HOW MANY OPINIONS DOES YOUR LLM HAVE? IMPROVING UNCERTAINTY ESTIMATION IN NLG

Lukas Aichberger[1], **Kajetan Schweighofer**[1], **Mykyta Ielanskyi**[1], **Sepp Hochreiter**[1,2]
[1] ELLIS Unit Linz and LIT AI Lab, Institute for Machine Learning,
Johannes Kepler University Linz, Austria
[2] NXAI GmbH, Linz, Austria

ABSTRACT

Large language models (LLMs) suffer from hallucination, where they generate text that is not factual. Hallucinations impede many applications of LLMs in society and industry as they make LLMs untrustworthy. It has been suggested that hallucinations result from predictive uncertainty. If an LLM is uncertain about the semantic meaning it should generate next, it is likely to start hallucinating. We introduce Semantic-Diverse Language Generation (SDLG) to quantify predictive uncertainty of LLMs. Our method detects if a generated text is hallucinated by offering a precise measure of aleatoric semantic uncertainty. Experiments demonstrate that SDLG consistently outperforms existing methods while being computationally the most efficient, setting a new standard for uncertainty estimation in NLG.

1 INTRODUCTION

Hallucinations are fragments of a generated text that, despite appearing cohesive, are not factual. They hinder a broad use of LLMs as they make them untrustworthy (Manakul et al., 2023). Hallucinations are found to mainly arise due to the predictive uncertainty inherent to probabilistic models (Xiao & Wang, 2021). While uncertainty estimation for classification has been developed extensively (Hüllermeier & Waegeman, 2021; Gawlikowski et al., 2023), uncertainty estimation for autoregressive models is still a challenging problem.

Semantic-Diverse Language Generation (SDLG) seeks to improve the efficiency of uncertainty estimation in autoregressive natural language generation (NLG) using importance sampling. Only semantically diverse output sequences should increase the *semantic uncertainty*. This involves estimating the probability of generating output sequences with divergent semantic meanings. SDLG uses a proposal distribution that samples semantically diverse output sequences. A natural language inference model is used not only to cluster generated sequences according to their semantic equivalence (Kuhn et al., 2023), but also to compute the contribution of every generated token to the final semantics. This allows for substituting the most important token to resample a sentence with a high likelihood but different semantic meaning. Such sentences are valuable summands for estimators of *aleatoric semantic uncertainty* that are not reached by multinomial sampling in practice.

2 PREDICTIVE UNCERTAINTY IN NLG

Estimating uncertainty in NLG differs from estimating uncertainty in classification tasks (see Sec. C in the appendix) in two key aspects. First, a sequence of predictions collectively forms the final output of a model. Second, different output sequences might be equivalent in their semantic meaning. To account for the latter aspect, Kuhn et al. (2023) introduce *semantic entropy*. We provide a principled derivation of this measure and discuss practical considerations of how to estimate it.

2.1 MEASURING PREDICTIVE UNCERTAINTY IN NLG

Prerequisites. Given are an autoregressive language model parametrized by $\boldsymbol{w}$ and an input sequence of tokens $\boldsymbol{x} = (x_1, ..., x_M)$ with $x \in \mathcal{V}$. An output of the model is a sequence of to-

kens $\boldsymbol{y} = (y_1, ..., y_T) \in \mathcal{Y}$ with $y \in \mathcal{V}$. The predictive distribution at step t of the output sequence $\boldsymbol{y}$ is conditioned on both the input sequence and all previously generated tokens, denoted as $p(y_t \mid \boldsymbol{x}, \boldsymbol{y}_{<t}, \boldsymbol{w})$. The probability of an output sequence is the product of the individual token probabilities: $p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w}) = \prod_{t=1}^{T} p(y_t \mid \boldsymbol{x}, \boldsymbol{y}_{<t}, \boldsymbol{w})$. In practice, $p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w})$ is often length-normalized to not favor short sequences (Cover & Thomas, 2006; Malinin & Gales, 2021; Kuhn et al., 2023), which results in $\bar{p}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w}) = \exp\left(\frac{1}{T}\sum_{t=1}^{T} \log p(y_t \mid \boldsymbol{x}, \boldsymbol{y}_{<t}, \boldsymbol{w})\right)$.

Semantic cluster probability distribution. Evaluating the whole set of possible output sequences $\mathcal{Y}$ is usually intractable, as it scales exponentially with the sequence length T, thus $\mathcal{O}(|\mathcal{V}|^T)$. Furthermore, semantic equivalences of sequences should be taken into account (Kuhn et al., 2023). A language model generating different output sequences from the same input sequence does not necessarily indicate high predictive uncertainty if they mean the same thing. Hence, predictive uncertainty should be considered high only when different output sequences also exhibit semantically diverse meanings. Instead of directly utilizing the distribution over output sequences $p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w})$, the distribution over semantic clusters

$$p(c \mid \boldsymbol{x}, \boldsymbol{w}) = \sum_{\mathcal{Y}} p(c \mid \boldsymbol{y})\, p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w}) = \sum_{\mathcal{Y}} \mathbb{1}\{\boldsymbol{y} \in c\}\, p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w}) \tag{1}$$

is used to derive the predictive uncertainty in the autoregressive language generation setting. It expresses the probability of the language model generating an output sequence belonging to a specific semantic cluster. The conditional probability distribution $p(c \mid \boldsymbol{y})$ expresses the probability of an output sequence $\boldsymbol{y}$ belonging to a semantic cluster $c \in \mathcal{C}$. We assume that $\boldsymbol{y}$ belongs to a single semantic cluster as is done in Kuhn et al. (2023). Thus, $\mathbb{1}\{\boldsymbol{y} \in c\} = 1$ iff $\boldsymbol{y}$ belongs to semantic cluster c. Following Kuhn et al. (2023), we employ the natural language inference model DeBERTa (Williams et al., 2018; He et al., 2021) as a bi-directional entailment classifier. It predicts whether two sequences entail each other, are neutral, or contradict each other. The two sequences belong to the same semantic cluster if they entail each other in both orders.

Semantic uncertainty. Adopting the definition for the predictive uncertainty of a given, pre-selected model in the classification setting introduced by Schweighofer et al. (2023a;b), the total predictive semantic uncertainty (see Sec. C in the appendix for details):

$$\underbrace{\mathrm{E}_{\tilde{\boldsymbol{w}}}\left[\mathrm{CE}(p(c \mid \boldsymbol{x}, \boldsymbol{w})\ ;\ p(c \mid \boldsymbol{x}, \tilde{\boldsymbol{w}}))\right]}_{\text{total}} = \underbrace{\mathrm{H}(p(c \mid \boldsymbol{x}, \boldsymbol{w}))}_{\text{aleatoric}} + \underbrace{\mathrm{E}_{\tilde{\boldsymbol{w}}}\left[\mathrm{KL}(p(c \mid \boldsymbol{x}, \boldsymbol{w}) \parallel p(c \mid \boldsymbol{x}, \tilde{\boldsymbol{w}}))\right]}_{\text{epistemic}} \tag{2}$$

can be additively decomposed into an aleatoric semantic and an epistemic semantic uncertainty. $\mathrm{E}_{\tilde{\boldsymbol{w}}} = \mathrm{E}_{\tilde{\boldsymbol{w}} \sim p(\tilde{\boldsymbol{w}} \mid \mathcal{D})}$ is a posterior expectation. The epistemic term is again a posterior expectation, which is particularly challenging to estimate for current language models that are in the range of billions of parameters (Zhang et al., 2022; Touvron et al., 2023). The aleatoric term is precisely the *semantic entropy* proposed by Kuhn et al. (2023). The aleatoric semantic uncertainty considers the entropy of the semantic cluster probability distribution as of Eq. (1) under a given language model:

$$\mathrm{H}(p(c \mid \boldsymbol{x}, \boldsymbol{w})) = -\sum_{\mathcal{C}} \log p(c \mid \boldsymbol{x}, \boldsymbol{w})\, p(c \mid \boldsymbol{x}, \boldsymbol{w})\ . \tag{3}$$

2.2 Estimating aleatoric uncertainty in NLG

Kuhn et al. (2023) proposes to approximate the semantic entropy given by Eq. (3) through Monte Carlo (MC) sampling. The estimator is thus given by

$$\mathrm{H}(p(c \mid \boldsymbol{x}, \boldsymbol{w})) \approx -\frac{1}{N}\sum_{n=1}^{N} \log p(c^n \mid \boldsymbol{x}, \boldsymbol{w})\ , \tag{4}$$

where c^n is sampled according to the semantic cluster probability distribution $p(c \mid \boldsymbol{x}, \boldsymbol{w})$. However, we cannot directly sample from $p(c \mid \boldsymbol{x}, \boldsymbol{w})$, but only from $p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w})$. Therefore, it is impossible to directly use the estimator in Eq. (4). Instead, one can first approximate

$$p(c \mid \boldsymbol{x}, \boldsymbol{w}) \approx \frac{1}{N}\sum_{n=1}^{N} \mathbb{1}\{\boldsymbol{y}^n \in c\} = \frac{1}{N}\sum_{n=1}^{N} \mathbb{1}\{\boldsymbol{y}^n \in c\}\, \frac{p(\boldsymbol{y}^n \mid \boldsymbol{x}, \boldsymbol{w})}{q(\boldsymbol{y}^n \mid \boldsymbol{x}, \boldsymbol{w})}\ , \tag{5}$$

through MC sampling, where $\boldsymbol{y}^n$ is sampled according to $p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w})$ or when using importance sampling according to a proposal distribution $q(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w})$. We want to utilize importance sampling, because drawing samples from a billion-parameter model is computationally costly. Sample sizes are typically in the low double-digit range (Malinin & Gales, 2021; Kuhn et al., 2023; Duan et al., 2023) and only go up to a few hundred for studies conducted on large-scale compute (Kadavath et al., 2022). We use the estimator given by Eq. (5) to approximate the aleatoric semantic uncertainty defined by the semantic entropy in Eq. (3) by summing over all clusters $c_1, ..., c_M$ found in $\{\boldsymbol{y}^n\}_{n=1}^N$ through the bi-directional entailment classifier (see Eq. (16) in the appendix). Importantly, the quality of the approximation strongly depends on how many clusters with substantial probability mass have been found by sampling. Our method SDLG (described in Sec. 3) is more likely to find such clusters. SDLG implicitly constructs the proposal distribution $q(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w})$. It takes the form

$$q(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w}) \;=\; \frac{p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w})}{p(y_t \mid \boldsymbol{y}_{<t}, \boldsymbol{x}, \boldsymbol{w})}\,, \tag{6}$$

where $p(y_t \mid \boldsymbol{y}_{<t}, \boldsymbol{x}, \boldsymbol{w})$ is the likelihood of the alternative token that is substituting a token at index t. Intuitively, it means that we have to adjust the MC estimate in Eq. (5), because SDLG interferes in sampling and changes y_t deterministically. For more details on the assumptions and a step-by-step derivation see Sec. D in the appendix.

3 Semantic-Diverse Language Generation

Recent work by Kuhn et al. (2023) samples the output sequences through multinomial sampling. This naive approach, however, is inherently inefficient, as it tends to generate multiple duplicates of likely sequences despite knowing the likelihood of generating each sequence. Furthermore, semantic clusters may be missed, which would be important for accurately estimating the aleatoric semantic uncertainty. On the contrary, SDLG explicitly searches for the output sequences that not only have a high likelihood but also a high semantic diversity. It seeks to efficiently explore semantic clusters, capturing important modes of $p(c \mid \boldsymbol{x}, \boldsymbol{w})$.

Semantic diversity and where to find it. Given an output sequence $\boldsymbol{y}$ that was generated by a given language model from an input sequence $\boldsymbol{x}$, how can we generate another output sequence that has different semantics from $\boldsymbol{y}$? In natural language, each sentence is composed of words that contribute to the semantics to varying extents. Our method focuses on identifying and substituting the words most important for the overall semantics. Thus, we score potential token substitutions based on the impact in altering the semantics of $\boldsymbol{y}$. To compute the scores, we utilize a 'self-loss' L that expresses to what degree $\boldsymbol{y}$ is semantically different from itself. L is used to calculate the gradient $\nabla_{\boldsymbol{z}_i}\mathrm{L}$ w.r.t. the token embedding $\boldsymbol{z}_i$ (representing y_i), which quantifies the required change to alter the semantic meaning. In the following, y_i refers to a present token at position i in the output sequence $\boldsymbol{y}$, while v_j refers to an alternative token at position j in the vocabulary $\mathcal{V}$.

1) Attribution score. Our first objective is to identify which present token should be changed according to the computed gradient vector. The *attribution* score of $y_i \in \boldsymbol{y}$

$$I_i \;=\; \|\boldsymbol{z}_i \odot \nabla_{\boldsymbol{z}_i}\mathrm{L}\|_2 \tag{7}$$

is defined as the Euclidean distance $\|\cdot\|_2$ of the gradient vector multiplied elementwise with the embedding vector $\boldsymbol{z}_i$ that represents the present token y_i. Higher scores indicate a higher impact of the token y_i on altering the semantics when being changed (Adebayo et al., 2018).

2) Substitution score. Identifying which present token should be changed is important but insufficient. There could be no proper substitution that alters the semantic meaning. Thus, our second objective is to identify appropriate alternative tokens that most effectively alter the semantics when substituting the present token. The *substitution* score of $v_j \in \mathcal{V} \setminus \{y_i\}$

$$I_{ij} \;=\; \frac{(\boldsymbol{z}_i - \boldsymbol{z}_j) \cdot \nabla_{\boldsymbol{z}_i}\mathrm{L}}{\|\boldsymbol{z}_i - \boldsymbol{z}_j\|_2 \;\; \|\nabla_{\boldsymbol{z}_i}\mathrm{L}\|_2} \tag{8}$$

is defined as the cosine similarity $sim(\cdot,\cdot)$ between the gradient vector and the difference between the present and the alternative token's embedding vectors $\boldsymbol{z}_i, \boldsymbol{z}_j$. Higher scores indicate closer alignment between changes in the embedding vector with changes in the semantics (Mikolov et al., 2013).

3) Importance score. Substituting the present token with a less likely alternative token might drastically reduce the overall likelihood of the output sequence. Thus, our third objective is to favor alternative tokens that also remain a high likelihood when substituting the present token. The *importance* score of $v_j \in \mathcal{V} \setminus \{y_i\}$

$$P_{ij} \;=\; p(v_j \mid \boldsymbol{y}_{<i}, \boldsymbol{x}, \boldsymbol{w}) \tag{9}$$

is simply defined as the probability that the language model assigns to v_j given the context. The context is the input plus output sequence up to the token that is to be substituted. Higher scores indicate a greater likelihood of substituting the present token with the alternative token.

Generation of semantic diverse sequences. For every present token y_i in the given output sequence $\boldsymbol{y}$, SDLG computes three distinct scores for each alternative token v_j from the vocabulary $\mathcal{V}$. Subsequently, the potential substitutions are ranked according to the value of each of the three distinct scores. A new output sequence is then generated by deliberately substituting the highest-ranked token pair. The subsequent tokens are disregarded as they get conditioned on the substituted token, affecting their predictive probability distributions. The rest of the new output sequence is generated by the language model with the usual sampling strategy.

4 EXPERIMENTS

Models and Data. We utilize the OPT model series (Zhang et al., 2022) throughout the experiments, with model sizes ranging from 2.7 to 30 billion parameters. The experiments were performed on three free-form question-answering datasets that are frequently used as benchmarks for NLG uncertainty estimation. We use TruthfulQA (Lin et al., 2022a) corresponding to whole sentence answers to closed-book questions, CoQA (Reddy et al., 2019) corresponding to medium to shorter length answers to open-book questions, and TriviaQA (Joshi et al., 2017) corresponding to short, precise answers to closed-book questions. In general, the four models and three datasets assess the performance of NLG uncertainty estimation methods across varying model capabilities, generation lengths, and retrieval methods from both the prompt and internal model weights.

Evaluation. The quality of an uncertainty estimator is evaluated by how well it correlates with the respective correctness of an answer of the model; correct answers should be assigned a lower uncertainty than incorrect answers. Following the evaluation protocol of Kuhn et al. (2023); Lin et al. (2023); Duan et al. (2023), we utilize the statistics-based metrics Rouge-L and Rouge-1 (Lin, 2004), together with the Transfer learning-based metric BLEURT (Sellam et al., 2020). The correctness is evaluated on the most-likely generation, sampled using a beam search with 5 beams and also serving as the first generation for every NLG uncertainty estimation method. AUROC is utilized as a metric for classifying the correct vs. incorrect answers, using the respective uncertainty estimator as a score. The higher the AUROC, the higher the correlation between the uncertainty estimator and the correctness of the answers.

Baselines. We compare SDLG against methods directly utilizing the predictive entropy on a token level, namely Predictive Entropy (PE), Length-Normalized Predictive Entropy (LN-PE) (Malinin & Gales, 2021) and Shifting Attention to Relevance (SAR) (Duan et al., 2023), as well as methods utilizing Semantic Entropy on a sequence level, namely Semantic Entropy with Multinomial Sampling (SE_{MS}) (Kuhn et al., 2023) and with Diverse Beam Search (SE_{DBS}) (Vijayakumar et al., 2018). Although DBS has not explicitly been proposed for uncertainty estimation in NLG, we use it as a traditional sampling method enforcing generation diversity. The SE_{MS} temperature and the SE_{DBS} penalty term have been optimized. All methods use 10 generations for the uncertainty estimate.

SDLG. To compute the token scores as discussed in Sec. 3, we use the same natural language inference model DeBERTa (Williams et al., 2018; He et al., 2021) that we also used for determining semantic clusters (see Sec. 2.2). To be precise, we predict the semantic similarity between the first (most-likely) generation and itself by feeding it into the DeBERTa model twice. The resulting prediction is then employed to compute the loss L to the target prediction `contradiction`, which in turn is used to compute the gradient vector for the token scores. The gradient vector quantifies the change in the token embedding that is required to push the prediction of the DeBERTa model towards `contradiction` and thus alters the semantic meaning. We empirically found that the performance of our method is quite robust with respect to the weighting of the three individual token rankings. Thus, throughout our final experiments, we simply average them to derive the final token ranking $\mathcal{R}$. Following this token ranking, nine output sequences are generated.

Table 1: AUROC using different uncertainty measures as a score to distinguish between correct and incorrect answers. The threshold of the correctness metric Rouge-L (F1 score) is set to 0.5 and is computed as *[max score to a true reference answer] - [max score to a false reference answer]*.

Dataset	Model	LN-PE	PE	SAR	SE_{MS}	SE_{DBS}	$\text{SE}_{\texttt{SDLG}}$
TruthfulQA	**OPT-2.7b**	.439	.517	.611	.846	.686	**.920**
	OPT-6.7b	.446	.510	.555	.781	.637	**.881**
	OPT-13b	.676	.712	.775	.896	.819	**.956**
	OPT-30b	.482	.542	.517	.864	.788	**.927**
CoQA	**OPT-2.7b**	.717	.693	.733	**.744**	.697	**.744**
	OPT-6.7b	.728	.703	.748	**.764**	.714	.759
	OPT-13b	.723	.697	.747	.758	.720	**.760**
	OPT-30b	.732	.698	.742	.767	.713	**.768**
TriviaQA	**OPT-2.7b**	.769	.787	.785	.804	.808	**.809**
	OPT-6.7b	.790	.805	.804	.822	.823	**.829**
	OPT-13b	.807	.820	.819	.838	.841	**.845**
	OPT-30b	.799	.812	.815	.831	.837	**.840**

Analysis of results. Our method largely outperforms all current methods on the three correctness metrics Rouge-L, Rouge-1, and BLEURT, nine different thresholds, and different numbers of sampled answers. The results are summarized in Tab. 1. It can be observed that simple token-level diversity enforced by higher temperatures in multinomial sampling or by Diverse Beam Search (Vijayakumar et al., 2018) is insufficient for capturing semantic diversity essential for uncertainty estimation.

Semantic clusters. Our method results in at least a 19% increase of semantic clusters after the second generation, as well as at least a 74% increase of semantic clusters after the tenth generation, compared to multinomial sampling with the highest-performing temperature when considering all model sizes and averaging over all CoQA instances. This is because SDLG explicitly searches for a different semantic meaning. Compared to current methods, SDLG does not sample the same output sequence twice. Also, unlike current methods that rely on finding the optimal sampling temperature, SDLG does not require hyperparameter tuning as it controls the sampling process by deterministically exchanging relevant tokens instead of relying on chance to obtain diverse samples. Future work could consider the semantics of all previously generated output sequences or applying SDLG recursively on those, which could lead to further improvements.

Computational expenses. SDLG requires an additional forward and backward pass through the DeBERTa model (Williams et al., 2018; He et al., 2021) when sampling answers, compared to multinomial sampling used by the current methods. However, the main computational effort is during the generation of output sequences, since forward and backward passes through the DeBERTa model with a few hundred million parameters is usually small compared to generating only a single token using a language model with billions of parameters. Also, since our method deterministically changes a specific token within the answer, preceding tokens are not regenerated, but only subsequent tokens. This results in SDLG requiring at least 33% fewer overall flops compared to multinomial sampling, when averaging over all CoQA instances. The advantage of our method over the current methods further increases with longer sequences and larger model sizes.

Experimental details, further results, and the two ablation studies are given in Sec. F in the appendix.

5 Conclusion

SDLG improves on uncertainty estimation in NLG by re-sampling words of the original output sequence that change the semantic meaning, while being likely to be generated themselves. This way it finds likely output sequences with different semantic meanings. Our experiments on free-form question answering show that SDLG increases the overall quality of the uncertainty estimator while being significantly more sample efficient. This work focuses on estimating the aleatoric component of semantic uncertainty given by Eq. (2). Future work should investigate how to effectively assess the epistemic component. SDLG can significantly enhance the applicability of uncertainty estimation in NLG and aids in detecting hallucinations.

ACKNOWLEDGEMENTS

The ELLIS Unit Linz, the LIT AI Lab, the Institute for Machine Learning, are supported by the Federal State Upper Austria. We thank the projects Medical Cognitive Computing Center (MC3), INCONTROL-RL (FFG-881064), PRIMAL (FFG-873979), S3AI (FFG-872172), DL for GranularFlow (FFG-871302), EPILEPSIA (FFG-892171), AIRI FG 9-N (FWF-36284, FWF-36235), AI4GreenHeatingGrids (FFG- 899943), INTEGRATE (FFG-892418), ELISE (H2020-ICT-2019-3 ID: 951847), Stars4Waters (HORIZON-CL6-2021-CLIMATE-01-01). We thank Audi.JKU Deep Learning Center, TGW LOGISTICS GROUP GMBH, Silicon Austria Labs (SAL), FILL Gesellschaft mbH, Anyline GmbH, Google, ZF Friedrichshafen AG, Robert Bosch GmbH, UCB Biopharma SRL, Merck Healthcare KGaA, Verbund AG, GLS (Univ. Waterloo), Software Competence Center Hagenberg GmbH, Borealis AG, TÜV Austria, Frauscher Sensonic, TRUMPF and the NVIDIA Corporation. Kajetan Schweighofer acknowledges travel support from ELISE (GA no 951847).

REFERENCES

Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2018.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In Jian Su, Kevin Duh, and Xavier Carreras (eds.), *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2289–2294, Austin, Texas, November 2016. Association for Computational Linguistics.

Alvin Chan, Yew-Soon Ong, Bill Pung, Aston Zhang, and Jie Fu. Cocon: A self-supervised approach for controlled text generation. In *International Conference on Learning Representations*, 2021.

Roi Cohen, Mor Geva, Jonathan Berant, and Amir Globerson. Crawling the internal knowledge-base of language models. In Andreas Vlachos and Isabelle Augenstein (eds.), *Findings of the Association for Computational Linguistics: EACL 2023*, pp. 1856–1869, Dubrovnik, Croatia, May 2023a. Association for Computational Linguistics.

Roi Cohen, May Hamri, Mor Geva, and Amir Globerson. LM vs LM: Detecting factual errors via cross examination. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12621–12640, Singapore, December 2023b. Association for Computational Linguistics.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, USA, 2006. ISBN 0471241954.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*, 2020.

Jinhao Duan, Hao Cheng, Shiqi Wang, Chenan Wang, Alex Zavalny, Renjing Xu, Bhavya Kailkhura, and Kaidi Xu. Shifting attention to relevance: Towards the uncertainty estimation of large language models. *ArXiv*, abs/2307.01379, 2023.

Deep Ganguli, Amanda Askell, Nicholas Schiefer, Thomas I. Liao, Kamilė Lukošiūtė, Anna Chen, Anna Goldie, Azalia Mirhoseini, Catherine Olsson, Danny Hernandez, Dawn Drain, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jackson Kernion, Jamie Kerr, Jared Mueller, Joshua Landau, Kamal Ndousse, Karina Nguyen, Liane Lovitt, Michael Sellitto, Nelson Elhage, Noemi Mercado, Nova DasSarma, Oliver Rausch, Robert Lasenby, Robin Larson, Sam Ringer, Sandipan Kundu, Saurav Kadavath, Scott Johnston, Shauna Kravec, Sheer El Showk, Tamera Lanham, Timothy Telleen-Lawton, Tom Henighan, Tristan Hume, Yuntao Bai, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, Christopher Olah, Jack Clark,

Samuel R. Bowman, and Jared Kaplan. The capacity for moral self-correction in large language models, 2023.

Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, Muhammad Shahzad, Wen Yang, Richard Bamler, and Xiao Xiang Zhu. A survey of uncertainty in deep neural networks. *Artificial Intelligence Review*, 56(1):1513–1589, Oct 2023. ISSN 1573-7462.

Marjan Ghazvininejad, Xing Shi, Jay Priyadarshi, and Kevin Knight. Hafez: an interactive poetry generation system. In Mohit Bansal and Heng Ji (eds.), *Proceedings of ACL 2017, System Demonstrations*, pp. 43–48, Vancouver, Canada, July 2017. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention. 2021.

Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. Learning to write with cooperative discriminators. In Iryna Gurevych and Yusuke Miyao (eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1638–1649, Melbourne, Australia, July 2018. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020.

Bairu Hou, Yujian Liu, Kaizhi Qian, Jacob Andreas, Shiyu Chang, and Yang Zhang. Decomposing uncertainty for large language models through input clarification ensembling, 2023.

Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning*, 110:457–506, 2021.

Daphne Ippolito, Reno Kriz, João Sedoc, Maria Kustikova, and Chris Callison-Burch. Comparison of diverse decoding methods from conditional language models. In Anna Korhonen, David Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3752–3762, Florence, Italy, July 2019. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, Vancouver, Canada, July 2017. Association for Computational Linguistics.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, Jackson Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom Brown, Jack Clark, Nicholas Joseph, Ben Mann, Sam McCandlish, Chris Olah, and Jared Kaplan. Language models (mostly) know what they know. 2022.

Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. CTRL: A conditional transformer language model for controllable generation. *CoRR*, abs/1909.05858, 2019.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *The Eleventh International Conference on Learning Representations*, 2023.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. A diversity-promoting objective function for neural conversation models. In Kevin Knight, Ani Nenkova, and Owen Rambow (eds.), *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 110–119, San Diego, California, June 2016. Association for Computational Linguistics.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori Hashimoto, Luke Zettlemoyer, and Mike Lewis. Contrastive decoding: Open-ended text generation as optimization. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12286–12312, Toronto, Canada, July 2023. Association for Computational Linguistics.

Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pp. 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods, 2022a.

Stephanie Lin, Jacob Hilton, and Owain Evans. Teaching models to express their uncertainty in words. *Transactions on Machine Learning Research*, 2022b. ISSN 2835-8856.

Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. Generating with confidence: Uncertainty quantification for black-box large language models, 2023.

Andrey Malinin and Mark Gales. Uncertainty estimation in autoregressive structured prediction. In *International Conference on Learning Representations*, 2021.

Potsawee Manakul, Adian Liusie, and Mark Gales. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 9004–9017, Singapore, December 2023. Association for Computational Linguistics.

Sabrina J. Mielke, Arthur Szlam, Emily Dinan, and Y-Lan Boureau. Reducing conversational agents' overconfidence through linguistic calibration. *Transactions of the Association for Computational Linguistics*, 10:857–872, 2022.

Tomas Mikolov, Kai Chen, G.s Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *Proceedings of Workshop at ICLR*, 2013, 01 2013.

Shrimai Prabhumoye, Alan W Black, and Ruslan Salakhutdinov. Exploring controllable text generation techniques. In Donia Scott, Nuria Bel, and Chengqing Zong (eds.), *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 1–14, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.

Victor Quach, Adam Fisch, Tal Schuster, Adam Yala, Jae Ho Sohn, Tommi S. Jaakkola, and Regina Barzilay. Conformal language modeling. 2023.

Siva Reddy, Danqi Chen, and Christopher D. Manning. Coqa: A conversational question answering challenge. 2019.

Jie Ren, Yao Zhao, Tu Vu, Peter J. Liu, and Balaji Lakshminarayanan. Self-evaluation improves selective generation in large language models, 2023.

Kajetan Schweighofer, Lukas Aichberger, Mykyta Ielanskyi, and Sepp Hochreiter. Introducing an improved information-theoretic measure of predictive uncertainty. *arXiv*, 2311.08309, 2023a.

Kajetan Schweighofer, Lukas Aichberger, Mykyta Ielanskyi, Günter Klambauer, and Sepp Hochreiter. Quantification of uncertainty with adversarial models. *arXiv*, 2307.03217, 2023b.

Thibault Sellam, Dipanjan Das, and Ankur P. Parikh. Bleurt: Learning robust metrics for text generation, 2020.

Yik-Cheung Tam. Cluster-based beam search for pointer-generator chatbot grounded by knowledge. *Computer Speech & Language*, 64:101094, 2020. ISSN 0885-2308.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher Manning. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 5433–5442, Singapore, December 2023. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R. Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. Diverse beam search: Decoding diverse solutions from neural sequence models. 2018.

Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In Marilyn Walker, Heng Ji, and Amanda Stent (eds.), *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1112–1122, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

Yijun Xiao and William Yang Wang. On hallucination and predictive uncertainty in conditional language generation. In Paola Merlo, Jorg Tiedemann, and Reut Tsarfaty (eds.), *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pp. 2734–2744, Online, April 2021. Association for Computational Linguistics.

Yuxin Xiao, Paul Pu Liang, Umang Bhatt, Willie Neiswanger, Ruslan Salakhutdinov, and Louis-Philippe Morency. Uncertainty quantification with pre-trained language models: A large-scale empirical analysis. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 7273–7284, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

Hanqing Zhang, Haolin Song, Shaoyu Li, Ming Zhou, and Dawei Song. A survey of controllable text generation using transformer-based pre-trained language models. *ACM Comput. Surv.*, 56(3), oct 2023. ISSN 0360-0300.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. Opt: Open pre-trained transformer language models, 2022.

Kaitlyn Zhou, Dan Jurafsky, and Tatsunori Hashimoto. Navigating the grey area: How expressions of uncertainty and overconfidence affect language models. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 5506–5524, Singapore, December 2023. Association for Computational Linguistics.

A IMPACT STATEMENT

This work focuses on assessing the uncertainty in natural language generation using language models. Our primary goal is to increase the robustness of language models, assess the reliability of their predicted output sequences, and detect when a language model is hallucinating. Therefore, we contend that our work makes a positive contribution to society in several aspects:

1. Improved discernment of certainty in model predictions enhances practical application in real-world scenarios. This can be implemented by signaling uncertainty to users, such as through highlighting dubious sections of responses or opting not to display uncertain outputs altogether.
2. Reliable uncertainty estimates may increase the trust of the user in the language model, as it provides a basis to gauge the quality of the answer.

However, while we expect mainly a positive impact on society, there are also potential negative aspects:

1. Enhanced uncertainty estimation might not yield expected outcomes if users lack the necessary training to interpret these estimates effectively.
2. While better uncertainty assessment can foster usability and user trust, it also carries the risk of creating undue reliance on these models. It is crucial to maintain human oversight and critical evaluation of language model outputs, as over-reliance can be detrimental.

It is important to note that our method evaluates uncertainty based on the information available to the language model. Therefore, it may inaccurately deem a factually incorrect answer as certain if the model's knowledge base contains similar errors. This issue, often perceived as model 'hallucination', is not a reflection of the model's uncertainty, but rather a result of factual inaccuracies in the underlying data that is additional to hallucinations due to uncertainty.

B RELATED WORK

Uncertainty Estimation in NLG. Several works utilized the language model itself to obtain a prediction of their uncertainty, whether that be numerical or verbal (Mielke et al., 2022; Lin et al., 2022b; Kadavath et al., 2022; Cohen et al., 2023a; Ganguli et al., 2023; Ren et al., 2023; Tian et al., 2023). Cohen et al. (2023b) utilizes cross-examination, where one language model generates the output sequence and the other model acts as an examiner to assess the uncertainty. Zhou et al. (2023) investigates the behavior of language models when expressing their (un)certainty.

A large body of work focuses on sampling a set of output sequences to obtain sampling-based uncertainty estimators. Xiao & Wang (2021); Malinin & Gales (2021); Hou et al. (2023) incorporate both aleatoric and epistemic estimates of uncertainty, where epistemic uncertainty due to model selection is considered. While Kuhn et al. (2023); Lin et al. (2023); Duan et al. (2023) evaluate only aleatoric uncertainty under a single given model, they take the semantic equivalence of potential output sequences into account. Manakul et al. (2023) also samples a set of output sequences, but utilizes them as input to another language model to assess the uncertainty.

Another approach to uncertainty estimation in NLG is conformal prediction (Quach et al., 2023), where a stopping rule for generating output sequences is calibrated. Additionally, Xiao et al. (2022) empirically analyzed how factors such as model architecture and training details influence the uncertainty estimates in language models.

Generating diverse output sequences. Li et al. (2016) proposes an alternative training procedure of language models to avoid generic, input-independent output sequences and increase diversity. Diverse beam search (Vijayakumar et al., 2018) optimizes for a diversity-augmented objective across beam groups, based on diversity heuristics. Ippolito et al. (2019) compares diversity encouraging decoding strategies. Nucleus sampling (Holtzman et al., 2020) generates higher quality as well as more diverse output sequences, but does not explicitly encourage semantic diversity. Contrastive decoding (Li et al., 2023) utilizes a second, weaker language model, where the decoding algorithm favors tokens generated by the stronger model and penalizes tokens generated by the weaker model. Tam (2020) utilizes semantic clustering during beam search, which is used to prune beams and diversify the remaining candidates. However, this only indirectly steers towards more diversity and relies on the diversity of the initial beams.

Closely related, but not directly targeting semantic diversity of output generations is the field of (neural) controllable text generation (Prabhumoye et al., 2020). Here, the generation process of the language model is steered by another model to e.g. adhere to a certain dialog structure, prevent toxic answers, or play a certain persona. Keskar et al. (2019) uses control codes added to the prompt to steer generation. Dathathri et al. (2020) propose the use of an external supervised classifier to control the generation. Chan et al. (2021) also utilizes an external classifier, but trains in a self-supervised setting. (Ghazvininejad et al., 2017; Holtzman et al., 2018) re-weight the probability distributions at each step of generating the output sequence. For further work in this field see the surveys by Prabhumoye et al. (2020); Zhang et al. (2023).

C Predictive uncertainty in classification.

We briefly revisit uncertainty estimation for classification tasks. A classification model parametrized by $\boldsymbol{w}$ and an input vector $\boldsymbol{x}$ are given. The predictive distribution under the given model is denoted as $p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w})$. We assume that the dataset $\mathcal{D}$ is fixed and was sampled according to the predictive distribution $p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w}^*)$ under true model parameters $\boldsymbol{w}^*$. Thus, we assume that the model class can approximate the true distribution sufficiently well, a common and often necessary assumption (Hüllermeier & Waegeman, 2021). The posterior distribution $p(\boldsymbol{w} \mid \mathcal{D})$ assigns a probability to how likely $\boldsymbol{w}$ matches $\boldsymbol{w}^*$. Following Schweighofer et al. (2023a;b), the predictive uncertainty of a given, pre-selected model parametrized by $\boldsymbol{w}$ is given by

$$\underbrace{\mathrm{E}_{\tilde{\boldsymbol{w}}}\left[\mathrm{CE}(p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w})\ ;\ p(\boldsymbol{y} \mid \boldsymbol{x}, \tilde{\boldsymbol{w}}))\right]}_{\text{total}} = \underbrace{\mathrm{H}(p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w}))}_{\text{aleatoric}} + \underbrace{\mathrm{E}_{\tilde{\boldsymbol{w}}}\left[\mathrm{KL}(p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w}) \parallel p(\boldsymbol{y} \mid \boldsymbol{x}, \tilde{\boldsymbol{w}}))\right]}_{\text{epistemic}} \tag{10}$$

where $\mathrm{E}_{\tilde{\boldsymbol{w}}} = \mathrm{E}_{\tilde{\boldsymbol{w}} \sim p(\tilde{\boldsymbol{w}} \mid \mathcal{D})}$. The total uncertainty, given by the posterior expectation of the cross-entropy $\mathrm{CE}(\cdot\ ;\ \cdot)$, is decomposed into the aleatoric and the epistemic uncertainty. The aleatoric uncertainty is the Shannon entropy $\mathrm{H}(\cdot)$ of the predictive distribution under the given model. The epistemic uncertainty is the posterior expectation of the Kullback-Leibler divergence $\mathrm{KL}(\cdot \parallel \cdot)$ between the given model and possible true models according to their posterior probability.

D On the proposal distribution induced by SDLG

In the following, we analyze the proposal distribution induced by SDLG. We consider a probabilistic transformation of one output sequence $\boldsymbol{y}'$ into another output sequence $\boldsymbol{y}$, given by $p(\boldsymbol{y} \mid \boldsymbol{y}', \boldsymbol{x}, \boldsymbol{w})$. This is introduced, because we have to sum over all possible output sequences $\boldsymbol{y}'$ that we could apply SDLG on, leading to

$$q(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w}) \ = \ \sum_{\boldsymbol{y}' \in \mathcal{Y}} p(\boldsymbol{y}' \mid \boldsymbol{x}, \boldsymbol{w})\, p(\boldsymbol{y} \mid \boldsymbol{y}', \boldsymbol{x}, \boldsymbol{w})\ . \tag{11}$$

We can write $p(\boldsymbol{y} \mid \boldsymbol{y}', \boldsymbol{x}, \boldsymbol{w})$ as an expected value over t, the index where SDLG chooses a different token:

$$p(\boldsymbol{y} \mid \boldsymbol{y}', \boldsymbol{x}, \boldsymbol{w}) \ = \ \sum_{t=1}^{T} p(t \mid \boldsymbol{y}', \boldsymbol{x}, \boldsymbol{w})\, p(\boldsymbol{y} \mid t, \boldsymbol{y}', \boldsymbol{x}, \boldsymbol{w})\ . \tag{12}$$

The construction of $\boldsymbol{y}$ from $\boldsymbol{y}'$ only changes one element of $\boldsymbol{y}'$ at position t and then generates the postfix new. Therefore, we have $p(\boldsymbol{y} \mid t, \boldsymbol{y}', \boldsymbol{x}, \boldsymbol{w}) = 0$ for $\boldsymbol{y}'_{<t} \neq \boldsymbol{y}_{<t}$. Consequently, $\sum_{\boldsymbol{y}' \in \mathcal{Y}} p(\boldsymbol{y}' \mid \boldsymbol{x}, \boldsymbol{w})$ can be reduced to $\sum_{\boldsymbol{y}'_{>t} \in \mathcal{Y}_{>t}} p(\boldsymbol{y}'_{>t} \mid \boldsymbol{y}_{\leqslant t}, \boldsymbol{x}, \boldsymbol{w})\, p(\boldsymbol{y}_{\leqslant t} \mid \boldsymbol{x}, \boldsymbol{w})$ if the factor $p(\boldsymbol{y} \mid t, \boldsymbol{y}', \boldsymbol{x}, \boldsymbol{w})$ is present. There is only one possibility for the prefix with $\boldsymbol{y}'_{<t} = \boldsymbol{y}_{<t}$.

Using Eq. (12) in Eq. (11) leads to

$$
\begin{aligned}
q(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w}) \; &= \qquad (13)\\
&= \sum_{\boldsymbol{y}' \in \mathcal{Y}} p(\boldsymbol{y}' \mid \boldsymbol{x}, \boldsymbol{w}) \sum_{t=1}^{T} p(t \mid \boldsymbol{y}', \boldsymbol{x}, \boldsymbol{w})\, p(\boldsymbol{y} \mid t, \boldsymbol{y}', \boldsymbol{x}, \boldsymbol{w})\\
&= \sum_{t=1}^{T} \sum_{\boldsymbol{y}' \in \mathcal{Y}} p(\boldsymbol{y}' \mid \boldsymbol{x}, \boldsymbol{w})\, p(t \mid \boldsymbol{y}', \boldsymbol{x}, \boldsymbol{w})\, p(\boldsymbol{y} \mid t, \boldsymbol{y}', \boldsymbol{x}, \boldsymbol{w})\\
&= \sum_{t=1}^{T} \sum_{\boldsymbol{y}'_{>t} \in \mathcal{Y}_{>t}} \sum_{\boldsymbol{y}'_{\leqslant t} \in \mathcal{Y}_{\leqslant t}} p(\boldsymbol{y}'_{>t} \mid \boldsymbol{y}'_{\leqslant t}, \boldsymbol{x}, \boldsymbol{w})\, p(\boldsymbol{y}'_{\leqslant t} \mid \boldsymbol{x}, \boldsymbol{w})\, p(t \mid \boldsymbol{y}', \boldsymbol{x}, \boldsymbol{w})\, p(\boldsymbol{y} \mid t, \boldsymbol{y}', \boldsymbol{x}, \boldsymbol{w})\\
&= \sum_{t=1}^{T} \sum_{\boldsymbol{y}'_{>t} \in \mathcal{Y}_{>t}} p(\boldsymbol{y}'_{>t} \mid \boldsymbol{y}_{\leqslant t}, \boldsymbol{x}, \boldsymbol{w})\, p(\boldsymbol{y}_{\leqslant t} \mid \boldsymbol{x}, \boldsymbol{w})\, p(t \mid \boldsymbol{y}'_{>t}, \boldsymbol{y}_{\leqslant t}, \boldsymbol{x}, \boldsymbol{w})\, p(\boldsymbol{y}_{>t} \mid \boldsymbol{y}'_{>t}, \boldsymbol{y}_{\leqslant t}, \boldsymbol{x}, \boldsymbol{w})\\
&= \sum_{t=1}^{T} \sum_{\boldsymbol{y}'_{>t} \in \mathcal{Y}_{>t}} p(\boldsymbol{y}'_{>t} \mid \boldsymbol{y}_{\leqslant t}, \boldsymbol{x}, \boldsymbol{w})\, p(\boldsymbol{y}_{\leqslant t} \mid \boldsymbol{x}, \boldsymbol{w})\, p(t \mid \boldsymbol{y}'_{>t}, \boldsymbol{y}_{\leqslant t}, \boldsymbol{x}, \boldsymbol{w})\, p(\boldsymbol{y}_{>t} \mid \boldsymbol{y}_{\leqslant t}, \boldsymbol{x}, \boldsymbol{w})\\
&= \sum_{t=1}^{T} p(\boldsymbol{y}_{\leqslant t} \mid \boldsymbol{x}, \boldsymbol{w}) \left(\sum_{\boldsymbol{y}'_{>t} \in \mathcal{Y}_{>t}} p(\boldsymbol{y}'_{>t} \mid \boldsymbol{y}_{\leqslant t}, \boldsymbol{x}, \boldsymbol{w})\, p(t \mid \boldsymbol{y}'_{>t}, \boldsymbol{y}_{\leqslant t}, \boldsymbol{x}, \boldsymbol{w}) \right) p(\boldsymbol{y}_{>t} \mid \boldsymbol{y}_{\leqslant t}, \boldsymbol{x}, \boldsymbol{w})\\
&= \sum_{t=1}^{T} p(\boldsymbol{y}_{\leqslant t} \mid \boldsymbol{x}, \boldsymbol{w})\, p(t \mid \boldsymbol{y}_{\leqslant t}, \boldsymbol{x}, \boldsymbol{w})\, p(\boldsymbol{y}_{>t} \mid \boldsymbol{y}_{\leqslant t}, \boldsymbol{x}, \boldsymbol{w})\\
&= \sum_{t=1}^{T} p(\boldsymbol{y}_{<t} \mid \boldsymbol{x}, \boldsymbol{w})\, p(y_t \mid \boldsymbol{y}_{<t}, \boldsymbol{x}, \boldsymbol{w})\, p(t \mid y_t, \boldsymbol{y}_{<t}, \boldsymbol{x}, \boldsymbol{w})\, p(\boldsymbol{y}_{>t} \mid y_t, \boldsymbol{y}_{<t}, \boldsymbol{x}, \boldsymbol{w})\\
&= \sum_{t=1}^{T} p(t \mid y_t, \boldsymbol{y}_{<t}, \boldsymbol{x}, \boldsymbol{w})\, p(\boldsymbol{y}_{<t} \mid \boldsymbol{x}, \boldsymbol{w})\, p(\boldsymbol{y}_{>t} \mid y_t, \boldsymbol{y}_{<t}, \boldsymbol{x}, \boldsymbol{w})\,,
\end{aligned}
$$

where we used $p(y_t \mid \boldsymbol{y}_{<t}, \boldsymbol{x}, \boldsymbol{w}) = 1$, since SDLG chooses y_t deterministically given $\boldsymbol{y}_{<t} = \boldsymbol{y}'_{<t}$. We assume that all probability mass in $p(t \mid y_t, \boldsymbol{y}_{<t}, \boldsymbol{x}, \boldsymbol{w})$ is at the actually observed t. This means, given all possible $\boldsymbol{y}'_{>t}$, t is the most probable position to induce a semantic change. This is a strong assumption, that needs further investigation in future work. Under this assumption, the final result in Eq. (13) reduces to

$$
q(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w}) \;=\; p(\boldsymbol{y}_{<t} \mid \boldsymbol{x}, \boldsymbol{w})\, p(\boldsymbol{y}_{>t} \mid y_t, \boldsymbol{y}_{<t}, \boldsymbol{x}, \boldsymbol{w})\,. \tag{14}
$$

We can re-write Eq. (14) in terms of the output sequence probability distribution $p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w})$ as

$$
q(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w}) \;=\; \frac{p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w})}{p(y_t \mid \boldsymbol{y}_{<t}, \boldsymbol{x}, \boldsymbol{w})}\,. \tag{15}
$$

E Details on Semantic Entropy Estimator

As it is unlikely that all semantic clusters $\mathcal{C}$ are found through clustering, it is not possible to calculate Eq. (3) directly. Instead, one can calculate

$$
\mathrm{H}(p(c \mid \boldsymbol{x}, \boldsymbol{w})) \;\approx\; -\sum_{m=1}^{M} \log p(c_m \mid \boldsymbol{x}, \boldsymbol{w})\, p(c_m \mid \boldsymbol{x}, \boldsymbol{w})\,. \tag{16}
$$

Inspecting the implementation of Kuhn et al. (2023) reveals that their estimator can be interpreted as Eq. (5) with additional importance sampling. Formally, they utilize an empirical proposal distribution $\hat{q}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w}) := \frac{1}{N} \sum_{n=1}^{N} \mathbb{1}\{\boldsymbol{y} = \boldsymbol{y}^n\}$, defined by the set of previously sampled output sequences $\{\boldsymbol{y}^n\}_{n=1}^{N}$. The approximation of the semantic cluster probability distribution given by

Eq. (5) (without importance sampling) thus changes to

$$p(c \mid \boldsymbol{x}, \boldsymbol{w}) \approx \frac{1}{N} \sum_{n=1}^{N} \mathbb{1}\{\boldsymbol{y}^n \in c\} \frac{p(\boldsymbol{y}^n \mid \boldsymbol{x}, \boldsymbol{w})}{\hat{q}(\boldsymbol{y}^n \mid \boldsymbol{x}, \boldsymbol{w})} , \tag{17}$$

where $\boldsymbol{y}^n$ is sampled according to $\hat{q}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w})$. As this distribution is known by design and can be enumerated, Eq. (17) simplifies to a weighted sum. The quality of this approximator strongly depends on the empirical distribution. Therefore, Eq. (17) should only be used in favor of Eq. (5) (without importance sampling) if $\{\boldsymbol{y}^n\}_{n=1}^N$ containts sequences that have very high probability under $p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w})$. The more these distributions differ, the higher the variance of the estimator, therefore, the lower the approximation quality. We utilized this variation both for the baseline using multinomial sampling, as well as in addition to the importance sampling we do with SDLG.

Furthermore, we found that the logarithm of the unnormalized probability estimator together with normalizing the probability estimator outside the logarithm in Eq. (16) improves empirical results for all methods that estimate the semantic entropy.

F Experimental Details and Further Experiments

Setup Details. To conduct the experiments, we use the over 800 closed-book questions in TruthfulQA (Lin et al., 2022a) corresponding to whole sentence answers, the almost 8,000 open-book questions in the development split of CoQA (Reddy et al., 2019) corresponding to medium to shorter length answers, and about 8,000 closed-book questions in the training split of TriviaQA (Joshi et al., 2017) corresponding to short, precise answers. We use a 5-shot, zero-shot, and 10-shot prompt for TruthfulQA, CoQA, and TriviaQA respectively. For computing the correctness of an answer, TurthfulQA provides both true and false reference answers, while CoQA and TriviaQA only provide true reference answers.

When computing the token scores, the natural language inference model might not build upon the same token embedding or even the same vocabulary $\mathcal{V}$ as the language model. Consequently, the embedding vectors need to be differentiably transformed to enable the computation of the gradients w.r.t. $\boldsymbol{z}_i$. Fortunately, there exist efficient exact methods to learn the optimal linear transformation between the two monolingual embedding spaces (Artetxe et al., 2016). We also note that substituting present tokens not corresponding to the beginning of a word is often impractical, particularly in the context of two monolingual embedding spaces. Consequently, we exclusively apply substitutions to tokens at the beginning of a word.

To further reduce the computational cost of our method, we decrease the number of computed token scores by implementing a token probability threshold of 0.001, under the rationale that tokens falling below this probability threshold would, in any case, be assigned a low importance rank.

Results. Tab. 2 summarizes the results on TruthfulQA, Tab. 3 the results on CoQA, and Tab. 4 the results on TriviaQA. Fig. 1 and Fig. 2 summarize the AUROC differences when sampling with SDLG instead of multinomial sampling (MS) when using three different correctness metrics (Rouge-L, Rouge-1, BLEURT), different thresholds for classifying answers as correct or incorrect, and a different number of samples without length-normalizing their probabilities. We observed that length normalizing yields superior results across uncertainty estimation methods only for the CoQA dataset when using a low correctness threshold. In summary, SDLG largely outperforms all other current methods regardless of the correctness metric, threshold, or number of samples considered. This highlights the fact that simple token-level diversity is insufficient for capturing semantic diversity, while explicitly searching for semantic diverse output sequences is essential for uncertainty estimation in NLG.

Semantic cluster ablation study. Fig. 4 shows the number of semantic clusters found when sampling with SDLG instead of multinomial sampling (MS), when using a different number of samples and averaging over all CoQA instances. SDLG samples more semantic clusters regardless of the number of samples considered.

Computational expenses ablation study. Fig. 4 shows the number of flops required when utilizing SDLG instead of multinomial sampling (MS) or Shifting Attention to Relevance (SAR), when using the OPT-2.7b model and averaging over all instances of the respective datasets. SDLG is computationally lessexpensive than the current methods across all three datasets.

Table 2: TruthfulQA results: AUROC using different uncertainty measures as a score to distinguish between correct and incorrect answers. Each method uses 10 samples without length-normalizing probabilities. The temperature for SE_{MS} is set to 2.0 and the penalty term for SE_{DBS} is set to 1.0.

Metric	**Model**	**LN-PE**		**PE**		**SAR**		**SE_{MS}**		**SE_{DBS}**		**$\text{SE}_{\texttt{SDLG}}$**	
Threshold	**$\geq$**	**0.3**	**0.5**	**0.3**	**0.5**	**0.3**	**0.5**	**0.3**	**0.5**	**0.3**	**0.5**	**0.3**	**0.5**
	2.7b	.438	.439	.523	.517	.604	.611	.786	.846	.647	.686	**.855**	**.920**
Rouge-L	**6.7b**	.470	.446	.530	.510	.570	.555	.739	.781	.621	.637	**.826**	**.881**
(F1 score)	**13b**	.653	.676	.693	.712	.754	.775	.869	.896	.795	.819	**.926**	**.956**
	30b	.478	.482	.531	.542	.506	.517	.828	.864	.746	.788	**.878**	**.927**
	2.7b	.438	.435	.515	.519	.599	.610	.774	.846	.644	.689	**.850**	**.923**
Rouge-1	**6.7b**	.464	.446	.513	.510	.560	.555	.717	.781	.608	.637	**.798**	**.881**
(F1 score)	**13b**	.657	.673	.692	.709	.753	.771	.872	.893	.795	.815	**.928**	**.953**
	30b	.482	.483	.532	.544	.614	.642	.827	.868	.749	.794	**.881**	**.930**
BLEURT	**2.7b**	.460	.456	.521	.535	.546	.573	.687	.727	.601	.643	**.723**	**.772**
	6.7b	.477	.497	.532	.560	.535	.564	.674	.706	.584	.623	**.715**	**.757**
	13b	.613	.628	.664	.686	.694	.728	.797	.836	.735	.777	**.844**	**.885**
	30b	.496	.489	.553	.546	.527	.519	.768	.799	.693	.710	**.800**	**.828**

Table 3: CoQA results: AUROC using different uncertainty measures as a score to distinguish between correct and incorrect answers. Each method uses 10 samples with length-normalized probabilities. The temperature for SE_{MS} is set to 0.5 and the penalty term for SE_{DBS} is set to 1.0.

Metric	**Model**	**LN-PE**		**PE**		**SAR**		**SE_{MS}**		**SE_{DBS}**		**$\text{SE}_{\texttt{SDLG}}$**	
Threshold	**$\geq$**	**0.3**	**0.5**	**0.3**	**0.5**	**0.3**	**0.5**	**0.3**	**0.5**	**0.3**	**0.5**	**0.3**	**0.5**
	2.7b	.712	.717	.672	.693	.727	.733	.743	**.744**	.707	.697	**.749**	**.744**
Rouge-L	**6.7b**	.725	.728	.680	.703	.747	.748	.768	**.764**	.731	.714	**.774**	.759
(F1 score)	**13b**	.719	.723	.672	.697	.744	.747	.765	.758	.745	.720	**.778**	**.760**
	30b	.734	.732	.676	.698	.738	.742	.779	.767	.742	.713	**.791**	**.768**
	2.7b	.711	.718	.669	.692	.726	.733	.742	.745	.707	.699	**.750**	**.746**
Rouge-1	**6.7b**	.726	.729	.679	.702	.747	.748	.771	**.765**	.734	.716	**.777**	.762
(F1 score)	**13b**	.719	.723	.671	.696	.744	.747	.765	.759	.747	.722	**.780**	**.762**
	30b	.736	.734	.677	.699	.755	.754	.780	**.768**	.744	.716	**.794**	**.768**
BLEURT	**2.7b**	.702	.703	.707	.714	.709	.708	**.736**	**.746**	**.736**	.745	**.736**	**.746**
	6.7b	.707	.705	.716	.722	.718	.717	**.746**	.752	.741	.748	**.746**	**.754**
	13b	.705	.704	.716	.720	.715	.714	.745	.751	.744	.750	**.747**	**.753**
	30b	.711	.711	.719	.724	.728	.728	.753	.759	.752	.758	**.754**	**.760**

Table 4: TriviaQA results: AUROC using different uncertainty measures as a score to distinguish between correct and incorrect answers. Each method uses 10 samples without length-normalizing probabilities. The temperature for SE_{MS} is set to 0.5 and the penalty term for SE_{DBS} is set to 1.0.

Metric	**Model**	**LN-PE**		**PE**		**SAR**		**SE_{MS}**		**SE_{DBS}**		**$\text{SE}_{\texttt{SDLG}}$**	
Threshold	**$\geq$**	**0.5**	**1.0**	**0.5**	**1.0**	**0.5**	**1.0**	**0.5**	**1.0**	**0.5**	**1.0**	**0.5**	**1.0**
	2.7b	.769	.775	.787	.813	.785	.800	.804	.831	.808	.831	**.809**	**.846**
Rouge-L	**6.7b**	.790	.792	.805	.823	.804	.811	.822	.833	.823	.833	**.829**	**.854**
(F1 score)	**13b**	.807	.810	.820	.836	.819	.828	.838	.849	.841	.849	**.845**	**.868**
	30b	.799	.795	.812	.820	.815	.816	.831	.834	.837	.835	**.840**	**.853**
	2.7b	.769	.775	.786	.812	.785	.800	.803	.830	.807	.831	**.808**	**.845**
Rouge-1	**6.7b**	.790	.792	.804	.823	.804	.811	.821	.833	.823	.832	**.829**	**.853**
(F1 score)	**13b**	.807	.810	.819	.836	.819	.828	.837	.848	.841	.849	**.843**	**.868**
	30b	.799	.796	.811	.820	.814	.815	.830	.833	.836	.835	**.838**	**.853**
BLEURT	**2.7b**	.750	.758	.793	.799	.771	.780	.813	.817	.817	.818	**.827**	**.833**
	6.7b	.770	.770	.807	.803	.788	.788	.822	.815	.823	.814	**.839**	**.835**
	13b	.787	.775	.820	.806	.804	.792	.837	.819	.840	.819	**.853**	**.840**
	30b	.776	.762	.806	.789	.796	.779	.823	.801	.828	.800	**.839**	**.818**

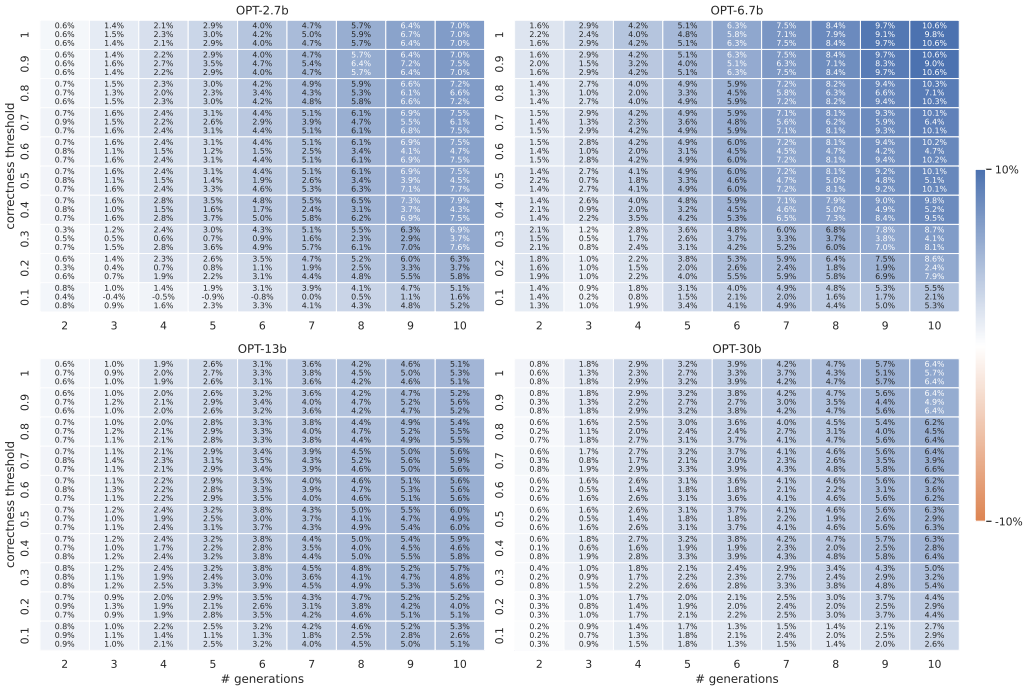

Figure 1: TurthfulQA dataset: AUROC difference when sampling with SDLG instead of multinomial sampling (MS), averaged over correctness metrics Rouge-L, Rouge-1, and BLEURT (values in that order). Positive values (blue) indicate higher average performance of SDLG.

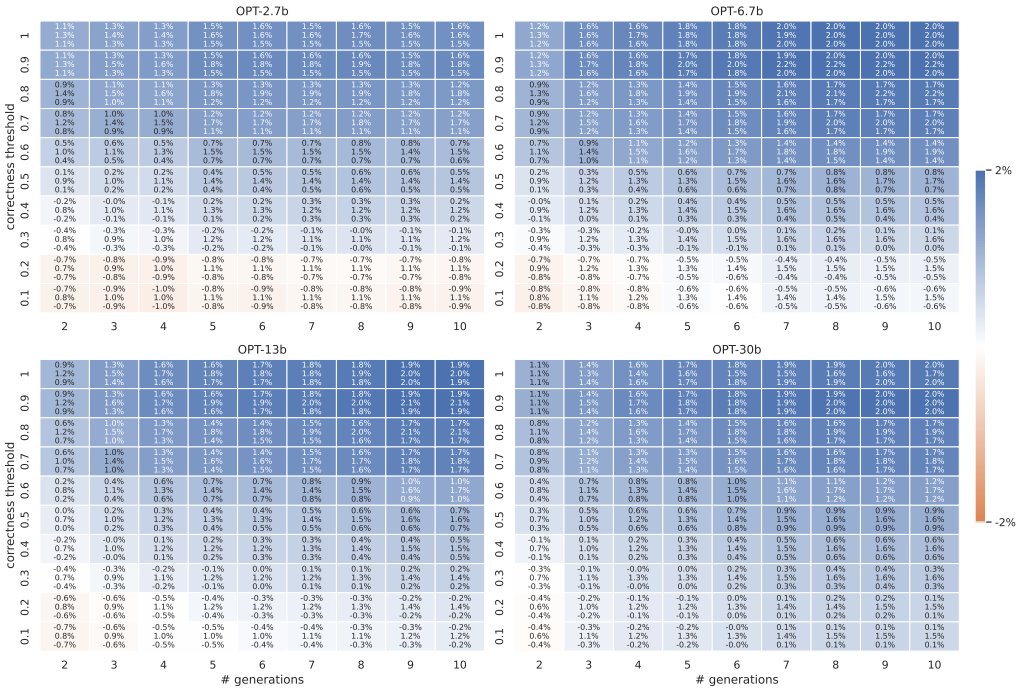

Figure 2: TriviaQA dataset: AUROC difference when sampling with SDLG instead of multinomial sampling (MS), averaged over correctness metrics Rouge-L, Rouge-1, and BLEURT (values in that order). Positive values (blue) indicate higher average performance of SDLG.

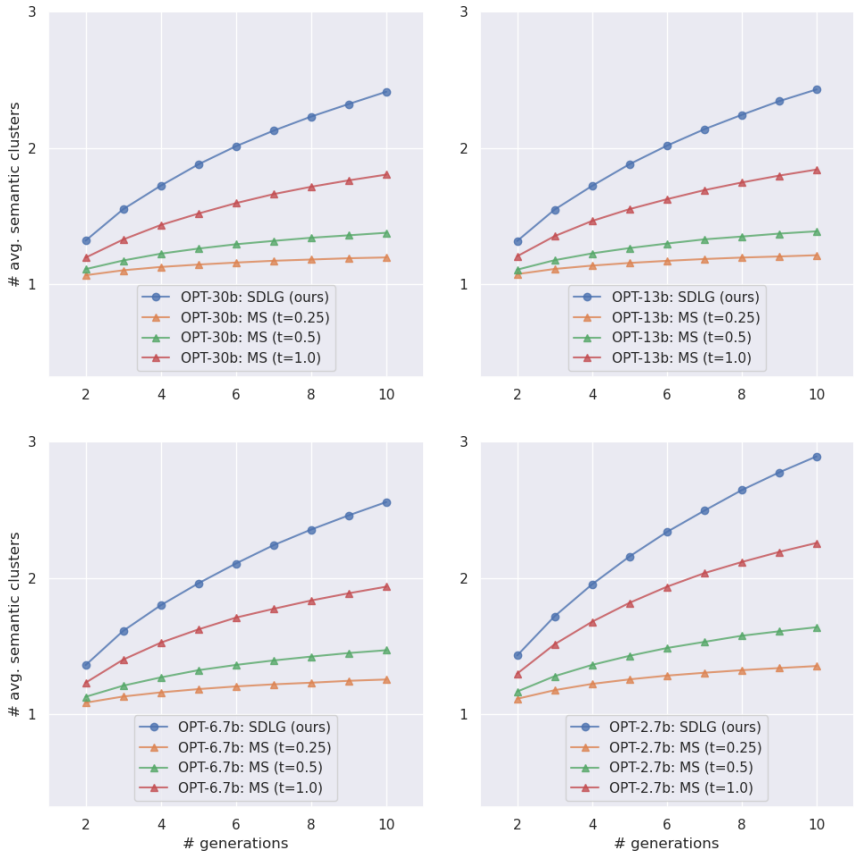

Figure 3: CoQA dataset: Average number of semantic clusters across instances. SDLG finds more semantic clusters than multinomial sampling (MS).

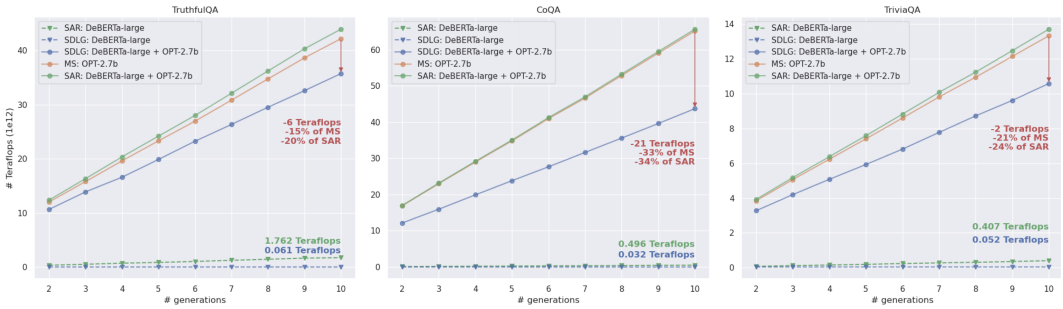

Figure 4: Average number of flops across instances. SDLG is computationally less expensive than current methods.

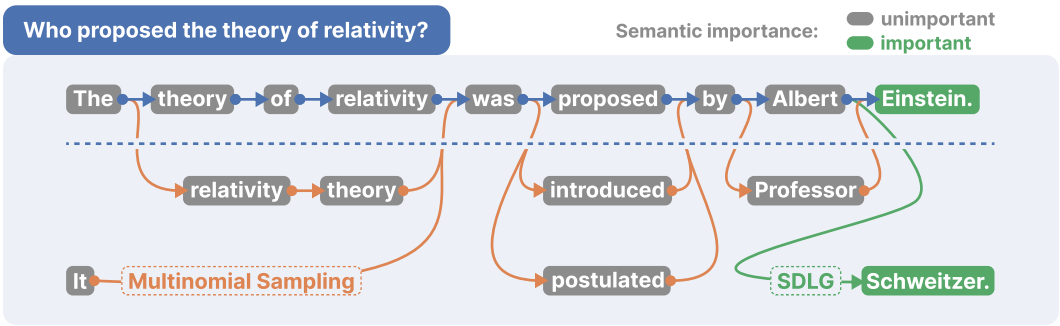

Figure 5: Standard multinomial sampling relies on chance to obtain semantically diverse output sequences, thus is prone to miss them. SDLG specifically searches for likely, but semantically diverse output sequences.

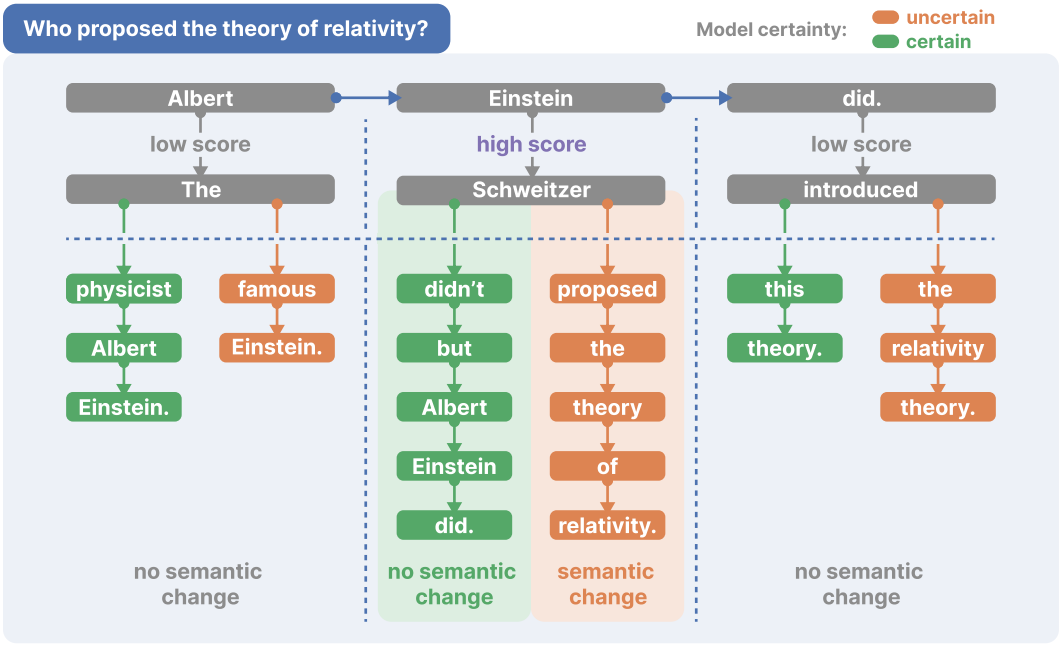

Figure 6: Illustrative example of applying SDLG.

G Insights into SDLG

Illustrative example. Fig. 6 considers the input sequence "Who proposed the theory of relativity?" with the given output sequence "Albert Einstein did.". When investigating the alternative tokens it becomes clear that not every substitution leads to a change in semantic meaning. It is important to substitute tokens that also receive a high score for altering the semantics. In this example, it is the present token corresponding to "Einstein" and the alternative token corresponding to "Schweitzer". Yet, this alone does not directly indicate a high level of uncertainty about the output sequence. High uncertainty should be attributed only if the new output sequence is completed and still has a different semantic meaning. If the language model is uncertain about the originator of the theory of relativity, it completes the new output sequence like "Albert Schweitzer proposed the theory of relativity.". This would suggest a high uncertainty estimate. However, if the language model is confident about the originator of the theory of relativity, it completes the new output sequence like "Albert Schweitzer didn't, but Albert Einstein did.". It is in favor of a low uncertainty estimate, since the model reinforces the original semantics. This illustrates that solely considering the predictive uncertainty on a token level is insufficient. Steering the generation towards a different semantics and then continuing the usual generation can be viewed as stress testing the language model.

Algorithm 1 Semantic-Diverse Language Generation

Output: Semantically diverse output sequences $\mathcal{S}$
Input: Input sequence $\boldsymbol{x}$, generative language model $g(\cdot)$, vocabulary $\mathcal{V}$, language inference model $e(\cdot,\cdot)$, cross-entropy loss function l, number of generations N

Initialize set of output sequences $\mathcal{S} \leftarrow \emptyset$
Generate most likely output sequence $\boldsymbol{y}^1 \leftarrow g(\boldsymbol{x})$
$\mathcal{S} \leftarrow \mathcal{S} \cup \{\boldsymbol{y}^1\}$
Get ranked token indices for replacement $\mathcal{R} \leftarrow$ Alg. 2
for $n = 2$ **to** N **do**
  Select token indices for replacement $(i, j) \leftarrow \mathcal{R}_n$
  Construct new input sequence $\boldsymbol{x}^n \leftarrow \boldsymbol{x} \oplus \boldsymbol{y}^1_{<i} \oplus v_j$
  Finish new generation $\boldsymbol{y}^n_{\text{rest}} \leftarrow g(\boldsymbol{x}^n)$
  Construct new full sequence $\boldsymbol{y}^n \leftarrow \boldsymbol{y}^1_{<i} \oplus v_j \oplus \boldsymbol{y}^n_{\text{rest}}$
  $\mathcal{S} \leftarrow \mathcal{S} \cup \{\boldsymbol{y}^n\}$
end for
return $\mathcal{S}$

Algorithm 2 Token Ranking by Semantic Impact

Output: Set of token pairs $\mathcal{R}$ ranked by semantic impact
Input: see Alg. 1

Initialize set of token scores $\mathcal{R} \leftarrow \emptyset$
Compute entailment $\boldsymbol{c} \leftarrow e(\boldsymbol{y}, \boldsymbol{y})$
Compute loss to contradiction $\mathrm{L} \leftarrow l(\boldsymbol{c}, \boldsymbol{c}_{\text{contradiction}})$
for $y_i \in \boldsymbol{y}$ **do**
  Compute gradient $\nabla_{\boldsymbol{z}_i}\mathrm{L} \leftarrow \frac{\partial \mathrm{L}}{\partial y_i}$
  Get attribution score $I_i \leftarrow \|\boldsymbol{z}_i \odot \nabla_{\boldsymbol{z}_i}\mathrm{L}\|_2$
  for $v_j \in \mathcal{V} \setminus \{y_i\}$ **do**
    Get substitution score $I_{ij} \leftarrow sim(\boldsymbol{z}_i - \boldsymbol{z}_j, \nabla_{\boldsymbol{z}_i}\mathrm{L})$
    Get importance score $P_{ij} \leftarrow p(v_j \mid \boldsymbol{y}_{<i}, \boldsymbol{x}, \boldsymbol{w})$
    $\mathcal{R} \leftarrow \mathcal{R} \cup \{(I_i, I_{ij}, P_{ij})\}$
  end for
end for
Rank token indices based on scores $\mathcal{R} \leftarrow \mathrm{Rank}(\mathcal{R})$
return $\mathcal{R}$

Algorithm. `SDLG` generates a new output sequence by deliberately substituting a token pair that has the best chance of altering the semantic meaning of a given output sequence $\boldsymbol{y}$ (see Alg. 1). To determine which token pair should be substituted, three distinct scores are computed for every present token in the given output sequence $\boldsymbol{y}$ and each alternative token from the vocabulary $\mathcal{V}$. The *attribution* score defined in Eq. (7) is identical for all alternative tokens associated with the same present token. The *substitution* score defined in Eq. (8) and the *importance* score defined in Eq. (9) are different for every alternative token. The potential substitutions are ranked according to the value of each of the three distinct scores (see Alg. 2).

Computational flow of calculating token scores. Fig. 7 shows the computational flow of how the three scores *Importance*, *Substitution*, and *Attribution* are computed for one specific token pair.

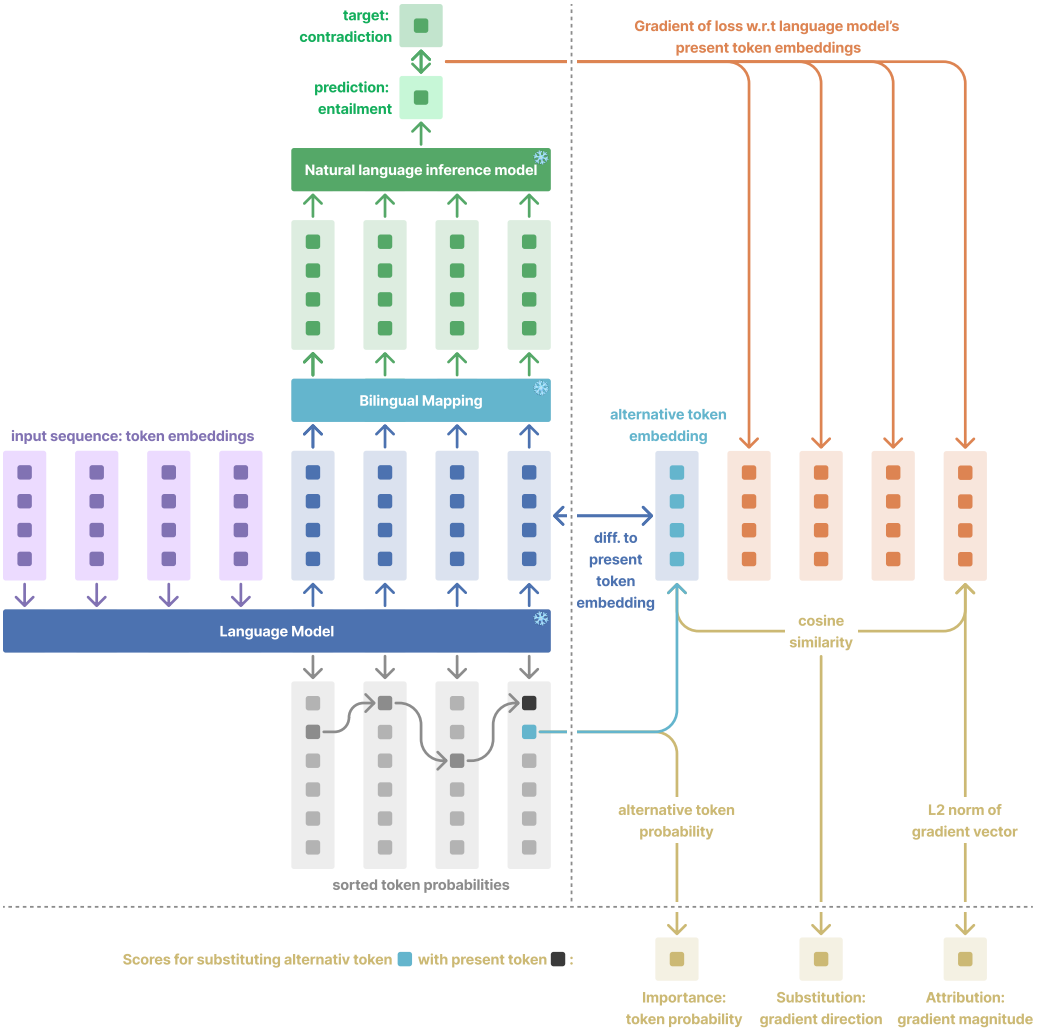

Figure 7: Visualization of how the three scores *Importance*, *Substitution*, and *Attribution* are computed for one specific token pair.