A Head to Predict and a Head to Question: Pre-trained Uncertainty Quantification Heads for Hallucination Detection in LLM Outputs

Anonymous ACL submission

Abstract

Large Language Models (LLMs) have the tendency to hallucinate, i.e., to sporadically generate false or fabricated information. This presents a major challenge, as hallucinations often appear highly convincing and users generally lack the tools to detect them. Uncertainty quantification (UQ) provides a framework for assessing the reliability of model outputs, aiding in the identification of potential hallucinations. In this work, we introduce pre-trained UQ heads: supervised auxiliary modules for LLMs that substantially enhance their ability to capture uncertainty compared to unsupervised UQ methods. Their strong performance stems from the powerful Transformer architecture in their design and informative features derived from LLM attention maps. Experimental evaluation shows that these heads are highly robust and achieve state-of-the-art performance in claim-level hallucination detection across both in-domain and out-of-domain prompts. Moreover, these modules demonstrate strong generalization to languages they were not explicitly trained on. We pre-train a collection of UQ heads for popular LLM series, including Mistral, Llama, and Gemma 2. We publicly release both the code and the pre-trained heads.[1]

1 Introduction

Uncertainty quantification (UQ) (Gal and Ghahramani, 2016; Baan et al., 2023; Geng et al., 2024; Zhang et al., 2024a) has become an increasingly important topic in natural language processing (NLP), particularly for addressing challenges with hallucinations and low-quality outputs of large language models (LLMs) (Malinin and Gales, 2021; Kuhn et al., 2023; Fadeeva et al., 2024). UQ offers the potential to improve the safety and reliability of LLM-based applications by flagging highly uncertain generations. Such generations could be discarded or marked as untrustworthy, thus reducing the risk of misleading information reaching users (Zhang et al., 2024a,b; Huang et al., 2024).

Current methods for detecting hallucinations and low-quality text often rely on external knowledge bases or additional LLMs (Manakul et al., 2023; Min et al., 2023; Chen et al., 2023). While useful, these approaches come with major drawbacks. Knowledge sources are often incomplete, and using a larger model to censor a smaller one is both computationally expensive and impractical. Instead, UQ assumes that LLMs naturally encode information about their own limitations, and this self-knowledge can be efficiently accessed to build safer, more practical systems.

There are many existing UQ techniques for well-defined tasks such as classification and regression (Zhang et al., 2019; He et al., 2020; Xin et al., 2021; Wang et al., 2022; Vazhentsev et al., 2023; He et al., 2024a). However, applying UQ to text generation has unique challenges including (*i*) an infinite number of possible generations, which complicates the normalization of the uncertainty scores, (*ii*) potentially multiple correct answers with different surface forms (Kuhn et al., 2023), (*iii*) need to aggregate uncertainties across multiple interdependent predictions corresponding to generated tokens (Zhang et al., 2023), (*iv*) generated tokens not contributing to uncertainty equally, as some tokens represent auxiliary words (Duan et al., 2024), and (*v*) some sources of uncertainty being irrelevant for hallucination detection (Fadeeva et al., 2024). These challenges hinder the performance of classical unsupervised UQ techniques, and addressing them explicitly in a single method is quite difficult. Recently, researchers have proposed automating the detection of these intricacies using machine learning. A series of supervised methods for UQ and hallucination detection has been proposed that

[1] https://anonymous.4open.science/r/llm-uncertainty-head-24DD

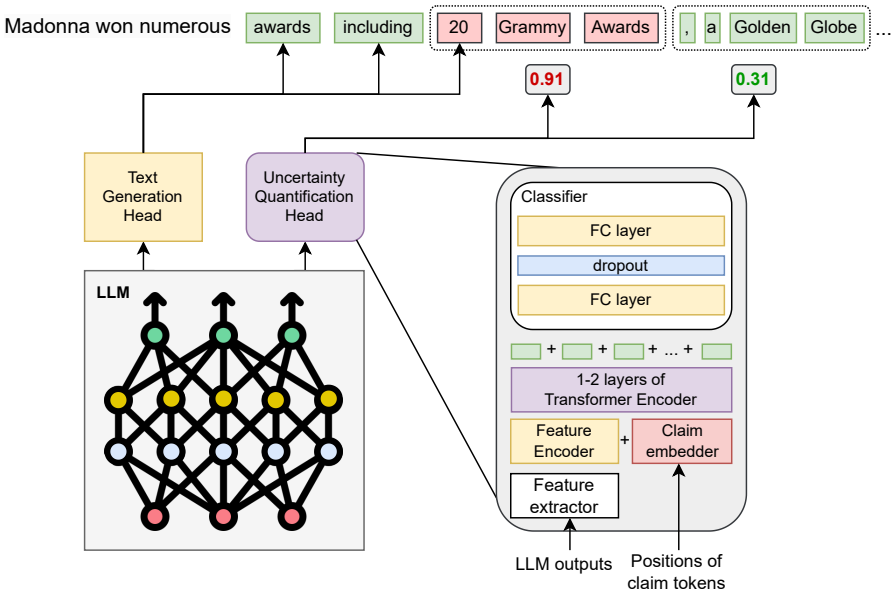

Figure 1: The architecture of uncertainty quantification heads. The example represents a text generated using an LLM, containing the hallucination *20 Grammy Awards* highlighted in red.

learn the aforementioned intricacies from the annotated data (Azaria and Mitchell, 2023; He et al., 2024b; Chuang et al., 2024).

We continue this line of work by introducing pre-trained UQ heads: supervised auxiliary modules for LLMs that substantially enhance their ability to capture uncertainty compared to unsupervised UQ methods. Their strong performance stems from the powerful Transformer architecture in their design and informative features derived from LLM attention maps. These heads do not require re-training of the entire LLM and do not alter its outputs. Despite their high performance, these methods maintain a relatively small memory and computational footprint, ensuring practical usability.

Experimental evaluation shows that our uncertainty heads are highly robust and achieve state-of-the-art performance in claim-level hallucination detection across both in-domain and out-of-domain prompts, outperforming other supervised and unsupervised techniques. Moreover, these modules demonstrate strong generalization to languages they were not explicitly trained on.

Training uncertainty quantification heads requires annotated hallucinations in LLM outputs. For constructing training data, we created an automatic pipeline for annotation of hallucinations of LLM outputs, which allows us to scale our experiments and to pre-train uncertainty heads for various LLMs. We release a collection of pre-trained UQ heads for popular open-source instruction-following LLMs, including Llama series (Dubey et al., 2024), Gemma 2 (Team et al., 2023), and Mistral-v0.2 (Jiang et al., 2023a).

The contributions of this work are as follows:

- We design a pre-trained uncertainty quantification head: a supplementary module for an LLM that yields substantially better performance for claim-level hallucination detection than classical unsupervised UQ methods and state-of-the-art supervised techniques.
- We conduct a vast empirical investigation and find that uncertainty heads show good generalization across various domains and languages. We also compare various feature sets used for building supervised UQ modules.
- We build and release a collection of pre-trained uncertainty quantification heads for popular series of open-source instruction-tuned LLMs: Llama, Gemma 2, Mistral. These modules could be seamlessly integrated into text generation code and be used as off-the-shelf hallucination detection tools.

2 Related Work

Unsupervised methods. UQ for LLMs has recently emerged as a prominent topic in NLP. This area has experienced a surge of work, with early efforts focusing on unsupervised techniques such as information-based approaches (Kuhn et al., 2023), density-based scores (Vazhentsev et al., 2022), self-consistency methods (Lin et al., 2023; Zhang et al.,

2024a), and verbalized (reflexive) strategies (Tian et al., 2023). While unsupervised approaches have shown some potential, they still fall short of offering a strong solution to the problem of LLM hallucinations (Vashurin et al., 2024).

Supervised methods. Recently, researchers have started exploring supervised UQ methods that leverage the internal states of LLMs during generation as features (Azaria and Mitchell, 2023; Slobodkin et al., 2023; Su et al., 2024; CH-Wang et al., 2024; He et al., 2024b). These methods achieve substantial performance gains over unsupervised approaches, especially for in-domain data.

Azaria and Mitchell (2023) proposed one of the first methods of this kind called SAPLMA, where they trained a perceptron using activations from various layers to detect when the LLM "agrees" with false statements. Slobodkin et al. (2023) trained a linear model on hidden states to detect question "answerability", effectively identifying unanswerable questions that typically lead to hallucinations.

Factoscope (He et al., 2024b) implemented a Siamese model with a rich feature set that incorporates activation maps, token ranks, and probabilities from unembedding matrices across layers. They reported performance improvements over SAPLMA within the training domain, but encountered challenges with generalization to other domains.

CH-Wang et al. (2024) trained simple linear and attention-based models (probes) for span-level hallucination detection, using manually annotated responses from multiple LLMs. They also tried to use synthetically generated data but found the results to be inferior to manual annotation, which limits the applicability of their approach.

Lookbacklens (Chuang et al., 2024) introduces a feature set derived from LLM attention maps. They calculate the ratio of attention weights for newly generated tokens to those in the input prompt. The ratios, computed across all attention heads and layers, are used in a linear regression model to predict an uncertainty score.

Limitations of Previous Methods. While all these works introduced a number of valuable ideas, they have notable limitations. Azaria and Mitchell (2023); Slobodkin et al. (2023); Su et al. (2024) focused on sequence-level methods and are not able to detect sub-sentence hallucinations. Many models, including Slobodkin et al. (2023); Azaria and Mitchell (2023); Chuang et al. (2024); Su et al. (2024) used non-contextualized architectures such as simple linear probes or multi-layer perceptron. Although He et al. (2024b) integrated a linear model with an attention mechanism and CH-Wang et al. (2024) used a contextualized model combining convolutions, ResNet, and GRU, these architectures are considered outdated and exhibit limitations in quality or computational efficiency. The features of the majority of models included only hidden states across layers (Azaria and Mitchell, 2023; Slobodkin et al., 2023; CH-Wang et al., 2024; Su et al., 2024), which limits their generalization. Only He et al. (2024b) and Chuang et al. (2024) performed more elaborate feature engineering. Finally, synthetic data that is leveraged through enforced decoding is used in some work (Azaria and Mitchell, 2023; Slobodkin et al., 2023). Compared to the native outputs generated by LLMs, such data may introduce additional biases and adversely affect the performance of hallucination detectors.

In contrast, here we aim to build uncertainty quantification heads for subsentence hallucination detection: on the level of atomic claims that leverage all the strengths of the aforementioned work and address their limitations: (*i*) instead of oversimple or outdated architectures, we build our solution on the powerful Transformer architecture, (*ii*) we investigate the importance of various features for hallucination detection, finding that the most informative features are derived from attention maps of LLMs, and (*iii*) we build an automatic pipeline for generating training data using the native LLM responses. This pipeline allows us to build training data at a larger scale and pre-train UQ heads for a range of popular LLMs.

3 Uncertainty Quantification Head

Consider the LLM $P(t_i \mid x, t_{<i})$ with L layers receiving a prompt x of length n and generating tokens $y = \{t_1, t_2, ..., t_T\}$. We also have a set of atomic claims $C = \{c_1, c_2, ..., c_K\}$, each representing a mapping to a subset of tokens in the output. Atomic claims, for example, can be extracted by another light-weight model. In this work, we formalize the claim-level uncertainty quantification task as building a function $U(c_i|x, y) \in [0, 1]$ that determines whether the claim $c_i \in C$ is a hallucination. A large value of $U(c_i|x, y)$ indicates a higher likelihood that the claim c_i is a hallucination.

Constructing a robust supervised hallucination detector, like any ML model, relies on a careful architecture design, the availability of high-quality

training data, and strategic feature selection. To build UQ heads, we combine a strong architectural solution based on self-attention with synthetic data based on native LLM outputs and a powerful feature set that leverages successful ideas from previous work on supervised UQ methods.

3.1 Background on Features for UQ and Hallucination Detection

Hidden states have been shown to serve as indicators of hallucinations in several studies (Azaria and Mitchell, 2023; CH-Wang et al., 2024). Hidden states $h(t)$ could be extracted from multiple layers of the LLM and aggregated, e.g., as a concatenation in a feature vector:

$$F_{\text{hs}}(t) = h_1(t) \circ h_1(t) \circ ... \circ h_L(t). \quad (1)$$

Lookbacklens (LBLens) Chuang et al. (2024) leverage features derived from the LLM's attention maps. The key idea is that when the model attends to the prompt, it attempts to solve the task, whereas attending to generated tokens causes it to disregard the prompt, increasing the likelihood of hallucination. The authors suggest using the so-called lookback ratio – the ratio of aggregated attention to tokens of the prompt and the generated tokens. Consider each layer of the LLM contains Q attention heads, and q is an index of a head. $A^{q,l}_{\text{context}}(t_i)$ and $A^{q,l}_{\text{gen}}(t_i)$ are the average attention weights to the input x and to the previously generated output $t_{<i}$, respectively:

$$A^{q,l}_{\text{context}}(t_i) = \frac{1}{n}\sum_{j=1}^{n} \alpha^{q,l}_{t_i,x_j},$$

$$A^{q,l}_{\text{gen}}(t_i) = \frac{1}{i-1}\sum_{j=n+1}^{i-1} \alpha^{q,l}_{t_i,t_j}.$$

Here, $\alpha^{h,l}_{t_i,t_j}$ represents the softmax-weighted attention score from token t_i to token t_j.

Then the lookback ratio of the model head q and the layer l for the token t_i is defined as follows:

$$LR^{q,l}(t_i) = \frac{A^{q,l}_{\text{context}}(t_i)}{A^{q,l}_{\text{context}}(t_i) + A^{q,l}_{\text{gen}}(t_i)},$$

$$F_{\text{LBLens}}(t_i) = \{LR^{q,l}(t_i)\}^{Q,L}_{q,l}. \quad (2)$$

Factoscope Min et al. (2023) in addition to model activations, introduced a set of features that leverage token probabilities, the similarity of token embeddings across layers, and the evolution of token ranks across layers. Commonly, given a token t_i at the position i, the LLM outputs hidden states $\{h_l(t_i)\}^{L}_{l=1}$, where the final hidden state $h_L(t_i)$ is passed through the unembedding matrix E to predict logits. Factoscope applies E to each LLM layer, obtaining a set of token probabilities on specific layer l with the highest values: $p^l_i = E\left(h_l(t_i)\right)$. Then, it extracts the probabilities of the top-m tokens from each layer l:

$$F_{\text{top-tokens}}(t_i) = \left\{\log p^l_i(t) \mid t \in \text{top}_m(p^l_i)\right\}^{L}_{l=1}. \quad (3)$$

To analyze token evolution across layers, Factoscope computes the cosine similarities between embeddings of top tokens from adjacent layers obtained by applying the unembedding matrix:

$$S^l(t_i) = \{\cos(E_{w_1}, E_{w_2}) \mid w_1 \in \text{top}_m(p^l_i), w_2 \in \text{top}_m(p^{l+1}_i)\}$$

$$F_{\text{tokens-sim}}(t_i) = \{S^l(t_i)\}^{L-1}_{l=1}. \quad (4)$$

Finally, Factoscope tracks token rank evolution across layers: $R^l(t_i) = \text{rank}[t_i, p^l_i]$, where rank indicates the position of t_i in the descending order of p^l_i values (top-ranked token receives 1). The ranks are further normalized to the range $[0, 1]$:

$$F_{\text{rank}}(t_i) = \{R^l(t_i)^{-1}\}^{L}_{l=1}. \quad (5)$$

3.2 Features for Pre-trained Uncertainty Quantification Heads

We experimented with all the aforementioned types of features and their combinations. However, we found that all of them exhibited various limitations. Hidden states encode a lot of domain-specific information, increasing the risk of overfitting. Factoscope features usually require substantial computational overhead and do not add much new information compared to hidden states. Attention features are quite powerful, but aggregation suggested in Lookbacklens results in the loss of valuable information. For our pre-trained uncertainty quantification heads, we use two groups of features.

Attention maps of the LLM. Attention seems to carry the key information about LLM uncertainty, which might be due to various reasons, including the fact that attention reflects the conditional dependency between the generation steps. For each token, we obtain the attention maps to k previous tokens from each attention head and layer and flatten them into a single feature vector:

$$F_{\text{att}} = \{\alpha^{q,l}_{t_i,t_{i-j}}\}^{n,k,Q,L}_{i,j,q,l}. \quad (6)$$

When $(i-j)$ is negative, we pad the feature vector with a zero placeholder. While considering many previous tokens that might explode the size of the feature space, we empirically found that the optimal value of k is typically very small: $2 \leq k \leq 5$. We believe that this is due to the powerful contextualized architecture of heads that leverages a transformer to automatically extract useful attention patterns across the full generated sequence.

Probability distributions of the LLM. Despite the fact that the probability distribution of an LLM might be misleading, it still carries useful information about the conditioned confidence of the LLM at the current step. This group of features consists of logarithms of the top-m token probabilities:

$$F_{\text{prob}}(t_i) = \{\log P(t \mid x, t_{<i}) \mid t \in \text{top}_m(P(\cdot \mid x, t_{<i}))\}. \quad (7)$$

We concatenate all groups of features into a token-level feature vector: $F(t) = F_{\text{att}}(t) \circ F_{\text{prob}}(t)$. Note that for the final feature set of uncertainty heads, we do not use features from the hidden states of the LLM layers: while they carry important information, they are usually domain-dependent and have less potential for generalization.

3.3 Architecture of Uncertainty Quantification Heads

The architecture of the UQ head is depicted in Figure 1. To make it versatile and powerful, we build it on a Transformer architecture. It consists of a feature size reduction neural network with two fully connected (FC) layers, a multi-layer transformer encoder, and a two-layer classification neural network. For each component, we use GELU activation functions and dropout regularization. To mark tokens as belonging to the claim being classified, we introduce an embedding matrix. Each token, depending on whether it belongs to the classified claim, receives a corresponding embedding that is summed up with the representation from the feature size reduction network. The resulting representations are fed into the transformer encoder. The outputs of the encoder are averaged and fed into the classifier. The UQ head is trained using a binary cross-entropy loss function. When we train heads, we freeze the "body" of the LLM, so that the LLM generations stay exactly the same.

4 Pipeline for Training Data Generation

The training data generation pipeline is presented in Figure 3 in the appendix. It starts with prompting the LLM to produce responses for a list of questions such as *Write a biography of person X* or *Write the history of the city Y*. We select relatively famous named entities so the task is not very hard for the model based on its parametric knowledge, while at the same time, it is not trivial, so outputs contain some hallucinated claims. We also do not use synthetically-generated hallucinations, as they introduce a bias between what the model actually generates vs. the synthetic data. The prompts for other domains can be found in Table 6.

We split the obtained responses into atomic claims using GPT-4o with the prompts from (Fadeeva et al., 2024; Vashurin et al., 2024). Each claim is then automatically classified by GPT-4o as *supported*, *unsupported*, or *unknown*. The last category is intended for general claims, for which estimating the veracity is meaningless. The claim labeling process is two-staged: in the first stage, we ask the model to provide an elaborated answer via chain-of-thought (CoT), and in the second stage, we ask it to summarize its answer into one word. The performance of this two-stage labeling is substantially better than for one-stage labeling, due to the well-known issue of lack of logical reasoning in LLMs without CoT (Wei et al., 2022).

The pipeline allows to construct relatively large-scale datasets annotated with claim-level hallucinations for various LLMs that are weaker than GPT-4o. Statistics about the training data used in our experiments are presented in Table 5.

5 Experiments

5.1 Experimental Setup

For evaluation, we used the LM-Polygraph framework (Fadeeva et al., 2023), which makes it easy to evaluate UQ for LLMs in a consistent way.

Evaluation Datasets. We constructed eight test sets of English questions designed to prompt LLMs to generate texts across various domains: *person biographies*, *cities*, *movies*, *inventions*, *books*, *artworks*, *landmarks*, and *events*. Each test set contains 100 questions, generated by prompting GPT-4o and Claude-3-Opus to output 100 famous domain items, e.g., 100 famous landmarks. An example structure of the prompts we used is presented

Method \ Test Sets	Biographies (in domain)	Cities	Movies	Inventions	Books	Artworks	Landmarks	Events
Random	0.29	0.21	0.10	0.16	0.11	0.26	0.12	0.11
MCP	0.41	0.31	0.20	0.32	0.14	0.32	0.14	0.14
Perplexity	0.36	0.23	0.17	0.23	0.14	0.34	0.13	0.12
Mean Token Entropy	0.42	0.29	0.24	0.38	0.17	0.32	0.14	0.16
CCP	0.50	0.37	0.27	0.38	0.17	0.38	0.20	0.17
SAPLMA	0.54	0.43	0.27	0.35	0.29	**0.53**	**0.35**	0.24
Factoscope	0.61	**0.47**	0.34	0.42	0.32	0.49	0.28	0.26
Lookback lens	0.56	0.45	0.25	0.39	0.26	0.46	0.26	0.29
UHead (Ours)	**0.63**	0.45	**0.39**	**0.48**	**0.36**	**0.53**	0.30	**0.30**

Table 1: PR-AUC for various UQ methods for hallucination detection of the Mistral 7B Instruct v0.2 model on English datasets. Biographies represent the in-domain dataset for supervised UQ methods.

Method \ Test Sets	Cities	Movies	Inventions	Books	Artworks	Landmarks	Events
UHead	0.45	0.39	0.48	0.36	0.53	0.30	0.30
UHead, bio + all - 1	**0.49**	**0.42**	**0.49**	**0.39**	**0.55**	**0.31**	**0.35**

Table 2: Introducing more diverse training data. UHead results are shown for two scenarios: when the UQ head is trained solely on the English biographies dataset, and when it is trained on the biographies dataset along with all other domains, excluding the test domain.

Method \ Test Set	Biographies
UHead (only hidden states)	57.3
UHead (att. + prob. + hs.)	57.7
UHead (Factoscope)	57.0
UHead (LookBack Lens)	60.9
UHead (att. + prob.) (Ours)	**62.1**

Table 3: PR-AUC scores for UQ heads trained with various feature sets on the Mistral 7B Instruct v0.2 model. Performance was evaluated using the validation set of the Biographies domain after hyperparameter tuning.

Method \ Language	English	Russian	Chinese	German
Random	0.13	0.34	0.23	0.15
MCP	0.18	0.43	0.31	0.20
Perplexity	0.14	0.40	0.29	0.15
Mean Token Entropy	0.20	0.44	0.44	0.22
CCP	0.31	0.49	0.44	0.31
SAPLMA	0.34	0.51	0.33	0.39
Factoscope	0.35	0.53	0.35	0.38
UHead (Ours)	**0.44**	**0.60**	**0.54**	**0.46**

Table 4: Performance comparison of the UQ head on different languages using the Gemma 2 9b Instruct model trained on English-only biographies data.

in Appendix B.1.[2] The labels for the test sets are obtained in the same way as the training sets: we generate responses using the LLM, automatically split the responses into atomic claims, and label them using GPT-4o.

To assess the cross-lingual generalizability of pre-trained UQ modules, we also conducted evaluation on Russian and Chinese prompts from (Vashurin et al., 2024), and additionally created a similar test set with German prompts. Test sets for each language consist of 100 biography-related questions. The statistics about all test sets are presented in Table 6.

Metrics. In the main experiments, we measured the claim-level performance of detecting invalid claims. For this purpose, we used PR-AUC, where "unsupported" claims represent the positive class.

Models. We conducted our primary experiments with Mistral 7b Instruct v0.2 (Jiang et al., 2023b) and Gemma 2 9b Instruct (Team et al., 2023).

Training the uncertainty heads and hyperparameter optimization. We trained the uncertainty heads using Adam with a linear learning rate decay and warmup. We selected the values of the hyper-parameters on the validation set of the *biographies* dataset using claim-level PR-AUC. We observed that among the important general hyperparameters are the weight of the instances with positive labels, the number of epochs, and the size of the learning rate. The best values of the hyperparameters for each of the tested models are presented in Table 7 in Appendix C.

[2]All data used for training and testing is available at <anonymized>

Baselines. We compare our method to several unsupervised baselines: Maximum Claim Probability (an adaptation of Maximum Sequence Probability for claims), Mean Token Entropy, Perplexity, and Claim Conditioned Probability (CCP) (Fadeeva et al., 2024). Additionally, we evaluated against supervised methods, including SAPLMA, Factoscope, and Lookbacklens. SAPLMA predicts token-level uncertainties using a 3-layer perceptron, and the mean uncertainty is calculated over claim-related tokens during inference. Note that both Lookbacklens and Factoscope operate at the claim level. Lookbacklens uses a Logistic Regression model trained on attention features. Our implementation of the Factoscope approach uses our Transformer-based architecture and the feature set that includes hidden states, top token embeddings with similarities, and token ranks. The values of the hyper-parameters we used for the baselines are given in Appendix C.

5.2 Results

Main results. Table 1 shows the performance of the unsupervised UQ techniques and the supervised UQ methods trained on persons' biographies for claim-level hallucination detection with Mistral 7B Instruct v0.2. For evaluating supervised methods, the domain *biographies* represents the in-domain test set and all other domains represent out-of-domain (OOD) test sets. Note that in this evaluation, both the questions and the LLM's responses across all domains are in English.

Among the unsupervised techniques, uncertainty scores based on CCP yield the best performance, confidently outperforming other methods on *biographies*, *cities*, *artworks*, and *landmarks*. Mean Token Entropy also achieves relatively good results on par with CCP on *books*, *inventions*, and *events*.

Supervised UQ methods greatly outperform unsupervised techniques on the in-domain test set. Moreover, remarkably, all considered supervised methods demonstrate substantial generalization and the ability to perform well beyond the training domain of people's biographies.

Our uncertainty head (UHead) demonstrates the best results in both in-domain and out-of-domain evaluations. For in-domain evaluation, UHead outperforms the best-unsupervised method CCP by 13 percentage points (pps) in terms of PR-AUC. The gap is also large for out-of-domain evaluation, e.g., for *books*, UHead outperforms CCP by 19 pps, for *artworks* and *movies* by 15 pps, and for *events* by 13 pps. Compared to supervised methods, UHead surpasses the closest competitor, Factoscope, by two pps for the in-domain evaluation. In OOD evaluation, it also consistently outperforms other supervised methods in most cases.

Analyzing other supervised methods, Factoscope demonstrates close performance in the in-domain evaluation and even slightly outperforms UHead for the *cities* domain by 2 pps. However, for other OOD domains, UHead retains better performance, outperforming Factoscope by up to 6 pps. We assume that the underperformance of the Factoscope baseline compared to UHead lies in the use of layer activations, which limits generalization. Another module that relies on hidden states is SAPLMA. In addition to the feature limitations, it also has architectural limitations, which further hurt its performance. Compared to UHead, it is behind by 9 pps on in-domain evaluation. For *artworks* and *landmarks*, SAPLMA shows good results, but for the majority of OOD test sets, it stays behind Factoscope and UHead. Lookbacklens also usually falls behind UHead and Factoscope; we believe that its main problem is its weak linear architecture. At the same time, we note that the feature set suggested by Lookbacklens based on attention is quite strong (see analysis of various feature sets below).

Introducing more diverse training data for UHead. Table 2 presents the results when we train uncertainty heads on *biographies* plus the data from all domains except one, which is used for OOD evaluation. In this scenario, uncertainty heads get access to bigger and more diverse training data. As we can see, for most of the domains, this yields a substantial boost in performance. For example, for *events* and *cities*, it gives 5 and 4 percentage points improvement, respectively. This is quite substantial as it improves the relative performance by around 10%. These results indicate that expanding the training data and enhancing its diversity could further increase the UQ performance, particularly in the OOD setting.

Analysis of feature sets. Table 3 presents the comparison of various feature sets in combination with the UHead architecture on the in-domain validation set. For each feature set, we perform an extensive hyper-parameter value search. We can see that all feature sets that leverage hidden states fall substantially behind attention-based features. The analysis of the validation loss dynamics shows

that this is probably due to quick overfitting. Models that leverage hidden states start overfitting after 1–3 epochs, while models that leverage attention might not overfit even after 10 epochs. We also note that Lookbacklens features combined with the UHead architecture provide strong performance. However, simple attention maps without feature engineering yield even better results.

Cross-lingual generalization. Table 4 presents the results for Gemma 2 9b Instruct. In this experiment, we train UQ modules on the English person's *biographies* as in the previous experiment, but we evaluate the performance on other languages. Surprisingly, UHead achieves strong cross-lingual generalization. For all OOD languages, UHead achieves substantial improvements over the best-unsupervised methods. For Chinese, UHead is better than MTE by 10 pps, for Russian, it is better than CCP by 16 pps, and for German by 20 pps. Notably, other supervised methods also demonstrate some level of generalization but have substantially worse performance. Overall, these results show that uncertainty heads, even if they are pre-trained on English data, can be good off-the-shelf hallucination detectors for LLM outputs in other languages.

Computational efficiency. Next, we evaluated the computational overhead of various UQ methods. To ensure a fair comparison, we focused only on the time required to generate texts and to compute uncertainty scores, excluding the time spent on claim extraction. The results were obtained using a multi-domain dataset containing 800 texts and a total of 18,852 claims and Mistral 7B Instruct v0.2.

Table 8 summarizes the results and provides the memory footprint of various methods. MCP and Perplexity incur no additional overhead, serving as baselines for comparison. The proposed UHead method introduces less than 10% overhead, only slightly higher than the best-unsupervised method CCP (8.6%). With around 20 million parameters, UHead has a minimal impact on GPU memory footprint (80MB). Thus, UHead is a very lightweight addition to multi-billion-parameter LLMs and is practical for real-world deployment.

6 Collection of Pre-trained Uncertainty Heads for Popular LLMs

Finally, we pre-trained a collection of Uncertainty Quantification (UQ) Heads for a range of popular 7B–9B parameter LLMs, including Mistral, various versions of LLaMA, and Gemma 2. In addition to model-level UQ, we release token-level UQ heads that can provide uncertainty scores directly for tokens without explicit claim annotation, which enables broader applicability across tasks.

```python
from transformers import AutoModelForCausalLM,
    AutoTokenizer
from luh import AutoUncertaintyHead,
    CausalLMWithUncertainty

llm = AutoModelForCausalLM.from_pretrained(
    model_name)
tokenizer = AutoTokenizer.from_pretrained(
    model_name)
uhead = AutoUncertaintyHead.from_pretrained(
    uhead_name, base_model=llm)
llm_adapter = CausalLMWithUncertainty(model, uhead,
    tokenizer=tokenizer)

# tokenize text and prepare inputs ...
output = llm_adapter.generate(inputs)
```

Figure 2: Code example for using uncertainty heads.

Our UQ heads are designed for use as an off-the-shelf tool for confidence estimation in LLMs. They could be loaded from the hub using a procedure that is similar to the "from_pretrained" API in the Hugging Face Transformers library and integrated into the LLM generation procedure with an adapter. A code example demonstrating how to use the UQ heads is provided in Figure 2. Thus, UQ heads could be integrated into third-party code with minimal modifications, which makes them an easy plug-and-play solution for researchers and practitioners.

7 Conclusion and Future Work

We presented pre-trained UQ heads – supplementary supervised modules for LLMs that help to capture their uncertainty much more effectively than unsupervised UQ methods. We demonstrated that they are quite robust and deliver state-of-the-art results for both in-domain and out-of-domain prompts. They also show remarkable generalization to other languages. Inspired by their good performance, we pre-trained a collection of UQ heads for a series of popular LLMs, including Mistral, Gemma 2, and LLama. We release the code and the pre-trained uncertainty heads so they could be used as off-the-shelf hallucination detectors for other researchers and practitioners.

We see that the performance of UQ heads improves with providing more training data from diverse domains. In future work, we plan to scale up the training data and explore the limits of the supervised approach to UQ.

Limitations

Our paper assumes a correlation between uncertainty and hallucinations. Uncertainty heads cannot solve the problem when LLMs are trained to provide misinformation. In this situation, models are confident in their deceptive answers. Uncertainty heads cannot provide ideal annotation of hallucinations, as some LLMs do not have enough capacity to provide information about what they know and what they do not know. While we see generalization in uncertainty heads, we should acknowledge that, as with any other supervised method, they work best for "in-domain" data. The bias present in LLMs could also be transferred into uncertainty heads.

Ethical Considerations

Responsible Use In our work, we considered open-weight LLMs and datasets not aimed at harmful content. However, LLMs may generate potentially damaging texts for various groups of people. Uncertainty quantification techniques can help create more reliable use of neural networks. Moreover, they can be applied to detecting harmful generations, but this is not our intention.

Limited Applicability Moreover, despite that our proposed method demonstrates sizable performance improvements, it can still mistakenly highlight correct and not dangerous generated text with high uncertainty in some cases. Thus, as with other uncertainty quantification methods, it has limited applicability.

Annotation Considerations We used GPT-4o for claim extraction and their annotation. This may introduce cultural, linguistic, or other biases into the data used to train the uncertainty heads.

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.

Joris Baan, Nico Daheim, Evgenia Ilia, Dennis Ulmer, Haau-Sing Li, Raquel Fernández, Barbara Plank, Rico Sennrich, Chrysoula Zerva, and Wilker Aziz. 2023. Uncertainty in natural language generation: From theory to applications. *arXiv preprint arXiv:2307.15703*.

Sky CH-Wang, Benjamin Van Durme, Jason Eisner, and Chris Kedzie. 2024. Do androids know they're only dreaming of electric sheep? In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 4401–4420, Bangkok, Thailand. Association for Computational Linguistics.

Yuyan Chen, Qiang Fu, Yichen Yuan, Zhihao Wen, Ge Fan, Dayiheng Liu, Dongmei Zhang, Zhixu Li, and Yanghua Xiao. 2023. Hallucination detection: Robustly discerning reliable answers in large language models. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 245–255.

Yung-Sung Chuang, Linlu Qiu, Cheng-Yu Hsieh, Ranjay Krishna, Yoon Kim, and James Glass. 2024. Lookback lens: Detecting and mitigating contextual hallucinations in large language models using only attention maps. *Preprint*, arXiv:2407.07071.

Jinhao Duan, Hao Cheng, Shiqi Wang, Alex Zavalny, Chenan Wang, Renjing Xu, Bhavya Kailkhura, and Kaidi Xu. 2024. Shifting attention to relevance: Towards the predictive uncertainty quantification of free-form large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5050–5063, Bangkok, Thailand. Association for Computational Linguistics.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Ekaterina Fadeeva, Aleksandr Rubashevskii, Artem Shelmanov, Sergey Petrakov, Haonan Li, Hamdy Mubarak, Evgenii Tsymbalov, Gleb Kuzmin, Alexander Panchenko, Timothy Baldwin, Preslav Nakov, and Maxim Panov. 2024. Fact-checking the output of large language models via token-level uncertainty quantification. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 9367–9385, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Ekaterina Fadeeva, Roman Vashurin, Akim Tsvigun, Artem Vazhentsev, Sergey Petrakov, Kirill Fedyanin, Daniil Vasilev, Elizaveta Goncharova, Alexander Panchenko, Maxim Panov, Timothy Baldwin, and Artem Shelmanov. 2023. LM-Polygraph: Uncertainty estimation for language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 446–461.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR.

Jiahui Geng, Fengyu Cai, Yuxia Wang, Heinz Koeppl, Preslav Nakov, and Iryna Gurevych. 2024. A survey of confidence estimation and calibration in large

language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6577–6595, Mexico City, Mexico. Association for Computational Linguistics.

Jianfeng He, Linlin Yu, Shuo Lei, Chang-Tien Lu, and Feng Chen. 2024a. Uncertainty estimation on sequential labeling via uncertainty transmission. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 2823–2835, Mexico City, Mexico. Association for Computational Linguistics.

Jianfeng He, Xuchao Zhang, Shuo Lei, Zhiqian Chen, Fanglan Chen, Abdulaziz Alhamadani, Bei Xiao, and ChangTien Lu. 2020. Towards more accurate uncertainty estimation in text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8362–8372.

Jinwen He, Yujia Gong, Zijin Lin, Yue Zhao, Kai Chen, et al. 2024b. Llm factoscope: Uncovering llms' factual discernment through measuring inner states. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 10218–10230.

Yukun Huang, Yixin Liu, Raghuveer Thirukovalluru, Arman Cohan, and Bhuwan Dhingra. 2024. Calibrating long-form generations from large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 13441–13460, Miami, Florida, USA. Association for Computational Linguistics.

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. 2023a. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023b. Mistral 7b. *CoRR*, abs/2310.06825.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. 2023. Generating with confidence: Uncertainty quantification for black-box large language models. *CoRR*, abs/2305.19187.

Andrey Malinin and Mark J. F. Gales. 2021. Uncertainty estimation in autoregressive structured prediction. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Potsawee Manakul, Adian Liusie, and Mark Gales. 2023. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9004–9017, Singapore. Association for Computational Linguistics.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100.

Aviv Slobodkin, Omer Goldman, Avi Caciularu, Ido Dagan, and Shauli Ravfogel. 2023. The curious case of hallucinatory (un)answerability: Finding truths in the hidden states of over-confident large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3607–3625, Singapore. Association for Computational Linguistics.

Weihang Su, Changyue Wang, Qingyao Ai, Yiran Hu, Zhijing Wu, Yujia Zhou, and Yiqun Liu. 2024. Unsupervised real-time hallucination detection based on the internal states of large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 14379–14391, Bangkok, Thailand. Association for Computational Linguistics.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

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

Roman Vashurin, Ekaterina Fadeeva, Artem Vazhentsev, Akim Tsvigun, Daniil Vasilev, Rui Xing, Abdelrahman Boda Sadallah, Lyudmila Rvanova, Sergey Petrakov, Alexander Panchenko, et al. 2024. Benchmarking uncertainty quantification methods for large language models with lm-polygraph. *arXiv preprint arXiv:2406.15627*.

Artem Vazhentsev, Gleb Kuzmin, Artem Shelmanov, Akim Tsvigun, Evgenii Tsymbalov, Kirill Fedyanin, Maxim Panov, Alexander Panchenko, Gleb Gusev,

Mikhail Burtsev, Manvel Avetisian, and Leonid Zhukov. 2022. Uncertainty estimation of transformer predictions for misclassification detection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8237–8252, Dublin, Ireland. Association for Computational Linguistics.

Artem Vazhentsev, Gleb Kuzmin, Akim Tsvigun, Alexander Panchenko, Maxim Panov, Mikhail Burtsev, and Artem Shelmanov. 2023. Hybrid uncertainty quantification for selective text classification in ambiguous tasks. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11659–11681, Toronto, Canada. Association for Computational Linguistics.

Yuxia Wang, Daniel Beck, Timothy Baldwin, and Karin Verspoor. 2022. Uncertainty estimation and reduction of pre-trained models for text regression. *Transactions of the Association for Computational Linguistics*, 10:680–696.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Ji Xin, Raphael Tang, Yaoliang Yu, and Jimmy Lin. 2021. The art of abstention: Selective prediction and error regularization for natural language processing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1040–1051, Online. Association for Computational Linguistics.

Caiqi Zhang, Fangyu Liu, Marco Basaldella, and Nigel Collier. 2024a. LUQ: Long-text uncertainty quantification for LLMs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5244–5262, Miami, Florida, USA. Association for Computational Linguistics.

Caiqi Zhang, Ruihan Yang, Zhisong Zhang, Xinting Huang, Sen Yang, Dong Yu, and Nigel Collier. 2024b. Atomic calibration of llms in long-form generations. *Preprint*, arXiv:2410.13246.

Tianhang Zhang, Lin Qiu, Qipeng Guo, Cheng Deng, Yue Zhang, Zheng Zhang, Chenghu Zhou, Xinbing Wang, and Luoyi Fu. 2023. Enhancing uncertainty-based hallucination detection with stronger focus. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 915–932.

Xuchao Zhang, Fanglan Chen, Chang-Tien Lu, and Naren Ramakrishnan. 2019. Mitigating uncertainty in document classification. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3126–3136, Minneapolis, Minnesota. Association for Computational Linguistics.

A Training Data Generation Pipeline

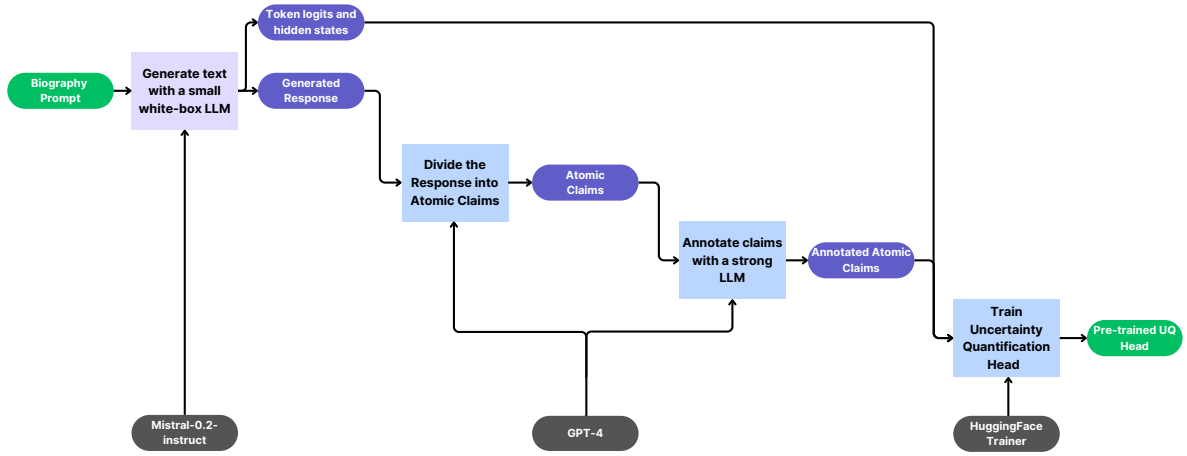

Figure 3: The training data generation pipeline.

B Dataset Details

B.1 Dataset Construction

We used few-shot learning to better guide the LLM to generate the items for the desired domain. The structure of the prompts looks as follows:

```
Continue the list of 100 most famous {domain items}:

1. <domain-item-1>

2. <domain-item-2>

3. <domain-item-3>
```

Example for the "cities" domain:

```
Continue the list of 100 most famous cities:

1. Paris, France

2. Amsterdam, Netherlands

3. Osaka, Japan
```

For claim extraction and their annotation, we use GPT-4o with prompts from (Fadeeva et al., 2024). Overall expenses for LLM API calls are approximately $4000.

B.2 Dataset Statistics

Table 5 presents the statistics of the datasets used for training and validation; Table 6 shows the statistics of the datasets used for testing.

Model	Dataset	# of texts	# of claims
Mistral 7b Instruct v0.2	biographies	3300	57,671
	multi-domain	700	14,554
Gemma 2 9b Instruct	biographies	3300	83,716

Table 5: Statistics about the training datasets used in our experiments.

Split	# of prompts	GPT-4 prompt used to generate questions	Testing prompt	# of claims	
				Mistral	Gemma
persons	100	Tell me a list of 100 most famous persons.	Tell me a bio of a <person>	2234	2857
cities	100	Tell me a list of 100 most famous cities.	Tell me a history of a <city>	2128	2684
movies	100	Tell me a list of 100 most famous movies.	Tell me about the movie <movie> and its cast.	2568	3121
inventions	100	Tell me a list of 100 most important inventions.	Tell me about the invention of <invention> and its inventor.	2269	2626
books	100	Tell me a list of 100 most famous books.	Tell me about the book <book> and its author.	2530	3070
artworks	100	Tell me a list of 100 most famous artworks.	Tell me about the artwork <artwork> and its artist.	2464	2873
landmarks	100	Tell me a list of 100 most famous landmarks.	Tell me about the landmark <landmark>.	2365	2566
events	100	Tell me a list of 100 most significant historical events.	Tell me about <event> event.	2294	2665
Russian	100	—	Расскажи биографию <person>	—	3572
Chinese	100	—	介绍一下<person>	—	2248
German	100	—	Erzhlen Sie mir eine Biografie von <person>	—	2815

Table 6: The statistics of the multi-domain test dataset and number of claims generated my Mistral 7B Instruct v0.2 and Gemma 2 9b Instruct models.

C Hyperparameters

Method	Model	Learning Rate	Num. Epochs	Weight Decay	Dropout rate	Hidden state layers	Attention window size
SAPLMA	Gemma 2 9b Instruct	1e-4	10	0.1	0.1	[-1]	–
	Mistral 7b Instruct v0.2	1e-4	10	0.1	0.1	[-1]	–
Lookbacklens	Gemma 2 9b Instruct	1e-2	13	0.1	0.1	–	–
	Mistral 7b Instruct v0.2	1e-2	13	0.1	0.1	–	–
UHead (Factoscope)	Gemma 2 9b Instruct	2e-4	3	0.1	0.2	[-1]	–
	Mistral 7b Instruct v0.2	2e-4	3	0.1	0.2	[-1,-15]	–
UHead	Gemma 2 9b Instruct	2e-4	6	0.1	0.05	–	2
	Mistral 7b Instruct v0.2	1e-4	10	0.1	0.1	–	5

Table 7: Optimal hyperparameters for each method and model.

For each tested model, we selected hyperparameters by optimizing the PR-AUC metric on the validation set of the "biographies" dataset. In training, we optimized the learning rate, warmup ratio, number of epochs, and the weight of positive examples in the cross-entropy loss. For the model architecture, we optimized the number of uncertainty layers, the number of heads, and the intermediate dimension. For feature extraction, we optimized the number of layers used to obtain hidden states, token probabilities, and attention weights, as well as the number of preceding tokens considered for attention. The optimal hyperparameters are summarized in Table 7. The hyperparameter grid is the following:

Learning rate: [1e-5, 3e-5, 5e-5, 1e-4, 2e-4, 5e-4, 1e-2];
Num. of epochs: $\{n \in \mathbb{N} \mid 2 \leq n \leq 15\}$;
Hidden state layers: [[-1], [-1, -16], [-1, -15, -30]];
Attention window size: [1, 2, 3, 4, 5, 10];
Dropout rate: [0., 0.05, 0.1, 0.2];
Weight decay: [0, 1e-2, 1e-1].

D Hardware and Computational Efficiency

All experiments were conducted on 8 NVIDIA RTX 6000 Ada GPUs. On average, training a single model with hyperparameter search takes around 150 GPU hours.

Method	Computational Overhead	GPU Memory Footprint, MB
MCP	0.0 %	0
Perplexity	0.0 %	0
Max Token Entropy	0.2 %	0
CCP	8.6 %	440
SAPLMA	4.7 %	5
Factoscope	122.1 %	70
UHead + Lookback Lens	18.4 %	55
Lookback Lens	17.0 %	<1
UHead	9.4 %	80

Table 8: Computational overhead of UQ methods using the Mistral 7B Instruct v0.2 model. Overhead is measured relative to the fastest method MCP.