gptscore: Evaluate as you desire. *arXiv preprint arXiv:2302.04166*.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323. JMLR Workshop and Conference Proceedings.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Adam Tauman Kalai and Santosh S Vempala. 2023. Calibrated language models must hallucinate. *arXiv preprint arXiv:2311.14648*.

Katie Kang, Eric Wallace, Claire Tomlin, Aviral Kumar, and Sergey Levine. 2024. Unfamiliar finetuning examples control how language models hallucinate. *arXiv preprint arXiv:2403.05612*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Deren Lei, Yaxi Li, Mingyu Wang, Vincent Yun, Emily Ching, Eslam Kamal, et al. 2023. Chain of natural language inference for reducing large language model ungrounded hallucinations. *arXiv preprint arXiv:2310.03951*.

Junyi Li, Xiaoxue Cheng, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2023. Halueval: A large-scale hallucination evaluation benchmark for large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6449–6464.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Tianyu Liu, Yizhe Zhang, Chris Brockett, Yi Mao, Zhifang Sui, Weizhu Chen, and William B Dolan. 2022. A token-level reference-free hallucination detection benchmark for free-form text generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6723–6737.

Potsawee Manakul, Adian Liusie, and Mark JF Gales. 2023. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *arXiv preprint arXiv:2303.08896*.

Grégoire Mialon, Roberto Dessì, Maria Lomeli, Christoforos Nalmpantis, Ram Pasunuru, Roberta Raileanu, Baptiste Rozière, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, et al. 2023. Augmented language models: a survey. *arXiv preprint arXiv:2302.07842*.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. *arXiv preprint arXiv:2305.14251*.

Niels Mündler, Jingxuan He, Slobodan Jenko, and Martin Vechev. 2023. Self-contradictory hallucinations of large language models: Evaluation, detection and mitigation. *arXiv preprint arXiv:2305.15852*.

OpenAI. 2023. Gpt-4 technical report.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Stanford alpaca: An instruction-following llama model.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Xiaohua Wang, Yuliang Yan, Longtao Huang, Xiaoqing Zheng, and Xuan-Jing Huang. 2023. Hallucination detection for generative large language models by bayesian sequential estimation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15361–15371.

Ziwei Xu, Sanjay Jain, and Mohan Kankanhalli. 2024. Hallucination is inevitable: An innate limitation of large language models. *arXiv preprint arXiv:2401.11817*.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34:27263–27277.

Jiawei Zhang, Chejian Xu, Yu Gai, Freddy Lecue, Dawn Song, and Bo Li. 2024. Knowhalu: Hallucination detection via multi-form knowledge based factual checking. *arXiv preprint arXiv:2404.02935*.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, et al. 2023. Siren's song in the ai ocean: A survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*.

## A Analysis of Our Experimental Results

Our method achieved state-of-the-art results on various datasets. Among all tasks, the QA task yielded the best performance, with an accuracy reaching 97%. In QA tasks, answers are highly dependent on the questions, and there is a certain degree of similarity between questions. Models are likely to make "heuristic errors," answering questions based on past learned knowledge and experiences without looking at the question stem. Therefore, our approach focuses on extracting the model's understanding of the question, addressing the common hallucination type in QA tasks: Input-conflicting hallucination. We have achieved excellent results in QA tasks. For dialogue tasks they typically involve longer contexts, and the questions from different samples are relatively independent. Instances of encountering the same question are less frequent, so the model is less likely to rely on "memorizing answers". Consequently, occurrences of Input-conflicting hallucinations are less common, resulting in less effective performance compared to QA tasks. In general QA tasks, our approach falls below the baseline. We believe there are two main reasons for this: First, there is an imbalance between hallucination and non-hallucination samples, with only 815 hallucination samples available. This leads to a significant bias towards non-hallucination samples in the classifier, resulting in poor classification of hallucination samples. Second, in general datasets, questions tend to be common and relatively short. Under this task, models are most likely to experience Fact-conflicting hallucination, where the output answers contain factual errors that contradict objective facts. Since our approach does not rely on external knowledge bases, its ability to detect such hallucinations is not very strong. We consider this to be a normal phenomenon.