Résumé : SELFCHECKGPT

Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models

1 Problématique

Les modèles de langage de grande taille (LLMs) comme GPT-3 génèrent des textes fluides et convaincants, mais souffrent d'un problème majeur : **les hallucinations**. Ces modèles peuvent générer des informations factuellement incorrectes avec une grande confiance, ce qui nuit à leur fiabilité.

Les approches existantes de détection d'hallucinations présentent des limitations :

— **Méthodes grey-box** : nécessitent l'accès aux distributions de probabilités (non disponible pour ChatGPT)
— **Fact-checking externe** : requièrent des bases de données externes et des modules complexes

2 Solution proposée : SelfCheckGPT

2.1 Principe fondamental

SelfCheckGPT repose sur une idée simple mais puissante :

> *Si un LLM possède des connaissances sur un concept donné, les réponses échantillonnées de manière stochastique seront similaires et contiendront des faits cohérents. En revanche, pour les faits hallucinés, les réponses échantillonnées divergeront et se contrediront.*

2.2 Caractéristiques

— **Zero-resource** : ne nécessite aucune base de données externe
— **Black-box** : applicable même sans accès aux probabilités du modèle
— **Sampling-based** : utilise uniquement des réponses échantillonnées du LLM

3 Méthodes

3.1 Approches grey-box (baselines)

Lorsque l'accès aux probabilités est disponible :

— **Probabilités des tokens** : $\text{Avg}(-\log p)$ et $\text{Max}(-\log p)$
— **Entropie** : $\text{Avg}(H)$ et $\text{Max}(H)$ où l'entropie H mesure l'incertitude de la distribution de probabilités et est définie par :

$$H_{ij} = -\sum_{\tilde{w} \in W} p_{ij}(\tilde{w}) \log p_{ij}(\tilde{w})$$

où $p_{ij}(\tilde{w})$ est la probabilité du mot $\tilde{w}$ à la position j de la phrase i, et W est l'ensemble des mots du vocabulaire. Une entropie élevée indique une distribution plate (incertitude), tandis qu'une entropie faible indique une distribution concentrée (certitude).

— **Proxy LLM** : utiliser un autre LLM (ex : LLaMA) pour approximer les probabilités

3.2 Variantes de SelfCheckGPT (black-box)

Cinq variantes pour mesurer la cohérence informationnelle entre la réponse principale R et N échantillons stochastiques $\{S_1, S_2, \ldots, S_N\}$:

1. SelfCheckGPT-BERTScore Calcule le BERTScore moyen entre chaque phrase de R et les phrases les plus similaires dans les échantillons :

$$S_{\text{BERT}}(i) = 1 - \frac{1}{N}\sum_{n=1}^{N} \max_k B(r_i, s_k^n)$$

2. SelfCheckGPT-n-gram Construit un modèle de langage n-gram à partir des échantillons et calcule :

$$S_{\text{n-gram}}^{\text{Max}}(i) = \max_j(-\log \tilde{p}_{ij})$$

Le unigram (max) s'avère particulièrement efficace.

3. SelfCheckGPT-NLI Utilise un modèle d'inférence de langage naturel (NLI) pour détecter les contradictions :

$$S_{\text{NLI}}(i) = \frac{1}{N}\sum_{n=1}^{N} P(\text{contradict}|r_i, S_n)$$

4. SelfCheckGPT-QA Cette méthode s'appuie sur le framework MQAG (Multiple-choice Question Answering Generation) pour évaluer la cohérence. L'idée est de générer des questions à choix multiples sur la réponse principale, puis de vérifier si un système de réponse indépendant donne les mêmes réponses lorsqu'il est conditionné sur les différents échantillons.

Étape 1 : Génération de questions

Pour chaque phrase r_i de la réponse R, on génère des questions avec deux générateurs complémentaires :

— G_1 génère la question q et la réponse correcte a

— G_2 génère les distracteurs $o\backslash a$ (les mauvaises réponses)

Formellement :

$$q, a \sim P_{G_1}(q, a|r_i) \quad ; \quad o\backslash a \sim P_{G_2}(o\backslash a|q, a, R)$$

où $o = \{a, o\backslash a\} = \{o_1, \ldots, o_4\}$ représente l'ensemble des options.

Étape 2 : Réponse aux questions

Le système de réponse A sélectionne les réponses pour la réponse principale et pour chaque échantillon :

$$a_R = \arg\max_k [P_A(o_k|q, R, o)]$$

$$a_{S_n} = \arg\max_k [P_A(o_k|q, S_n, o)]$$

On compare ensuite si $a_R = a_{S_n}$ pour chaque échantillon, obtenant ainsi le nombre de correspondances N_m et de non-correspondances N_n.

Filtrage par answerability (Appendix B)

Pour éliminer les questions mal formulées, on utilise un score d'answerability (capacité à répondre) :

$$\alpha = P_U(\text{answerable}|q, \text{context})$$

où $\alpha \to 0.0$ pour les questions sans réponse et $\alpha \to 1.0$ pour les questions auxquelles on peut répondre.

Score avec théorème de Bayes et soft-counting (Appendix B)

Au lieu de simplement compter les correspondances, on applique le théorème de Bayes pour calculer la probabilité que la phrase soit non-factuelle :

$$P(F|L_m, L_n) = \frac{P(L_m, L_n|F)}{P(L_m, L_n|F) + P(L_m, L_n|T)}$$

En supposant que pour une phrase fausse, la probabilité d'une correspondance est $(1 - \beta_1)$ et pour une phrase vraie est β_2, on obtient :

$$S_{\text{QA}}(i, q) = \frac{\gamma_2^{N'_n}}{\gamma_1^{N'_m} + \gamma_2^{N'_n}}$$

où :

— $\gamma_1 = \frac{\beta_2}{1-\beta_1}$ et $\gamma_2 = \frac{\beta_1}{1-\beta_2}$ (avec $\beta_1 = \beta_2 = 0.8$ empiriquement)

— $N'_m = \sum_{n \text{ t.q. } a_n \in L_m} \alpha_n$: nombre effectif de correspondances (soft-counting)

— $N'_n = \sum_{n \text{ t.q. } a_n \in L_n} \alpha_n$: nombre effectif de non-correspondances (soft-counting)

Le score final pour la phrase i est la moyenne sur toutes les questions générées :

$$S_{\text{QA}}(i) = \mathbb{E}_q[S_{\text{QA}}(i, q)]$$

Implémentation : Les systèmes utilisés sont T5-Large fine-tuné sur SQuAD et RACE pour la génération, et Longformer fine-tuné sur RACE pour répondre aux questions.

5. SelfCheckGPT-Prompt Cette approche exploite directement les capacités d'évaluation des LLMs pour mesurer la cohérence. Des travaux récents ont montré que les LLMs sont efficaces pour évaluer la cohérence informationnelle en mode zero-shot (sans entraînement spécifique).

Principe

Pour chaque phrase r_i de la réponse R et chaque échantillon S_n, on interroge un LLM pour déterminer si la phrase est supportée par l'échantillon en utilisant le prompt suivant :

```
-----------------------------------------------
Context: {sample}
Sentence: {sentence}
Is the sentence supported by the context above?
Answer Yes or No:
-----------------------------------------------
```

Conversion en score

La réponse du LLM est convertie en score numérique x_i^n selon le mapping :

— "Yes" $\to$ 0.0 (la phrase est supportée, donc factuelle)

— "No" $\to$ 1.0 (la phrase n'est pas supportée, donc potentiellement hallucinée)

— "N/A" ou autre $\to$ 0.5 (incertain)

Le score final d'inconsistance pour la phrase i est la moyenne sur tous les échantillons :

$$S_{\text{Prompt}}(i) = \frac{1}{N} \sum_{n=1}^{N} x_i^n$$

Une valeur proche de 0 indique que la phrase est consistante avec la majorité des échantillons (factuelle), tandis qu'une valeur proche de 1 indique une inconsistance (hallucination).

Observations empiriques

— GPT-3 (text-davinci-003) répond "Yes" ou "No" dans 98% des cas, rendant la méthode très fiable

— Les modèles moins capables (text-curie-001, LLaMA) échouent à effectuer cette tâche de manière cohérente

— ChatGPT (gpt-3.5-turbo) montre une légère amélioration par rapport à GPT-3

Auto-évaluation

Une découverte intéressante : GPT-3 peut efficacement évaluer son propre texte. Même avec seulement 4 échantillons, l'auto-évaluation surpasse la méthode unigram avec 20 échantillons (corrélation de Pearson : 73.11 vs 64.71).

Coûts computationnels

Cette méthode est la plus coûteuse en termes de temps et d'argent :

— Coût API (à l'époque de l'étude) : \$0.020 pour 1000 tokens (GPT-3) vs \$0.002 (ChatGPT)

— Pour évaluer 1908 phrases avec 20 échantillons : ∼\$200 (GPT-3) ou ∼\$20 (ChatGPT)

— Temps : nécessite N appels API par phrase

Comparaison des réponses GPT-3 vs ChatGPT

Sur 7632 évaluations, la matrice de confusion montre :

	ChatGPT : Yes	ChatGPT : No
GPT-3 : Yes	3179	1038
GPT-3 : No	367	3048

Accord de ∼81% entre les deux modèles, ChatGPT étant légèrement plus performant.

4 Expériences

4.1 Dataset

— **Source** : 238 passages générés par GPT-3 (text-davinci-003) sur des individus du dataset WikiBio

— **Annotation manuelle** : 1908 phrases annotées au niveau factuel

— **Labels** :

  — Major Inaccurate (39.9%) : phrase entièrement hallucinée

  — Minor Inaccurate (33.1%) : information partiellement incorrecte

  — Accurate (27.0%) : information factuelle

— **Accord inter-annotateur** : $\kappa = 0.595$ (3 classes) et $\kappa = 0.748$ (2 classes)

4.2 Configuration

— Réponse principale : température = 0.0 (beam search)

— Échantillons stochastiques : température = 1.0, $N = 20$ échantillons

— Proxy LLM : LLaMA-30B

5 Résultats

5.1 Détection au niveau des phrases (AUC-PR)

Méthode	Non-Factuel	Non-Factuel*	Factuel
Random	72.96	29.72	27.04
Grey-box (GPT-3 probabilities)			
Avg$(-\log p)$	83.21	38.89	53.97
Max$(-\log p)$	87.51	35.88	50.46
Black-box (Proxy LLM - LLaMA-30B)			
Avg(H)	80.80	39.01	42.97
SelfCheckGPT (Black-box)			
BERTScore	81.96	45.96	44.23
QA	84.26	40.06	48.14
Unigram (max)	85.63	41.04	58.47
NLI	92.50	45.17	66.08
Prompt	**93.42**	**53.19**	**67.09**

5.2 Classement au niveau des passages (Corrélation)

Méthode	Pearson	Spearman
GPT-3 Avg$(-\log p)$	57.04	53.93
LLaMA Avg(H)	33.80	39.49
SelfCk-BERTScore	58.18	55.90
SelfCk-QA	61.07	59.29
SelfCk-Unigram	64.71	64.91
SelfCk-NLI	74.14	73.78
SelfCk-Prompt	**78.32**	**78.30**

5.3 Observations clés

1. **SelfCheckGPT surpasse les méthodes grey-box** : SelfCheckGPT-Prompt obtient des performances considérablement supérieures aux méthodes nécessitant l'accès aux probabilités.
2. **Les proxy LLMs sont peu efficaces** : Utiliser un LLM différent (LLaMA) pour approximer les probabilités de GPT-3 donne des résultats nettement inférieurs, car les modèles ont des patterns de génération différents.
3. **Trade-off performance/coût** :
   — SelfCheckGPT-Prompt : meilleure performance mais coûteux computationnellement
   — SelfCheckGPT-NLI : excellent compromis performance/coût
   — SelfCheckGPT-Unigram : méthode la moins coûteuse avec de bonnes performances
4. **Impact du nombre d'échantillons** : La performance augmente avec le nombre d'échantillons (N), avec des gains décroissants au-delà de 10-15 échantillons.

6 Études d'ablation

6.1 Comparaison avec connaissances externes

Lorsqu'on compare SelfCheckGPT avec l'utilisation du paragraphe WikiBio de référence :

— SelfCheckGPT-NLI/Prompt bénéficient d'un accès aux connaissances externes

— SelfCheckGPT-BERTScore/QA performent aussi bien ou mieux avec les échantillons seuls

— SelfCheckGPT-n-gram échoue avec une seule source externe (insuffisant pour entraîner le modèle)

6.2 Choix du LLM pour SelfCheckGPT-Prompt

— GPT-3 peut évaluer son propre texte efficacement

— ChatGPT montre une légère amélioration par rapport à GPT-3

— Modèles moins capables (GPT-Curie, LLaMA) échouent à cette tâche

7 Conclusions

7.1 Contributions principales

1. **Premier travail** sur la détection d'hallucinations pour les réponses générales de LLMs
2. **Méthode zero-resource** ne nécessitant aucune base de données externe
3. **Approche black-box** applicable à n'importe quel LLM, même via API limitée
4. **Dataset annoté** : 238 passages et 1908 phrases avec labels de factualité

7.2 Points clés

— SelfCheckGPT exploite la cohérence interne des LLMs pour détecter les hallucinations

— Les méthodes proposées surpassent significativement les baselines grey-box

— SelfCheckGPT-Prompt obtient les meilleures performances (AUC-PR : 93.42, Corrélation : 78.32)

— SelfCheckGPT-NLI offre le meilleur compromis pratique

— La méthode est particulièrement efficace pour détecter les "total hallucinations"

7.3 Limitations

— Dataset principalement composé de biographies (WikiBio)

— Évaluation au niveau des phrases (une phrase peut contenir information factuelle et hallucinée)

— SelfCheckGPT-Prompt est computationnellement coûteux

— Performances variables selon la capacité du LLM utilisé

7.4 Perspectives

— Extension à d'autres types de concepts (lieux, objets)

— Évaluation à granularité plus fine (faits atomiques)

— Optimisation du coût computationnel

— Application à d'autres tâches de génération de texte

8 Impact et applications

SelfCheckGPT représente une avancée significative pour :

— Améliorer la fiabilité des systèmes basés sur les LLMs

— Permettre la détection d'hallucinations sans accès privilégié aux modèles

— Fournir une baseline solide pour de futures recherches sur ce problème crucial

— Réduire les risques de désinformation liés aux LLMs

Référence : Manakul, P., Liusie, A., & Gales, M. J. F. (2023). SELFCHECKGPT : Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models. *arXiv :2303.08896*.