A STATISTICAL MEASURE OF TRUST IN GENERATIVE AI

Anonymous authors
Paper under double-blind review

ABSTRACT

Trustworthiness is a crucial issue for generative AI, as modern models can produce convincing and confident responses even when their outputs are incomplete, unsupported, or wrong. The frequent absence of ground-truth knowledge or labels complicates verification of generative AI outputs. To address this issue, we propose a trustworthiness framework grounded in model-internal confidence calibrated against an external expert reference. We estimate **per-token trustworthiness** of a student generative model as the statistical typicality of each generated token relative to an expert model's distribution, where the expert is used as a distributional reference rather than as a source of factual verification labels. To this end, we introduce an **AI ontology calibrator** that leverages the student model's internal latent features and token-level output statistics to compute expert-reference acceptance scores. Combined with traditional token-level confidence, entropy, and perplexity statistics, calibrator scores yield a compact **set of** 15 **response-level indicators** for correctness classification. Evaluated across BF program generation, mathematical problem solving, and medical question answering, combined calibrator and traditional indicators provide complementary reliability signal which could be used to build detectors of untrustworthy outputs and hallucinations. Remarkably, our experiments show that these detectors are transferable across domains, although transferability is not always symmetric.

1 INTRODUCTION

Generative AI has become a common component of modern computational systems, supporting coding, text processing, search, learning, and decision making. As these systems are increasingly used in consequential domains, the problem of trustworthiness becomes central. This is particularly visible for Large Language Models (LLMs), which can produce fluent and confident responses even when the underlying content is incomplete, unsupported, or wrong.

Current gold standard approaches to trustworthiness in LLMs emphasize factual incorrectness (Bazarova et al., 2025; Li et al., 2024; Ye et al., 2023), unfaithfulness to input context (Zhang et al., 2023; Cossio, 2025), or internal inconsistency (Mündler et al., 2023). They presuppose access to some notion of ground truth, such as an external knowledge base, a reference document, an objective fact, or a set of internally consistent statements (Huang et al., 2023). Dependence on ground truth introduces a practical limitation. In open-domain settings, long-tail knowledge may be unavailable or poorly documented. In domains such as medicine, correctness can be context-dependent, and accepted standards may evolve over time. These constraints motivate methods that estimate trustworthiness without requiring verified factual labels at inference time.

Our goal is circumvent this issue by showing learnability of alternative confidence-style reliability signals in a student-expert framework enabling to ensure statistical trustworthiness. There a smaller generative model constitutes a *student*. A larger *expert* model can be used offline during calibration. The expert is not treated as an oracle and does not define absolute truth. Instead, it provides a distributional reference against which the student's token-level behavior can be calibrated. At deployment time, the resulting trustworthiness indicators are computed from the student model's own internal states and token-level statistics.

To this end, we propose an **AI ontology calibrator**. By ontology we denote a task domain, an operational representation of its outputs, and evaluation protocol. The calibrator compares student

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

hidden representations and token-level output with expert-reference acceptance scores. The scores estimate if a generated token agrees with the expert reference under chosen alignment criteria.

We evaluate the framework on three ontologies of increasing complexity. In *BF program generation*, correctness can be verified by executing a generated program and comparing its output to the requested behavior. In *mathematical reasoning*, correctness is evaluated using reference-based and ProcessBench-style labeling protocols, reflecting the fact that multiple solution paths may be valid. In *medical multiple-choice question answering*, we evaluate several student LLMs using response-level labels obtained through an automated annotation pipeline, and interpret detector outputs as reliability or routing signals rather than clinical validation. Our theoretical analysis gives a finite-sample justification for the calibration step. Empirical results show that calibrator-derived indicators provide information complementary to standard confidence features.

Our specific contributions are as follows:

1. We frame trustworthiness estimation through the lens of confidence, distinguishing it from uncertainty and consistency. Complementary to approaches looking for factual errors, we treat unreliable generations as outputs whose token-level internal signals are atypical relative to an expert-reference calibration criterion.
2. We propose *an AI ontology calibrator* – a tool for estimating token-level trustworthiness of a student generative model using an expert model as an offline distributional reference.
3. As part of the AI ontology calibrator, we introduce expert-reference acceptance scores: confidence-style token-level statistics grounded in the typicality of generated tokens relative to the expert reference. These scores are supported by bounds on threshold-defined acceptance and rejection probabilities.
4. We construct a small set of 15 response-level indicators by aggregating calibrator scores with traditional token-level confidence, entropy, and perplexity statistics. These indicators can be used to train supervised classifiers for discriminating correct and incorrect responses under ontology-specific labeling protocols.
5. We evaluate the proposed indicators across BF program generation, mathematical problem solving, and medical question answering. The combined calibrator and traditional indicators provide complementary reliability signal, with the clearest gains in Medicine and BF and more mixed behavior in Math.
6. We analyze transfer across models within the same ontology and across different ontologies. The results show selected improvements from calibrator-derived features, but transfer remains heterogeneous rather than universal.

2 Related Work

An overview of classical approaches such as consistency-based, retrieval-based, and model-based are discussed in Appendix A. Since our method studies trustworthiness through the lens of confidence, we concentrate here on confidence-based and internal-signal methods.

Low-confidence validation methods monitor minimum token probabilities, token entropy, or related scalar statistics to flag candidates for external verification (Varshney, 2023). Length-Normalized Token Probability (LNTP) (Bouchard & Chauhan, 2025) is a representative example: it aggregates token probabilities into a length-normalized sequence-level score, providing a strong and efficient baseline that avoids multiple generations. However, such approaches usually rely on raw intrinsic confidence signals and remain vulnerable to miscalibration, especially after instruction tuning, where aleatoric ambiguity and epistemic ignorance can become conflated (Tao et al., 2025).

To better isolate *epistemic uncertainty*, ensemble-based estimation has been proposed. Input clarification ensembling generates K semantic rephrasings, while Monte Carlo Dropout approximates Bayesian posterior variance via T stochastic forward passes (Hou et al., 2023; Gal & Ghahramani, 2016; Zeevi et al., 2024). These methods can provide richer uncertainty estimates, but they incur linear overhead in the number of samples or stochastic passes and typically operate at the sequence or concept level rather than as token-level reliability scores available from a single generation.

Beyond scalar probability and entropy measures, hidden-state and logit-based methods aim to use richer internal model information. *INSIDE* (Chen et al., 2024) and related probing approaches leverage hidden states across transformer layers via eigenvalue analysis or learned classifiers, while energy-based methods such as Semantic Energy use pre-softmax logits to mitigate entropy saturation (Ma et al., 2025). These methods show that internal representations contain reliability-relevant information, but they are often framed as out-of-distribution or uncertainty detectors and do not directly calibrate token-level student behavior against an external expert reference.

Our work occupies a complementary position. Like confidence-based methods, it uses model-internal signals and can produce lightweight indicators from a single student generation. Unlike raw confidence proxies, however, the proposed AI ontology calibrator uses an expert model offline as a distributional reference to calibrate student hidden states and token-level statistics. The resulting expert-reference acceptance scores are not factuality guarantees, but calibrated reliability features that can be aggregated with traditional confidence, entropy, and perplexity-style statistics for supervised response-level correctness prediction.

3 Overall Approach

In what follows, a generative model whose trustworthiness is being assessed will be referred to as a **student**. We assume that the student's generation process can be decomposed into individual steps and at each step the student generates a **token**. Since we focus on LLMs, tokens include whole words, subword units, or single characters as determined by the underlying LLM tokenizer.

At the core of our approach is the observation that a statistically grounded measure of a student model's token-level behavior, relative to what an expert would produce in the same context and ontology, correlates with human judgments of the quality and factuality of the student's overall response (Yuan et al., 2021; Fu et al., 2024). We build on this by treating such a measure as a signal, among others, for detecting whether a response can be trusted.

In order to test and assess our method, we consider three ontologies of increasing complexity, **BF**, **Math**, and **Medicine**, and their corresponding trustworthiness/correctness definitions:

- **BF program generation (BF)**: a student-LLM generated program is considered trustworthy if it compiles, executes, and produces correct intended output.
- **Math problem solving (Math)**: a generated solution is considered trustworthy if it correlates with reference solution according to an agreed metric. Reference solution constitutes solution provided by an external ***expert***, a human or another AI system. Technical details explaining how we compared student's and expert's responses are provided in Section 4.3.
- **Medical question answering (Medicine)**: student LLMs answer multiple-choice questions with all reasoning traces, and ***expert*** responses are taken as a reference. In this paper, we treat all inaccuracies as equally important. A promising direction for future research is to introduce semantic-context awareness, where errors are weighted by their clinical significance. A variant of labeling accounting for context criticality is presented in Appendix C.3.

Before diving into details of the method, we want to distinguish between three related but fundamentally different concepts for evaluating statistical trustworthiness of a model:

- **Uncertainty** is a *problem-level* property that characterizes the spread of the answer space for a given prompt, regardless of any particular output.
- **Consistency** measures *behavioral stability* across multiple outputs, requiring at least two generations or inputs to compare.
- **Confidence**, by contrast, is an *instance-level* property of a *single prediction* – it reflects how strongly internal or expressed signals point to correctness for that specific answer.

Our framework is specifically designed to estimate trustworthiness through a *single generation pass* and with no ground truth information required. This contrasts with consistency measures which require at least two outputs or inputs to be compared. Regarding remaining two measures, we prefer confidence over uncertainty for two complementary reasons. First, uncertainty is a *prompt-level*

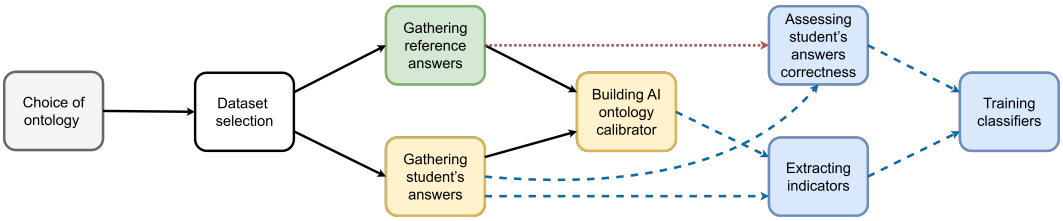


Figure 1: Our workflow. **Solid lines** indicate the necessary steps for building an AI ontology calibrator; **dashed lines** correspond to the proposed procedure for obtaining indicators and evaluating how well they reflect response correctness; the **dotted line** highlights the optional use of reference answers for labeling the student's correctness for classifier training.

property shaped by factors external to the prediction itself such as training data distribution, prompt ambiguity, and the overall spread of the answer space. Confidence, by contrast, is an *instance-level* signal reflecting the model's internal conviction about a specific generated element, independently of these distributional factors and without any knowledge of the model's training data. Second, in many real-world ontologies, ground truth is neither static nor universally accessible: medical treatment protocols are revised as evidence accumulates, scientific consensus shifts, and accepted best practices change over time. By grounding our framework in the model's own internal confidence signals, we obtain a formulation that remains feasible in absence of external ground truth stability.

4 METHODOLOGY

Our framework comprises two components: (1) measuring **statistical confidence of each generated token**, relative to that of expert's through a single generation pass without accessing expert's own confidence, uncertainty and state, and without ground truth information; and (2) determining whether the **generated response as a whole is trustworthy**, by aggregating per-token indicators that combine our proposed statistical confidence scores with traditional measures. The resulting workflow is illustrated in Figure 1, whose components are described in detail in the following sections.

4.1 FRAMEWORK COMPONENT 1: AI ONTOLOGY CALIBRATOR

The AI ontology calibrator's aim is to assess each student token's typicality with respect to the conditional distribution induced by the reference answer. The process of its construction is further illustrated with **solid lines** in Figure 1. Reference answers are responses provided by ***expert*** models – SOTA models of much larger size than the student models. For each token position we learn token-conditional generation probability with expert's response taken as a reference.

More formally, let G be a generative AI model mapping inputs $u \in \mathcal{U}$ to finite output sequences $(T_1, T_2, \ldots, T_m)$, $m \in \mathbb{N}$, where m is the length of generated sequence and T_i represents a token. In this paper, we consider the map G represents an LLM, while our method can be implemented to other generative models. Let us denote the set of possible tokens by $\mathcal{T} = \{\tau_1, \ldots, \tau_q\}$. Since output sequences are generated sequentially, generation decomposes into individual steps. For each step of the generation, G generates many objects, including **the vector of probabilities for all possible tokens** and **the vector of features** $\phi = \Phi(\cdot)$. Here we use outputs of the last fully connected layer before the heads as features, albeit other model variables could be used too.

The response generation process of an AI model proceeds as follows. Let us denote input u corresponding to generation of the first token as $u_1 = u$. Output of the first step of generation is a q dimensional vector of probabilities s_1, and the vector of features is $\Phi(u) \in \mathbb{R}^d$, where d is the dimension of the last fully connected layer before the heads. Based on the values of s_1, the system selected one token T_1. At the next step, the model's input becomes $u_2 = u \cup \{T_1\}$, and the process repeats. As a result, generation of a single token in the output sequence can be modelled as a map from the set of inputs $\mathcal{U}$ (inputs are different for different steps) to a finite set of outputs $\mathcal{T}$:

$$G : \mathcal{U} \to \mathcal{T}. \tag{1}$$

To distinguish different steps of generation, input at the k-th step is denoted as $u_k = u \cup_{j=1}^{k-1} \{T_j\}$.

In what follows student model will be denoted as G^S, and expert (a model judge or a human) will be referred to as G^E. In response to the same initial input these models produce two sequences of tokens: $\left(T_1^E, \ldots, T_{m_E}^E\right)$ (expert) and $\left(T_1^S, \ldots, T_{m_S}^S\right)$ (student). For the student, we assume access to its vector of features $\Phi(u_j)$. For the expert model, this is not required.

Our goal is to estimate the probability of correctness of token T_i^S. Token T_i^S is deemed correct if $T_i^S = T_i^E$, and it is considered incorrect otherwise. If $m_E < m_S$, then all tokens T_i^S with $i > m_E$ are deemed wrong. This enables posing the task of identifying "correct" generations as a classical classification problem at the level of each individual token τ_i: we need to find a map accepting or rejecting tokens produced by G^S with some inferrable confidence. This map could formally be defined as:

$$\mathcal{A}^i : \mathcal{T} \times \mathbb{R}^d \to \mathcal{T} \cup \{\tau^\times\},$$

where $\tau^\times$ is the label corresponding to rejected tokens.

As a measure of model G^S confidence relative to expert's response we consider conditional probabilities (or suitable proxies of):

$$P(\mathcal{A}^i(G^S(u_k), \Phi(u_k)) = \tau_j | G^S(u_k) = \tau_j, G^E(u_k) = \tau_j),$$

where $\mathcal{A}^i$ is map discussed above, and k is the token's position. Low values of these probabilities indicate disagreement between G^S's outputs and the target distribution, and therefore insufficient trust. We also can estimate the probability of the opposite event that token τ_j produced by G^S is in position k does not match that of G^E:

$$P(\mathcal{A}^i(G^S(u_k), \Phi(u_k)) = \tau^\times | G^S(u_k) = \tau_j, G^E(u_k) \neq \tau_j).$$

If the distribution underpinning the above data generation process would have been known then these probabilities could be computed. However, in most practically-relevant instances, this is not the case, and hence we need to learn or estimate the corresponding probabilities from empirical data.

In this work, we introduce a tool to estimate these probabilities which we are referring to in the text as an **AI ontology calibrator**. To form/build the calibrator, we assume access to a dataset, which we will refer to as the **forming dataset**:

$$\mathcal{S} = \{(u_i, \tau_i) \ : \ u_i \in \mathcal{U}, \ \tau_i \in \mathcal{T}, \ i = 1, \ldots, M\}. \tag{2}$$

The forming dataset $\mathcal{S}$ is further partitioned into $\mathcal{S}_j = \{(u, \tau) \in \mathcal{S} \ : \ \tau = \tau_j\}$, where each such set is non-empty. For each $j = 1, \ldots, q$, partition $\mathcal{S}_j$ into:

$$\mathcal{S}_{+,j} = \{(u, \tau) \in \mathcal{S}_j \mid \tau = \tau_j\}, \ \mathcal{S}_{-,j} = \{(u, \tau) \in \mathcal{S}_j \mid \tau \neq \tau_j\},$$

that are assumed not empty, with cardinalities $M_{+,j}$ and $M_{-,j}$. Let $\mathcal{H} = \{h_1, \ldots, h_q\}$ be a class of measurable projection functions $h : \mathbb{R}^d \to \mathbb{R}$. Each function h_j defines empirical cumulative distribution functions:

$$F_{+,j}(s) = \frac{1}{M_{+,j}} \sum_{(u,\tau) \in \mathcal{S}_{+,j}} \mathbf{1}_{h_j(\Phi(u)) \leq s}, \ \ F_{-,j}(s) = \frac{1}{M_{-,j}} \sum_{(u,\tau) \in \mathcal{S}_{-,j}} \mathbf{1}_{h_j(\Phi(u)) \leq s}, \tag{3}$$

and $\mathbf{1}_D$ is the indicator function of the set D. Detailed theoretical background and justification of the calibrator's capability to learn desired probability estimates from empirical data are presented in Appendix B.

In general, we have several attributes for each token: the output token itself $G^S(u_j) = T_j$, the vector of output probabilities (softmaxes in standard models) $(s_1^j, \ldots, s_q^j)$, and the vector of features $\Phi(u_j) = (\phi_1^j, \ldots, \phi_n^j)$, j is the position in the generated sequence. The vectors ϕ^j are used as feature vectors $\Phi(u)$ for the calibrator. These vectors could be used 'as is' without any pre-processing. For each token i with rich enough forming set (in this study we consider only tokens that occur at least $1,000$ times in the dataset) we computed three arrays: (1) projection functions $h_j \in \mathcal{H}$, which for simplicity have been chosen as functionals projecting data onto normalized direction of the corresponding Fisher discriminants; (2) thresholds of acceptance th_i^j (score representation of $th_i^j = F_{-,j}^\dagger(\Delta_j^i)$), where $\Delta_j^i \in (0, 1)$; (3) the vector of the adjusted lower bounds on the probabilities of acceptance p_i^j for the corresponding threshold th_i^j.

Calculation (estimation) of the probability of correct acceptance of the token τ_j with feature vector ϕ can be computed as follows. Calculate dot product of feature vector with corresponding projection function $r = (h_\tau, \phi)$. Define the maximal k, such that $th_\tau^k < r$. Take $p_\tau = p_\tau^k$ as answer. For each generated token T_j, compute the following quantities:

1. the probability of correct acceptance p_j (or its lower bound), hereafter referred to as **calibrator probabilities**;
2. the maximum softmax probability of the generated token $s^j = s_1^j$ (where $s_1^j, ..., s_q^j$ are sorted in the descending order), i.e., the probability assigned to the model's top-predicted token at position j, hereafter referred to as **softmaxes**;
3. the Shannon entropy of the predictive distribution $e^j = -\sum_{i=1}^q s_i^j \ln(s_i^j)$, hereafter referred to as **entropies**;
4. the log-probability of the next generated token under the model's predictive distribution at position j, $l^j = \ln P(T_{j+1} \mid T_1, \ldots, T_j) = \ln s^j_{\mathrm{rank}(T_{j+1})}$, where $\mathrm{rank}(T_{j+1})$ indicates the position of the actual token T_{j+1} in the sorted softmaxes list; hereafter referred to as **token log-probabilities**.

Quantities (1)-(3) are collectively used to construct a set of **indicators** for training trustworthiness classifiers, while quantities (4) are utilized for **baseline indicators (baselines)** calculation. Details on indicator construction and classifier training are provided in the following section.

4.2 Framework component 2: determining response correctness

For each generated response of an LLM, we obtain four per-token quantities described in Section 4.1. To determine trustworthiness of responses, we aggregate quantities into indicators, and train a classifier mapping indicators of each generated sequence to a probability that the response/generation is correct (trustworthy). We thus now deal with a binary classification task where labels are available. During training, responses whose probability of being correct ≥ 0.5 are considered correct (see Section 4.3 for details).

Our set of indicators for each generated sequence includes the following:

- Mean ($\bar{\cdot}$), standard deviation ($std(\cdot)$), and geometric mean ($\bar{G}.$) aggregations of calibrator probabilities (p), softmaxes (s), and entropies (e).
- Entropy-like ($E.$) and perplexity-like ($PP.$) values of calibrator probabilities and softmaxes.
- Pearson (PCC) and Spearman (SCC) correlation coefficients between calibrator probabilities and entropies.

The complete set of indicators comprises 15 quantities:

$$\{\bar{p}, \bar{s}, \bar{e},\ std(p), std(s), std(e),\ PCC, SCC,\ E_p, E_s,\ PP_p, PP_s,\ \bar{G}_p, \bar{G}_s, \bar{G}_e\}. \tag{4}$$

Among these indicators, we have two traditional confidence measures, namely, Mean Token Entropy ($\bar{e}$) and LNTP (Bouchard & Chauhan, 2025)($\bar{G}_s$), along with the newly proposed ones. Moreover, two of our indicators coincide with standard baselines under greedy decoding: Mean Token Probability ($\overline{TP}$) $\approx \bar{s}$ and Geomean Token Probability ($\bar{G}_{TP}$) $\approx \bar{G}_s$. Note also that LM Perplexity ($LM\ PP$) $= 1/\bar{G}_{TP}$. We also consider MSP, Max Token Entropy ($\max(TE)$), and Min Token Probability ($\min(TP)$) as baselines. A diagram illustrating computation of all provided indicators and baselines based on quantities from Section 4.1 is shown in Figure 2.

4.3 Specifics of our framework and the considered ontologies

For math, we use the first $5,000$ problems to build the calibrator and train classifiers in the following settings: (V1) concatenate the first $5,000$ problems with the second $5,000$ problems and train classifiers via 20-fold cross-validation; and (V2) train classifiers on the first $5,000$ problems and test them on the second $5,000$ problems. As Figure 1 shows, using reference answers when labeling students' responses is not always necessary. For math, we compare two labeling strategies: (1) **LLM-Judge** labeling with Qwen2.5-32B, which uses expert answers, and (2) **ProcessBench** labeling

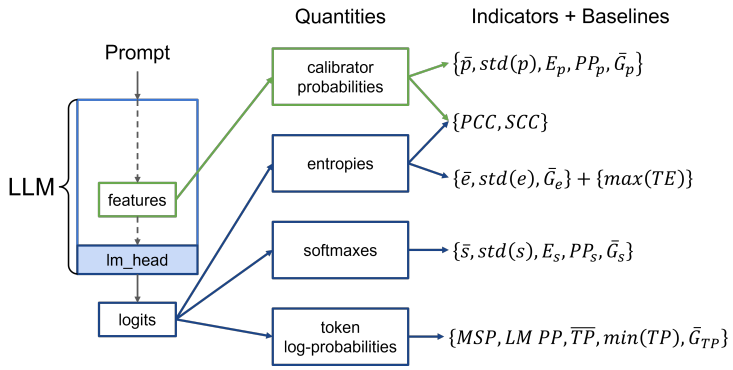


Figure 2: Sources of derivation of quantities, indicators, and baselines. Entropies provide both indicators and baselines, token log-probabilities – only baselines, calibrator probabilities and softmaxes – only indicators. **Green arrows** highlight indicators produced by our proposed calibrator probabilities, **blue arrows** – traditional indicators. $\{PCC, SCC\}$ – indicators of mixed category: to calculate them, both calibrator probabilities and traditional indicators are needed.

Table 1: Experimental setups, naming convention, and the statistics on the built calibrators and the classifier data for all ontologies. In the classifiers category, we indicate the total number of considered responses with labels of correctness (denoted as "Total"), the number of correct ("1s") and incorrect ("0s") responses. "-" means the absence of an experiment for an ontology.

Ontology		BF	Math		Medicine			
Data for building calibrators		Custom 433,688 pairs of (program, output)	Open-source 5,000 pairs of (problem, solution) – batch 1		Open-source 12,369 pairs of (problem, answer)			
Expert		BF compiler	DeepSeek-R1		GPT-OSS-120B			
Student (N params) if not mentioned in model name		Custom Transformer (22M)	Qwen3-8B		MedGemma (4B)	Qwen3-8B	Qwen3-0.6B	Qwen3.5-0.8B
Unique tokens		8	18,089		38,926	29,481	27,783	29,826
Unique calibrated tokens		8	808		851	1,061	1,058	1,010
Classifiers	20-fold cross-validation (V1)	Total: 10,000 1s: 3,818 0s: 6,182	Two concatenated batches (batch 1 and batch 2) – Total: 6,845		All data from 12,369 question-answer pairs, successfully labeled with ProcessBench			
			LLM-Judge (Qwen2.5-32B) 1s: 6,453 0s: 392	ProcessBench (DeepSeek-V3.2) 1s: 3,452 0s: 3,393	Total: 5,978 1s: 3,919 0s: 2,059	Total: 3,207 1s: 2,326 0s: 881	Total: 3,663 1s: 564 0s: 3,099	Total: 1,954 1s: 595 0s: 1,359
	ID for transfer experiment	A	B	C	D	E	F	G
	Train/test (V2)	-	Train – batch 1 Total: 3,493		-	-	-	-
			1s: 3,200 0s: 293	1s: 1,029 0s: 2,464				
			Test – batch 2 Total: 3,352					
			1s: 3,253 0s: 99	1s: 2,423 0s: 929				

with DeepSeek-V3.2, which does not require expert answers. For the (V1) and (V2) settings described above, we use only those problems from the first $5,000$ and the second $5,000$ that have both LLM-Judge and ProcessBench labels. For BF and medicine, we build calibrators on the full datasets and train classifiers via 20-fold cross-validation. For BF labeling, we select $10,000$ programs and label them by compiling each program and comparing its output with the desired output: programs whose outputs match are assigned label 1 (correct), while those that do not are assigned label 0 (incorrect). For medical question answers, we apply the ProcessBench technique with GLM-4.7.

5 Experiments

Table 1 details experimental setups for all ontologies and the statistics of built calibrators. Full implementation is available in (Anonymous Authors, 2026).

Ontology 1: BF. We consider the problem of developing a program in the BF language that produces a specified output. BF is a Turing-complete language consisting of only eight possible symbols. We chose BF due to its simplicity and the straightforward process of verifying the correctness of a generated program. The dataset comprises $433,688$ program-output pairs. Programs are constrained to a maximum of 18 commands, while outputs are limited to 256 character codes. A detailed description of BF is presented in Appendix C.1.1. Details regarding the model architecture and training procedure are provided in Appendix C.1.2.

Ontology 2: Math. In this setup, we consider mathematical tasks from (Mattern et al., 2025). Since the tasks are open-ended mathematical problems requiring full reasoning traces rather than just final answers, evaluation of their correctness cannot be reduced to simple answer matching. Manual annotation of such responses would be both time-consuming and costly – particularly given that most tasks admit multiple valid solution paths. To address this, we employ an LLM to generate reference solutions (as described in Section 4.1), offering a practical balance between annotation quality and scalability. Our aim is not to establish absolute ground truth, but to validate a general method for assessing student-model quality in a regime where expert's answers can be treated as references. Specifically, as *an expert*, we use DeepSeek R1 (DeepSeek-AI, 2025a) (671B parameters), one of the strongest publicly available LLMs (Guo et al., 2025), generating a single reference solution per task, while as *a student* we use Qwen3-8B (Qwen Team, 2025), which is a relatively small LLM. To evaluate the correctness of the student's solutions, we employ two techniques. Details about labeling strategies and their comparison are provided in Appendix C.2.

Ontology 3: Medicine. We use *Medical-Reasoning-SFT-GPT-OSS-120B* dataset (Panahi, 2025). From the original collection, we sample and curate $12,369$ questions, focusing on multiple-choice items that require reasoning about human biology and medicine. To improve topical consistency, we apply a series of filters and exclude: (i) animal-related questions, (ii) questions about medical legal procedures, (iii) ecology- and taxonomy-focused questions, (iv) non-English questions, (v) plant and marine biology questions, (vi) questions centered on organizations (e.g., WHO, UNICEF), and (vii) microbiology questions (e.g., bacteria-related items). We found that a subset of slightly more than $12,000$ questions provides sufficient coverage for our experiments while keeping evaluation and analysis tractable. Most questions focus on medical conditions and diagnosis, as well as the biological processes underlying diseases and broader aspects of human biology. As the dataset has already provided the results of GPT-OSS-120B generations, we consider them as the *expert's answers*. As student LLMs, we consider four models:

- **MedGemma-1.5-4B** (Sellergren et al., 2026) is a multimodal instruction-tuned variant of Gemma 3, specifically trained on medical text and images;
- Qwens (Qwen Team, 2025; 2026): two representatives of the Qwen3 family – **Qwen3-8B** and **Qwen3-0.6B** – and one from the more recent Qwen3.5 small series, **Qwen3.5-0.8B**.

To annotate correct answers we use ProcessBench labeling (Zheng et al., 2025) with GLM-4.7.

Results: prediction of generations correctness. As described in Section 4.2, for each student from each ontology, we obtain a set of 15 indicators and six baselines: MSP, $LM\ PP$, $\max(TE)$, $\overline{TP}$, $\min(TP)$, and $\bar{G}_{TP}$. Feature correlations across all considered ontologies are provided in Appendix D.1, and classifier technical details in Appendix E. Notably, several papers (Kadavath et al., 2022; Lin et al., 2022) report a correlation between generation correctness and the model's self-assessment labels; our results on medical ontology show no such correlation (see Appendix D.2).

Table 2 presents the classifiers' results. Logistic regression was trained only on the provided cases, while MLPs were trained on all possible combinations of the 15 indicators, with only the best results reported here. The table shows that our proposed indicator set consistently outperforms traditional confidence measures. Even using features derived from calibrator probabilities alone provides reasonable quality across all ontologies. This also supports the validity of using the same data for classifier training and calibrator construction, since cross-validation relies on stratified 20-fold splitting and a weighted loss function. MLPs' results show that the best results were obtained on the indicators subsets including at least one derived from calibrators. The poor performance on Math, V2 reflects a train/test distribution mismatch rather than weaker classifier quality – particularly for ProcessBench labeling, where the class ratio is inverted between train and test.

Experiments demonstrating transferability of our trustworthiness detectors across different domains and models are provided in Appendix F.

6 Conclusion

In this work we considered the problem of assessing the trustworthiness of outputs of generative AI models. The major challenge of this task is that the ground truth data or gold-standard methods for checking the correctness of responses are not always available. Sometimes, it may not even be possible

Table 2: **Mean ± standard deviation** of balanced accuracy of Logistic Regression and MLP across 20 folds for all ontologies. For MLP, we provide only the best results with the subset of indicators used for training. For **Math, V2**, we provide **train/test** balanced accuracy. "15 indicators" denotes using all proposed indicators, whereas "12 indicators" refers to the first 12 indicators defined in Equation 4. For each ontology, top-1 scores are in **bold**, top-2 in **bold underlined**, and top-3 in ***bold italic***. We also provide results for four naive baselines: constant predictions of ones (Const = 1) and zeros (Const = 0), and random predictions where the probability of predicting 1 is either fixed at 0.5 (Rand. 0.5) or set to the majority class proportion (Rand. major).

Indicators		Medicine				BF	Math, V1		Math, V2	
		MedGemma	Qwen3-8B	Qwen3-0.6B	Qwen3.5-0.8B	Custom Transformer	LLM-Judge	ProcessBench	LLM-Judge	ProcessBench
Logistic Regression										
Individual (ours)	$\bar{p}$	0.5681±0.0299	0.5131±0.0433	0.5674±0.0403	0.4922±0.0562	***0.7179±0.1410***	0.5049±0.0730	0.4798±0.0221	0.5026/0.5220	0.5162/0.4777
	$std(p)$	0.5779±0.0329	0.5281±0.0416	0.5564±0.0396	0.5263±0.0718	0.5709±0.1358	0.5397±0.0832	0.5676±0.0674	0.5101/0.5055	0.5217/0.4691
	E_p	0.6701±0.0377	0.4857±0.0392	0.7890±0.0518	0.5311±0.0486	0.5509±0.2132	0.6511±0.0725	0.5134±0.0570	**0.6004/0.7610**	**0.5379/0.5463**
	PP_p	0.6693±0.0361	0.5708±0.0356	0.6375±0.0447	0.5154±0.0483	0.5028±0.2195	**0.6528±0.0831**	0.4985±0.0540	0.6080/0.7349	0.5390/0.5244
	$\bar{G}_p$	0.5019±0.0025	0.5098±0.0180	0.6180±0.0350	0.5107±0.0077	0.6365±0.1184	0.5531±0.1048	0.5910±0.1169	0.4927/0.5034	0.4953/0.4967
Mixed (ours and traditional)	PCC	0.5231±0.0309	0.5017±0.0466	0.5869±0.0369	0.5189±0.0438	0.5314±0.1194	0.4338±0.0579	0.5703±0.0784	0.5453/0.5163	0.5120/0.5096
	SCC	0.5161±0.0239	0.5359±0.0261	0.5406±0.0542	0.4818±0.0611	0.5235±0.0934	0.5238±0.0524	0.5118±0.0500	0.5036/0.5267	0.4992/0.5108
	12 indicators	**0.7081±0.0366**	**0.7245±0.0414**	**0.8432±0.0313**	**0.6626±0.0514**	**0.7813±0.1483**	**0.6529±0.0892**	***0.7099±0.2264***	0.6115/0.5323	0.5739/0.4985
	15 indicators	**0.7197±0.0320**	**0.7276±0.0485**	**0.8426±0.0305**	**0.6634±0.0557**	**0.7815±0.1403**	***0.6523±0.0901***	0.7095±0.2275	0.6121/0.5233	0.5709/0.4954
Traditional (included to our indicators' set)	$\bar{e}$	0.5892±0.0284	0.5680±0.0509	0.8087±0.0361	0.5026±0.0037	0.6921±0.2067	0.6099±0.1639	0.6723±0.2059	0.5482/0.4997	0.5485/0.4991
	$std(e)$	0.5507±0.0296	0.5940±0.0456	0.8203±0.0298	0.5367±0.0510	0.6807±0.1161	0.6179±0.1918	0.6935±0.2198	0.5304/0.4980	0.5259/0.4967
	$\bar{G}_e$	0.6009±0.0343	0.5494±0.0504	0.7394±0.0441	0.5305±0.0443	0.7049±0.1976	0.5968±0.1190	0.6324±0.1675	0.5456/0.4997	0.5365/0.4997
	$\bar{s}$	0.5761±0.0311	0.4839±0.0659	0.7525±0.0426	0.5026±0.0037	0.6842±0.2050	0.6016±0.1594	0.6611±0.1954	0.5448/0.5127	0.5520/0.4991
	$std(s)$	0.5815±0.0312	0.4292±0.0567	0.8108±0.0327	0.5469±0.0543	0.6957±0.1446	0.6280±0.1823	0.6837±0.2147	0.5447/0.5044	0.5405/0.4981
	E_s	0.6816±0.0349	0.4835±0.0372	0.7873±0.0508	0.5026±0.0037	0.6861±0.1634	0.6359±0.0938	0.5409±0.0408	***0.6023/0.7501***	**0.5367/0.5580**
	PP_s	***0.6825±0.0333***	0.5669±0.0355	0.6337±0.0491	0.5026±0.0037	0.6861±0.1636	0.6506±0.1003	0.5289±0.0422	**0.6105/0.7650**	***0.5349/0.5352***
Baselines	$\bar{G}_s$	0.5757±0.0293	0.5516±0.0516	0.7884±0.0350	0.5026±0.0037	0.6867±0.2047	0.6129±0.1640	0.6651±0.1989	0.5491/0.5087	0.5534/0.4990
	MSP	0.6531±0.0303	0.5685±0.0380	0.6635±0.0531	***0.5976±0.0613***	0.6853±0.1494	0.6317±0.0861	0.6136±0.1575	0.6011/0.5289	0.5454/0.4978
	$LM\ PP$	0.5362±0.0274	0.6036±0.0412	***0.8224±0.0327***	0.5309±0.0469	0.5026±0.0037	0.6012±0.1823	0.6148±0.1866	0.5188/0.5000	0.5367/0.5000
	$\max(TE)$	0.5409±0.0217	0.4892±0.0548	0.7402±0.0321	0.5270±0.0292	0.6679±0.1637	0.6286±0.2203	**0.7147±0.2461**	0.5003/0.5000	0.5004/0.5000
	$\overline{TP}$	0.5552±0.0293	0.5740±0.0498	0.8124±0.0311	0.5026±0.0037	0.7126±0.2022	0.6148±0.1866	0.6884±0.2165	0.5357/0.4998	0.5437/0.5002
	$\min(TP)$	0.4971±0.0223	***0.6225±0.0311***	0.5074±0.0391	0.4954±0.0567	0.4996±0.0094	0.5302±0.0644	0.5478±0.0548	0.5060/0.4998	0.5059/0.5002
	$\bar{G}_{TP}$	0.5370±0.0265	0.5862±0.0348	0.8019±0.0362	0.5026±0.0037	0.4930±0.0333	0.6277±0.2206	**0.7148±0.2438**	0.5087/0.5000	0.5370/0.5000
	Const = 1	0.5000±0.0000	0.5000±0.0000	0.5000±0.0000	0.5000±0.0000	0.5000±0.0000	0.5000±0.0000	0.5000±0.5000	0.5000/0.5000	0.5000/0.5000
	Const = 0	0.5000±0.0000	0.5000±0.0000	0.5000±0.0000	0.5000±0.0000	0.5000±0.0000	0.5000±0.0000	0.5000±0.5000	0.5000/0.5000	0.5000/0.5000
	Rand. 0.5	0.5018±0.0304	0.5009±0.0430	0.5046±0.0314	0.5112±0.0367	0.5223±0.0015	0.5387±0.0304	0.4821±0.0046	0.5201/0.5375	0.5078/0.5185
	Rand. major	0.4935±0.0320	0.5099±0.0329	0.5108±0.0221	0.5040±0.0485	0.4877±0.0007	0.4965±0.0084	0.5150±0.0046	0.4898/0.5099	0.5016/0.4859
MLP										
Best results MLP		0.7141±0.0286	0.7284±0.0417	0.8604±0.0334	0.7035±0.0557	0.8299±0.1130	0.6389±0.2116	0.7153±0.2354	0.6148/0.7823	0.5309/0.5703
Best indicators combination		$\bar{s}, \bar{e}, std(s), \mathbf{E_p}, \mathbf{PP_p}, PP_s, \bar{\mathbf{G}}_\mathbf{p}, \bar{G}_s, \bar{G}_e$	$\bar{s}, \bar{e}, std(s), \mathbf{E_p}, E_s, PP_s, \bar{G}_s, \bar{G}_e$	$\bar{\mathbf{p}}, \mathbf{std(p)}, \bar{e}, std(e), \bar{s}, E_s, \bar{\mathbf{G}}_\mathbf{p}, \bar{G}_s$	$\bar{s}, std(e), \mathbf{E_p}, PP_s, \bar{\mathbf{G}}_\mathbf{p}, \bar{G}_s$	$\bar{\mathbf{p}}, \bar{s}, \mathbf{std(p)}, std(s), std(e), \mathbf{PCC}, \mathbf{E_p}, E_s, PP_s, \bar{\mathbf{G}}_\mathbf{p}, \bar{G}_s, \bar{G}_e$	$\bar{s}, \bar{e}, \mathbf{PCC}, \mathbf{SCC}, \mathbf{E_p}, \mathbf{PP_p}, \bar{G}_e$	$\bar{\mathbf{p}}, \mathbf{std(p)}, std(s), \mathbf{E_p}, E_s, \mathbf{PP_p}, PP_s, \bar{\mathbf{G}}_\mathbf{p}, \bar{G}_s, \bar{G}_e$	$\bar{s}, E_s, \bar{\mathbf{G}}_\mathbf{p}, \bar{G}_e$	$\mathbf{std(p)}, \mathbf{PP_p}, PP_s$

to expect that such information could be provided. To mitigate this issue, we "calibrate" outputs of generative models with that of experts models, representing intended model outputs to given inputs. We also introduced a metric of trust characterising probabilities of the model's correct generations conditional on the generations of expert models or agents. These estimated probabilities (calibrator probabilities) are statically grounded and together with a set of traditional features, after aggregations form a set of 15 indicators that may constitute a new basis for AI reliability evaluation. Our results demonstrate that classifiers trained on these 15 indicators can discriminate correct and incorrect answers. Our experiments illustrate this across three ontologies: the BF programming language, math problems, and medical question answering. In all settings, subsets including probability-based indicators yielded better discrimination than subsets comprising traditional indicators. Moreover, adding probability-based indicators to the traditional ones either improves transferability or leaves it nearly unchanged, both across models within the same ontological class and across ontologies entirely. Where the subsets of only traditional indicators transfer better, the margin is small.

7 LIMITATIONS

While the proposed framework is theoretically applicable to diverse generative AI modalities – for instance, by treating image patches as discrete tokens – the empirical validation in this study was limited to generative language models. The observed transferability of the trust measure within and across ontologies suggests that universal patterns underlying hallucinations may exist across language model outputs. However, since the indicators are predicated on probability distributions derived from an ontology-specific calibrator, adapting the classifier to a novel ontology (including alternative language model architectures) necessitates construction of a new calibrator for the target ontology. Calibrator-based probabilities improve performance only in combination with the traditional indicators (see Appendix F), and the most effective combination may differ across ontologies.

AI USE STATEMENT

In this work, we used generative AI tools for text checking on grammatical issues. We have not used generative AI tools for research ideas generation, experiment setup and run, making conclusions.

REPRODUCIBILITY STATEMENT

The code for all experiments is provided in our repository (Anonymous Authors, 2026). Section 4 provides a detailed methodological descriptions of our framework, Appendix E – technical details for experiments reproducing.

REFERENCES

Vibhor Agarwal, Yulong Pei, Salwa Alamir, and Xiaomo Liu. Codemirage: Hallucinations in code generated by large language models. *arXiv preprint arXiv:2408.08333*, 2024. doi: 10.48550/arxiv.2408.08333. Published: 2024-08-14.

Anonymous Authors. A statistical measure of trust in generative AI. `https://anonymous.4open.science/r/statistical_trust_measure/`, 2026. Anonymized repository for peer review.

Alexandra Bazarova, Aleksandr Yugay, Andrey Shulga, Alina Ermilova, Andrei Volodichev, Konstantin Polev, Julia Belikova, Rauf Parchiev, Dmitry Simakov, Maxim Savchenko, Andrey Savchenko, Serguei Barannikov, and Alexey Zaytsev. Hallucination detection in llms with topological divergence on attention graphs, 2025. URL `https://arxiv.org/abs/2504.10063`.

Patrick Billingsley. *Probabiliy and measure*. Wiley and Sons, 1995.

Dylan Bouchard and Mohit Singh Chauhan. Uncertainty quantification for language models: A suite of black-box, white-box, llm judge, and ensemble scorers. *arXiv preprint arXiv:2504.19254*, 2025.

Richard C Bradley. Basic properties of strong mixing conditions. a survey and some open questions. *Probability Surveys*, 2:107–144, 2005.

Chao Chen, Kai Liu, Ze Chen, Yi Gu, Yue Wu, Mingyuan Tao, Zhihang Fu, and Jieping Ye. Inside: Llms' internal states retain the power of hallucination detection. *arXiv preprint arXiv:2402.03744*, 2024.

Manuel Cossio. A comprehensive taxonomy of hallucinations in large language models. *arXiv preprint arXiv:2508.01781*, 2025. doi: 10.48550/arxiv.2508.01781. Published: 2025-08-03.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025a. URL `https://arxiv.org/abs/2501.12948`.

DeepSeek-AI. Deepseek-v3.2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025b.

A. Dvoretzky, J. Kiefer, and J. Wolfowitz. Asymptotic minimax character of the sample distribution function and of the classical multinomial estimator. *Ann. Math. Statist.*, 27(3):642–669, 1956.

Yijun Feng. Counterfactual probing for hallucination detection and mitigation in large language models. *arXiv preprint arXiv:2508.01862*, 2025. doi: 10.48550/arxiv.2508.01862. Published: 2025-08-03.

Jinlan Fu, See Kiong Ng, Zhengbao Jiang, and Pengfei Liu. Gptscore: Evaluate as you desire. In *Proceedings of the 2024 conference of the north american chapter of the association for computational linguistics: Human language technologies (volume 1: Long papers)*, pp. 6556–6576, 2024.

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

Anish Gunjal, Jihan Yin, and Erhan Bas. Detecting and preventing hallucinations in large vision language models. *arXiv preprint arXiv:2308.06394*, 2023. doi: 10.48550/arxiv.2308.06394. Published: 2023-08-11.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

Abdolhossein Hoorfar and Mehdi Hassani. Inequalities on the lambert w function and hyperpower function. *J. Inequal. Pure and Appl. Math*, 9(2):5–9, 2008.

Bairu Hou, Yujian Liu, Kaizhi Qian, Jacob Andreas, Shiyu Chang, and Yang Zhang. Decomposing uncertainty for large language models through input clarification ensembling. *arXiv preprint arXiv:2311.08718*, 2023.

L Huang, W Yu, W Ma, W Zhong, Z Feng, H Wang, Q Chen, W Peng, X Feng, B Qin, et al. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions (arxiv: 2311.05232). arxiv, 2023.

Florian A Huber and Roman Guggenberger. Ai msk clinical applications: spine imaging. *Skeletal Radiology*, 51(2):279–291, 2022.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

Minsuk Kahng, Ian Tenney, Mahima Pushkarna, Michael Xieyang Liu, James Wexler, Emily Reif, Krystal Kallarackal, Minsuk Chang, Michael Terry, and Lucas Dixon. Llm comparator: Interactive analysis of side-by-side evaluation of large language models. *IEEE Transactions on Visualization and Computer Graphics*, 2024.

Junyi Li, Jie Chen, Ruiyang Ren, Xiaoxue Cheng, and Wayne Xin Zhao. The dawn after the dark: An empirical study on factuality hallucination in large language models. *arXiv preprint arXiv:2401.03205*, 2024. doi: 10.48550/arxiv.2401.03205. Published: 2024-01-06.

Mengfei Liang, Archish Arun, Zekun Wu, Cristian Munoz, and Jonathan Lutch. Thames: An end-to-end tool for hallucination mitigation and evaluation in large language models. *arXiv preprint arXiv:2409.11353*, 2024. doi: 10.48550/arxiv.2409.11353. Published: 2024-09-17.

Stephanie Lin, Jacob Hilton, and Owain Evans. Teaching models to express their uncertainty in words. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL https://openreview.net/forum?id=8s8K2UZGTZ.

Huan Ma, Jiadong Pan, Jing Liu, Yan Chen, Joey Tianyi Zhou, Guangyu Wang, Qinghua Hu, Hua Wu, Changqing Zhang, and Haifeng Wang. Semantic energy: Detecting llm hallucination beyond entropy. *arXiv preprint arXiv:2508.14496*, 2025.

P. Massart. The tight constant in the Dvoretzky-Kiefer-Wolfowitz inequality. *Ann. Probab.*, 18(3): 1269–1283, 1990.

Justus Mattern, Sami Jaghouar, Manveer Basra, Jannik Straube, Matthew Di Ferrante, Felix Gabriel, Jack Min Ong, Vincent Weisser, and Johannes Hagemann. Synthetic-1: Two million collaboratively generated reasoning traces from deepseek-r1, 2025. URL https://www.primeintellect.ai/blog/synthetic-1-release.

Niels Mündler, Jingxuan He, and Martin Vechev. Self-contradictory hallucinations of large language models: Evaluation, detection and mitigation. *arXiv preprint arXiv:2305.15852*, 2023. doi: 10.48550/arXiv.2305.15852. Published: 2023-05-25.

Michael Naaman. On the tight constant in the multivariate dvoretzky–kiefer–wolfowitz inequality. *Statistics & Probability Letters*, 173:109088, 2021.

Maziyar Panahi. Medical-reasoning-sft-gpt-oss-120b. https://huggingface.co/datasets/OpenMed/Medical-Reasoning-SFT-GPT-OSS-120B, 2025. Hugging Face Dataset.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Murray Rosenblatt. A central limit theorem and a strong mixing condition. *Proceedings of the national Academy of Sciences*, 42(1):43–47, 1956.

Andrew Sellergren, Chufan Gao, Fereshteh Mahvar, Timo Kohlberger, Fayaz Jamil, Madeleine Traverse, Alberto Tono, Bashir Sadjad, Lin Yang, et al. Medgemma 1.5 technical report. *arXiv preprint arXiv:2604.05081*, 2026.

Adi Simhi, Jonathan Herzig, Idan Szpektor, and Yonatan Belinkov. Constructing benchmarks and interventions for combating hallucinations in llms. *arXiv preprint arXiv:2404.09971*, 2024. doi: 10.48550/arxiv.2404.09971. Published: 2024-04-15.

Linwei Tao, Yi-Fan Yeh, Minjing Dong, Tao Huang, Philip Torr, and Chang Xu. Revisiting uncertainty estimation and calibration of large language models. *arXiv preprint arXiv:2505.23854*, 2025.

Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL https://qwenlm.github.io/blog/qwen2.5/.

Hasimah Ibrahim Teo and Amilah Shari. Enhance fake review detection: A hybrid approach of implicit absa and imbalanced dataset handling. *Journal of Theoretical and Applied Information Technology*, 103(13), 2025.

Neeraj Varshney. A stitch in time saves nine: Detecting and mitigating hallucinations of llms by validating low-confidence generation. *arXiv preprint arXiv:2307.03987*, 2023. doi: 10.48550/arxiv.2307.03987. Published: 2023-07-08.

Hongbin Ye, Tong Liu, Aijia Zhang, Wei Hua, and Weiqiang Jia. Cognitive mirage: A review of hallucinations in large language models. *arXiv preprint arXiv:2309.06794*, 2023. doi: 10.48550/arxiv.2309.06794.

Ryozo Yokoyama. Moment bounds for stationary mixing sequences. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 52(1):45–57, 1980.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. Bartscore: Evaluating generated text as text generation. *Advances in neural information processing systems*, 34:27263–27277, 2021.

Tal Zeevi, Rajesh Venkataraman, Lawrence H Staib, and John A Onofrey. Monte-carlo frequency dropout for predictive uncertainty estimation in deep learning. In *2024 IEEE International Symposium on Biomedical Imaging (ISBI)*, pp. 1–5. IEEE, 2024.

Yue Zhang, Yafu Li, Leyang Cui, Deng Li Cai, and Lemao Liu. Siren's song in the ai ocean: A survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*, 2023. doi: 10.48550/arxiv.2309.01219. Published: 2023-09-03.

Yue Zhang, Yafu Li, Leyang Cui, Cai Deng, and Lemao Liu. Siren's song in the ai ocean: A survey on hallucination in large language models. *Computational Linguistics*, 2025. doi: 10.1162/coli.a.16. Published: 2025-07-07.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Processbench: Identifying process errors in mathematical reasoning. In *The 63rd Annual Meeting of the Association for Computational Linguistics*, 2025.

A Related work

In this Section, we review existing work on hallucination detection and mitigation in LLMs, with a particular focus on approaches that motivate or relate to our proposed framework.

A.1 CONSISTENCY-BASED DETECTION

Consistency-based methods detect hallucinations by identifying contradictions or instability within a model's outputs or across multiple generations. These approaches rely on internal agreement rather than external knowledge, making them applicable in black-box settings.

Self-contradiction detection generates paired sentences or internal claims and employs automated checking (via prompting or classification) to identify logical inconsistencies. Constrained prompting techniques elicit comparable sentence pairs that can be systematically evaluated for contradictions (Mündler et al., 2023). Self-contradiction methods, however, only detect internal contradictions; many factual hallucinations produce no internal contradiction. Moreover, these methods require generating multiple outputs or auxiliary prompts, which increases computational cost and introduces sensitivity to the prompting strategy used to elicit contradictions.

Self-consistency and *Best-of-N sampling* approaches sample multiple outputs (n-best) and measure agreement through majority voting or semantic similarity. Outputs with low agreement across samples are flagged as potentially hallucinated (Zhang et al., 2023; 2025). These methods are computationally expensive, requiring multiple forward passes; they may miss consistently repeated hallucinations that appear across all samples.

A.2 EXTERNAL VERIFICATION AND RETRIEVAL-BASED METHODS

External verification methods ground model outputs in retrieved sources or knowledge bases, enabling detection of unsupported claims through comparison with authoritative information.

Retrieval-Augmented Generation (RAG) and grounding conditions generation on retrieved documents or knowledge snippets, allowing outputs to be traced to sources. Detection involves comparing generated claims to retrieved evidence or enforcing grounding constraints during decoding (Liang et al., 2024). However, retrieval noise, stale sources, or biased corpora can propagate errors. Incorrect alignment between retrieved passages and claims can produce grounded-but-incorrect outputs.

Fact-checking with reward models and rejection sampling. Fine-grained reward or classifier models are trained on human annotations marking hallucinations. These models score generations, enabling rejection sampling or fine-tuning to prefer non-hallucinatory outputs (Gunjal et al., 2023). The approach requires substantial annotated data, which is costly to obtain. It inherits annotator biases and coverage limitations. Furthermore, generalization to out-of-distribution ontologies can be limited.

Knowledge base verification and benchmarks verify model statements by querying structured knowledge bases or curated benchmarks to identify unsupported facts (Li et al., 2024). Their drawbacks include limited recall when knowledge bases lack coverage for the ontology. Also, mapping free-text claims to structured knowledge base queries is non-trivial.

A.3 MODEL-BASED AND PROMPT-BASED DETECTION

This category encompasses training separate classifiers to detect hallucinations and using prompt engineering to elicit self-verification from the model itself.

Separate detector models and classifiers. A classifier or detector model is fine-tuned on annotated examples to predict whether generated content is hallucinated. Detectors can operate at token, span, or full-answer levels (Agarwal et al., 2024; Gunjal et al., 2023). The methods are heavily dataset-dependent and require costly annotation efforts.

Prompted self-verification and *Chain-of-Thought (CoT) checks*. The model is prompted to list supporting evidence, explain its reasoning chain, or directly judge whether a generated claim is supported (self-critic approach) (Mündler et al., 2023; Liang et al., 2024). Models, however, can produce plausible but fabricated supporting evidence. Self-verification may fail when models lack knowledge to judge their own outputs.

A.4 COUNTERFACTUAL PERTURBATIONS AND INTERVENTION METHODS

These methods intentionally alter claims or model components to measure sensitivity, using behavioral responses to distinguish hallucinations from robust knowledge.

Counterfactual probing and perturbation sensitivity generate counterfactual variants of a claim (introducing subtle factual changes), and the model's confidence or output consistency is measured. Hallucinated content is expected to show inconsistent sensitivity compared to grounded knowledge (Feng, 2025). The challenge, however, is that generating high-quality counterfactuals is non-trivial and may require ontology knowledge. False conclusions can arise if counterfactuals are unnatural or easily distinguished. Computational cost increases with the volume of counterfactuals tested.

Component interventions during generation target specific model components (attention heads, MLP layers, residual streams) to identify causal factors in hallucination generation. Steering pre-hallucination signals can prevent hallucinations, informing detection strategies (Simhi et al., 2024). Intervention success varies by component and may harm language modeling quality. It also requires extra computational resources. Importantly, findings may not generalize across model architectures.

A.5 SUMMARY

Despite the breadth and depth of the repertoire of methods, approaches and tools assisting the identification of trusted AI generations, several issues remain open. Common limitations across detection approaches include:

Dataset and ontology sensitivity: Detectors and reward models rely on annotated distributions and can fail under ontology shift (Agarwal et al., 2024; Simhi et al., 2024).

Overconfidence problem: Model confidence is an imperfect proxy for factuality, producing both false negatives (overconfident hallucinations) and false positives (uncertain correct outputs) (Varshney, 2023).

Computational cost: Many effective methods (ensemble sampling, retrieval, counterfactual generation) impose significant computational overhead (Zhang et al., 2023; Feng, 2025).

Sentence-level detection: Most methods only support sentence- or response-level hallucination detection, which can jeopardize the perceived reliability of the entire answer even when it is largely correct.

B THEORETICAL JUSTIFICATION

In what follows we provide a theoretical justification of the possibility to learn ontology-calibrated confidence from expert's data. We do so by showing that the probability bounds on threshold-defined acceptance or rejection events, specified by Δ_j^i in Algorithm 1, can indeed be inferred from empirical calibration. In fact these bounds converge to true probabilities as the size of the calibration sets grow. For simplicity, the results are presented in two steps. In Section B.1 we show how these bounds can be derived within standard i.i.d. setting. This enables a clear and focused explanation of main theoretical machinery supporting our approach. In Section B.2 we demonstrate how the results can be extended to more practical setups in which i.i.d. assumptions could be violated.

B.1 SIMPLIFIED SETTINGS

Let elements of the set $\mathcal{S}$ be independent samples drawn from the same, albeit unknown, distribution. In this case conditional probabilities of confidence judges' $\mathcal{A}^i$ acceptance or rejection of AI model's output at thresholds Δ_j^i could be estimated from empirical data. Moreover, when subsets $\mathcal{S}_{+,j}$ and $\mathcal{S}_{-,j}$ are sufficiently large, these estimates converge to true conditional probabilities. The results are formally stated in Theorem 1.

Theorem 1. *Let measurable maps G^S, G^E, feature map Φ, and confidence judges' training set $\mathcal{S}$ be defined as in (1) – (2). Suppose that the elements of $\mathcal{S}$ are independently sampled from the (unknown) data distribution P_D, that $\mathcal{S}_{+,j}$ and $\mathcal{S}_{-,j}$ are not empty, and that G^S, G^E and Φ are chosen independently of $\mathcal{S}$. Pick $\lambda \in (0, \sqrt{16/e})$.*

For each class index j, select h_j from $\mathcal{H}$ independently of $\mathcal{S}$, and let $\mathcal{A}^i$ be the outputs of Algorithm 1 with inputs $G^S, \Phi, \mathcal{S}, \{h_j \in \mathcal{H} | j = 1, \ldots, q\}$ and $\Delta_j^i \in (0, 1)$.

Algorithm 1 Confidence Judge

Require: Generative model G, feature map Φ, forming dataset $\mathcal{S}$, projectors $h_j \in \mathcal{H} : \mathbb{R}^d \to \mathbb{R}$, and parameters $\Delta_j^i \in (0,1)$, $i = 1, \ldots, k$, $k \in \mathbb{N}$ defining the judge's confidence grid.

1: **for** $j = 1$ to q **do**

2: Compute $\mathcal{S}_{-,j}$, $\mathcal{S}_{+,j}$ and functions $F_{-,j}, F_{+,j}$ (as defined in (3)).

3: Define the class-wise calibrators

$$\mathcal{A}_j^i(z) = \begin{cases} \tau^{\times}, & \text{if } h_j(z) \le F_{-,j}^{\dagger}(\Delta_j^i), \\ \tau_j, & \text{otherwise.} \end{cases}$$

The map $F_{-,j}^{\dagger}$ is defined as

$$F_{-,j}^{\dagger}(z) = \inf\{x \in \mathbb{R} \,:\, F_{-,j}(x) \ge z\}.$$

4: **end for**

5: Define global calibrators

$$\mathcal{A}^i(\tau_j, z) = \mathcal{A}_j^i(z).$$

Ensure: calibrator maps $\mathcal{A}^i$.

Then, for a new independent sample $(u, \ell) \sim P_D$ and each $j = 1, \ldots, q$, conditional on the realized $M_{+,j}, M_{-,j}$ the following hold:

With probability at least

$$\max\{1 - \lambda r(M_{+,j}, \lambda), 0\}$$

over the draw of $\mathcal{S}_{+,j}$

$$\begin{aligned} 1 - F_{+,j}(F_{-,j}^{\dagger}(\Delta_j^i)) + r(M_{+,j}, \lambda) &\ge P(\mathcal{A}^i(G^S(u), \Phi(u)) = \tau_j \,|\, G^S(u) = \tau_j,\ G^E(u) = \tau_j) \\ &\ge 1 - F_{+,j}(F_{-,j}^{\dagger}(\Delta_j^i)) - r(M_{+,j}, \lambda). \end{aligned}$$

Similarly, with probability at least

$$\max\{1 - \lambda r(M_{-,j}, \lambda), 0\}$$

over the draw of $\mathcal{S}_{-,j}$

$$\begin{aligned} F_{-,j}(F_{-,j}^{\dagger}\Delta_j^i) + r(M_{-,j}, \lambda) &\ge P(\mathcal{A}^i(G^S(u), \Phi(u)) = \tau^{\times} \,|\, G^S(u) = \tau_j,\ G^E(u) \ne \tau_j) \\ &\ge \Delta_j^i - r(M_{-,j}, \lambda), \end{aligned}$$

where

$$r(x, \lambda) = \left(\frac{1}{4x} \left[\log(16x\lambda^{-2}) - \frac{1}{2} \log\log(16x\lambda^{-2}) \right] \right)^{1/2}.$$

Proof of Theorem 1. Given that G^S, G^E are measurable and elements in $\mathcal{S}$ are independently sampled from the distribution P_D, we can conclude that $\mathcal{S}_j$ and $\mathcal{S}_{+,j}$, $\mathcal{S}_{-,j}$ are independent samples from their corresponding conditional distributions $P_D((u,\tau) \,|\, G^S(u) = \tau_j)$, $P_D((u,\tau) \,|\, G^E(u) = \tau_j,\ G^S(u) = \tau_j)$, and $P_D((u,\tau) \,|\, G^E(u) \ne \tau_j,\ G^S(u) = \tau_j)$. Moreover, since $h_j \in \mathcal{H}$ is chosen independently of $\mathcal{S}$ and that h_j and Φ are measurable, we may conclude that elements of the multi-sets

$$\begin{aligned} \mathcal{X}_{+,j} &= \{h_j(\Phi(u)),\ \text{for each } (u,\tau) \in \mathcal{S}_{+,j}\}, \\ \mathcal{X}_{-,j} &= \{h_j(\Phi(u)),\ \text{for each } (u,\tau) \in \mathcal{S}_{-,j}\}, \end{aligned}$$

are independent samples from the respective conditional probability distributions induced on $\mathbb{R}$ by h_j.

To proceed with the proof we will need the following lemma.

Lemma 1. *Let $\theta \in \mathbb{R}$, $\lambda \in (0, \infty)$, and $\mathcal{X} \subset \mathbb{R}$ be a finite multi-set whose elements are sampled independently from some unknown probability distribution. Let $|\mathcal{X}| = n$, and F_n be defined as follows:*

$$F_n(s) = \frac{1}{n} \sum_{x \in \mathcal{X}} \mathbf{1}_{x \le s}.$$

Let z be a new independent sample from the same distribution and consider the events:

$$E_1 : \ z \leq \theta, \ E_2 : \ z > \theta.$$

Then, conditional on the realized n, with probability at least

$$\max\{1 - \lambda \left(\tfrac{1}{4n} W_0 \left(\tfrac{16n}{\lambda^2}\right)\right)^{1/2}, 0\}$$

over the draw of $\mathcal{X}$

$$F_n(\theta) + \left(\tfrac{1}{4n} W_0 \left(\tfrac{16n}{\lambda^2}\right)\right)^{1/2} \geq P(E_1) \geq F_n(\theta) - \left(\tfrac{1}{4n} W_0 \left(\tfrac{16n}{\lambda^2}\right)\right)^{1/2} \tag{5}$$

and

$$1 - F_n(\theta) + \left(\tfrac{1}{4n} W_0 \left(\tfrac{16n}{\lambda^2}\right)\right)^{1/2} \geq P(E_2) \geq 1 - F_n(\theta) - \left(\tfrac{1}{4n} W_0 \left(\tfrac{16n}{\lambda^2}\right)\right)^{1/2}, \tag{6}$$

where W_0 is the principal branch of the Lambert W function.

Proof of Lemma 1. Let P_c be the cumulative distribution function associated with the unknown probability distribution from which both the multi-set $\mathcal{X}$ and z were independently drawn. It is clear that

$$P(E_1) = P_c(\theta), \qquad P(E_2) = 1 - P_c(\theta).$$

Introduce the auxiliary event

$$E(\varepsilon) : \ \sup_{s \in \mathbb{R}} |F_n(s) - P_c(s)| > \varepsilon,$$

and its complement $\bar{E}(\varepsilon) : \text{not } E(\varepsilon)$. The Dvoretzky-Kiefer-Wolfowitz inequality (Dvoretzky et al., 1956; Massart, 1990) states that, for any $\varepsilon > 0$,

$$P\left(\sup_{s \in \mathbb{R}} |F_n(s) - P_c(s)| > \varepsilon\right) \leq 2\exp(-2n\varepsilon^2), \tag{7}$$

and therefore, event $E(\varepsilon)$ occurs with probability at most $2\exp(-2n\varepsilon^2)$ over the draw of $\mathcal{X}$. Similarly, event $\bar{E}(\varepsilon)$ occurs with probability at least $1 - 2\exp(-2n\varepsilon^2)$ over the draw of $\mathcal{X}$.

Now, consider the following equation

$$\lambda\varepsilon = 2\exp(-2\varepsilon^2 n).$$

Its solution ε^* is

$$\varepsilon^* = \left(\frac{1}{4n} W_0 \left(\frac{16n}{\lambda^2}\right)\right) > 0.$$

Given that the Dvoretzky-Kiefer-Wolfowitz inequality holds true for any $\varepsilon > 0$, it is certainly true for ε^*. Substituting ε^* into the bounds and recalling that $\lambda\varepsilon^* = 2\exp(-2{\varepsilon^*}^2 n)$ completes the proof. □

Proof of Theorem 1 continued. By the construction of $\mathcal{A}^i$ in Algorithm 1, the probability that the confidence judge correctly accepts the label assigned by the underlying AI system is:

$$\begin{aligned} &P(\mathcal{A}^i(G^S(u), \Phi(u)) = \tau_j \,|\, G^S(u) = \tau_j, \ G^E(u) = \tau_j) \\ &= P(h_j(\Phi(u)) > F^{\dagger}_{-,j}(\Delta^i_j) \,|\, G^S(u) = \tau_j, \ G^E(u) = \tau_j). \end{aligned} \tag{8}$$

Since $F_{+,j}$, as defined in (3), is empirical cumulative distribution function of the conditional distribution $P(h_j(\Phi(u)) \,|\, G^S(u) = \tau_j, \ G^E(u) = \tau_j)$, bounds (6) for event E_2 from Lemma 1 apply (with $\theta = F^{\dagger}_{-,j}(\Delta^i_j)$, $F_n = F_{+,j}$, $n = M_{+,j}$).

Turning to the probability of the confidence judge $\mathcal{A}^i$ correctly rejecting an incorrect label assigned by the underlying AI system, we observe that

$$\begin{aligned} &P(\mathcal{A}^i(G^S(u), \Phi(u)) = \tau^{\times} \,|\, G^S(u) = \tau_j, \ G^E(u) \neq \tau_j) \\ &= P(h_j(\Phi(u)) \leq F^{\dagger}_{-,j}(\Delta^i_j) \,|\, G^S(u) = \tau_j, \ G^E(u) \neq \tau_j). \end{aligned}$$

Similarly to the previous case, we note that the conditional distribution $P(h(\Phi(u)) \,|\, G^S(u) = \tau_j, \ G^E(u) \neq \tau_j)$ admits a cumulative distribution function with empirical approximation $F_{-,j}$ which is used by Algorithm 1. Therefore bounds (5), for the event E_1 from Lemma 1 apply (with $\theta = F^{\dagger}_{-,j}(\Delta^i_j)$, $F_n = F_{-,j}$, $n = M_{-,j}$, and observing that $F_{-,j}(F^{\dagger}_{-,j}(\Delta^i_j)) \geq \Delta^i_j$).

Finally, we note that for any $x \geq e$, the principal branch W_0 of the Lambert W function satisfies (Hoorfar & Hassani, 2008)

$$\log(x) - \log\log(x) \leq W_0(x) \leq \log(x) - \frac{1}{2}\log\log(x).$$

Combining this with bound stemming from Lemma 1 completes the proof. □

B.2 GENERALIZATION BEYOND I.I.D. ASSUMPTIONS

Theorem 1 explicitly assumes that 1) elements of the set $\mathcal{S}$ are all drawn independently from the same distribution, and 2) a new measurement is sampled independently from the same distribution. Here we show how these results could be generalised to dependent samples.

In order to do so, we need to introduce notion enabling quantification of various degrees of dependencies. For LLMs the most relevant framework concerns dependencies in sequences and sequential data. Therefore, consider $(x_i)_{i\in\mathbb{Z}}$, a strictly stationary sequence of real-valued random variables defined on a common probability space $(\Omega, \mathcal{B}, P)$. For integers $k \in \mathbb{Z}$, let

$$\mathcal{F}_{-\infty}^{k} \;=\; \sigma(x_j : j \le k), \qquad \mathcal{F}_{k}^{\infty} \;=\; \sigma(x_j : j \ge k)$$

denote the σ-algebras of the past up to time k and the future from time k onward, respectively.

Definition 1 (Strong-mixing coefficient)**.** *The strong-mixing (or α-mixing) coefficient at lag $n \ge 1$ is*

$$\alpha(n) \;=\; \sup_{k\in\mathbb{Z}} \sup_{\substack{A\in\mathcal{F}_{-\infty}^{k} \\ B\in\mathcal{F}_{k+n}^{\infty}}} \big|P(A\cap B) - P(A)\,P(B)\big|.$$

The sequence (x_i) is said to be α-mixing *(or* strongly mixing*, in the sense of Rosenblatt (Rosenblatt, 1956)) if $\alpha(n) \to 0$ as $n \to \infty$.*

We say that the mixing is geometric *(or that the sequence is geometrically α-mixing) if there exists $\lambda > 0$ such that $\alpha(n) = O(e^{-\lambda n})$ as $n \to \infty$.*

The mixing is polynomial of order p if there exist constants $K_0 > 0$ and $p > 1$ such that

$$\alpha(n) \;\le\; K_0\, n^{-p} \qquad \textit{for all } n \ge 1. \tag{9}$$

The α-mixing coefficient measures essentially how far the probability $P(A \cap B)$ of the joint event $A \cap B$ deviates from the product $P(A)\,P(B)$.

We note that for stationary sequences the supremum over k in Definition 1 is redundant and may be replaced by any single value of k. We also note that the motion of mixing presented in Definition 1 is the least constrained amongst other classical mixing notions. Indeed it is implied by β-mixing (absolute regularity) and ϕ-mixing (uniform mixing), with $2\alpha(n) \le \beta(n) \le \phi(n)$. See (Bradley, 2005) for details, Eqns. (1.11)–(1.18)).

At the heart of the proof of Theorem 1, and the sole place where independence of samples is required, is Lemma 1. There the independence is needed to appropriately invoke the Dvoretzky–Kiefer–Wolfowitz inequality (Dvoretzky et al., 1956; Massart, 1990). Hence if a suitable replacement to the classical Dvoretzky–Kiefer–Wolfowitz inequality is used that does not require independence, then Lemma 1 and Theorem 1 could be generalized to dependent settings too.

B.2.1 GEOMETRIC MIXING

The replacement follows immediately from (Naaman, 2021).

Lemma 2 (Naaman 2021, Lemma 4.3, $k = 1$)**.** *Let (x_i) be a stationary α-mixing sequence of real-valued random variables with marginal CDF F, and suppose $\alpha(n) = O(e^{-\lambda n})$ for some $\lambda > 0$. Let (δ_n) be a non-decreasing positive integer sequence satisfying*

$$\delta_n \to \infty \qquad \textit{and} \qquad \frac{\sqrt{n}}{\delta_n} \to \infty \qquad \textit{as } n \to \infty. \tag{10}$$

Then for every $t \ge 0$ and all n sufficiently large,

$$P\left(\sup_{\theta\in\mathbb{R}} \left| \sum_{i=\delta_n}^{n+\delta_n-1} \Big(\mathbf{1}\{x_i \le \theta\} - F(\theta)\Big) \right| \;\ge\; nt \right) \;\le\; (n+\delta_n)^2 \exp\!\left(-\frac{n\,t^2}{32\,\delta_n^2\,(1+\ln n)}\right). \tag{11}$$

The sequence δ_n trades off the polynomial pre-exponential factor against the rate inside the exponent in (11). With the choice $\delta_n = \lfloor n^{1/p} \rfloor$ for any fixed $p > 2$, the conditions in (10) are satisfied, and the exponent in (11) becomes

$$\frac{n\,t^2}{32\,\delta_n^2(1+\ln n)} \;=\; \frac{n^{1-2/p}\,t^2}{32(1+\ln n)},$$

while the prefactor $(n+\delta_n)^2$ remains $O(n^2)$. As $p \to \infty$, the exponential rate approaches $n/\log n$, almost matching (up to constants and the polynomial prefactor) the exponential rate of the classical Dvoretzky–Kiefer–Wolfowitz inequality.

B.2.2 POLYNOMIAL MIXING

When the mixing is polynomial, the dependencies have longer tails and convergence of empirical estimates of the cumulative distribution function to the true one may no longer follow exponential law. Yet, for large enough data samples the convergence still occurs, as is shown in the theorem below.

Theorem 2. *Let (x_i) be strictly stationary and α-mixing with polynomial rate* (9) *for some $p > 1$ and $K_0 > 0$. For every $q \in (2,\, p+1)$ there exists a constant $C = C(q, p, K_0) < \infty$ such that for every $n \geq 2$ and every $t \in (0, 1]$,*

$$P\left(\sup_{\theta\in\mathbb{R}} \frac{1}{n}\Big|\sum_{i=1}^{n}\Big(\mathbf{1}\{x_i \leq \theta\}) - F(\theta)\Big| > t\right) \;\leq\; \frac{C}{n^{\,q/2}\, t^{q+1}}. \tag{12}$$

Proof of Theorem 2. The proof proceeds in three steps: a moment bound for partial sums via Yokoyama's inequality (step 1), a pointwise tail via Markov inequality (step 2), followed by a standard deterministic grid reduction (step 3).

Recall that

$$F_n(\theta) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}\{x_i \leq \theta\}\,.$$

In what follows we will use the standard notation for the *left limit* of a non-decreasing function $g : \mathbb{R} \to \mathbb{R}$ at $x \in \mathbb{R}$:

$$g(x^-) \;:=\; \lim_{\varepsilon\downarrow 0} g(x-\varepsilon) \;=\; \lim_{y\uparrow x} g(y).$$

It is clear that

$$F(x^-) \;=\; P(x_1 < x), \qquad F_n(x^-) \;=\; \frac{1}{n}\,\big|\{i : x_i < x\}\big|,$$

with the gap $F(x) - F(x^-) = P(x_1 = x)$ equal to the atom mass of F at x (which vanishes when F is continuous at x). Both F and F_n are right-continuous and non-decreasing, so all left limits exist and lie below the corresponding values: $F(x^-) \leq F(x)$ and $F_n(x^-) \leq F_n(x)$.

Step 1 (Accounting for dependencies via the Yokoyama's moments inequality). Fix $\theta \in \mathbb{R}$ and define two centred indicator processes

$$w_i(\theta) \;=\; \mathbf{1}\{x_i \leq \theta\} - F(\theta), \quad w_i^-(\theta) \;=\; \mathbf{1}\{x_i < \theta\} - F(\theta^-), \qquad i \in \mathbb{Z}.$$

Observe that $\mathbb{E}(w_i(\theta)) = 0$, $\mathbb{E}(w_i^-(\theta)) = 0$ and $|w_i(\theta)| \leq 1$, $|w_i^-(\theta)| \leq 1$. That is, both sequences are bounded by 1 in absolute value and centred. Moreover, they are measurable functions of x_i and as such inherit the α-mixing of (x_i) with coefficients no larger than $\alpha(\cdot)$. To see this, note first that the maps $\phi(y) = \mathbf{1}\{y \leq \theta\} - F(\theta)$ and $\phi^-(y) = \mathbf{1}\{y < \theta\} - F(\theta^-)$ are Borel-measurable functions $\mathbb{R} \to \mathbb{R}$: they are indicators of the Borel sets $(-\infty, \theta]$ and $(-\infty, \theta)$ respectively. By the measurability of compositions (Billingsley, 1995, Theorem 13.1, p. 182), $\phi(x_i) = w_i(\theta)$ and $\phi^-(x_i) = w_i^-(\theta)$ are themselves measurable random variables on the underlying probability spaces $(\Omega, \mathcal{F}_{-\infty}^{k}, P)$, $(\Omega, \mathcal{F}_{k+n}^{\infty}, P)$ (see Definition 1).

Let $\mathcal{B}_{\mathbb{R}}$ denote the Borel σ-field on $\mathbb{R}$. Recall that the σ-field generated by a random variable $X : \Omega \to \mathbb{R}$ on $(\Omega, \mathcal{B}, P)$ is the σ-field of preimages $\sigma(X) = X^{-1}(\mathcal{B}_{\mathbb{R}}) = \{X^{-1}(B) : B \in \mathcal{B}_{\mathbb{R}}\} \subseteq \mathcal{B}$. For $\psi \in \{\phi, \phi^-\}$, and $\mathcal{B} \in \{\mathcal{F}_{-\infty}^{k}, \mathcal{F}_{-\infty}^{k}\}$

$$\sigma(\psi(x_i)) \;=\; (\psi \circ x_i)^{-1}(\mathcal{B}_{\mathbb{R}}) \;=\; x_i^{-1}\big(\psi^{-1}(\mathcal{B}_{\mathbb{R}})\big) \;\subseteq\; x_i^{-1}(\mathcal{B}_{\mathbb{R}}) \;=\; \sigma(x_i),$$

where σ stands for the corresponding sigma-fields, and the inclusion uses $\psi^{-1}(\mathcal{B}_{\mathbb{R}}) \subseteq \mathcal{B}_{\mathbb{R}}$, i.e. Borel-measurability of ψ. Therefore, all three σ-fields $\sigma(x_i)$, $\sigma(w_i(\theta))$ and $\sigma(w_i^-(\theta))$ are sub-σ-fields of the same ambient $\mathcal{B}$. Hence the σ-field monotonicity of the mixing coefficient in Definition 1 implies that the same mixing coefficient is inherited by $\omega_i(\theta), \omega_i^-(\theta)$.

Observe that both $\omega_i(\theta)$ and $\omega_i(\theta^-)$ are bounded. This implies that their q-th moments, $q \in \mathbb{N}$ are all defined and boun dned. By Yokoyama's moment inequality (Yokoyama, 1980, Theorem 1), for every $q > 2$ such that

$$\sum_{k=1}^{\infty} (k+1)^{q-2}\, \alpha(k) \;<\; \infty, \tag{13}$$

there exists a constant $A_q < \infty$, depending only on q and on the sum in (13), such that

$$\mathbb{E}\left|\sum_{i=1}^{n} w_i(\theta)\right|^q \;\le\; A_q\, n^{q/2}, \qquad \mathbb{E}\left|\sum_{i=1}^{n} w_i^-(\theta)\right|^q \;\le\; A_q\, n^{q/2}, \quad n \ge 1. \tag{14}$$

Under polynomial mixing (9), condition (13) reduces to $\sum_k (k+1)^{q-2-p} < \infty$, i.e. $q-2-p < -1$, equivalently $q < p+1$.

Step 2 (point-wise tail bounds). Set $S_n(\theta) = \sum_{i=1}^n w_i(\theta)$. According to Markov's inequality and (14), we obtain that for any $t > 0$,

$$\begin{aligned} P\big(|F_n(\theta) - F(\theta)| \ge t\big) = P\big(|S_n(\theta)| \ge nt\big) \;\le\; \frac{\mathbb{E}|S_n(\theta)|^q}{(nt)^q} \;\le\; \frac{A_q}{n^{q/2}\, t^q}, \\ P\big(|F_n(\theta^-) - F(\theta^-)| \ge t\big) \le \frac{A_q}{n^{q/2}\, t^q}. \end{aligned} \tag{15}$$

Step 3 (Bounding the probability of supremum over $\mathbb{R}$). For an integer $N \ge 1$, define the grid via the standard quantile function:

$$\theta_0 = -\infty, \qquad \theta_N = +\infty, \qquad \theta_i = \inf\{x \in \mathbb{R} : F(x) \ge i/N\}, \quad i = 1, \dots, N-1.$$

Note, that the grid points θ_i, $i = 1, \dots, N-1$ satisfy $\theta_{i-1} \le \theta_i$, but they are not necessarily all distinct (consider e.g. distributions comprising a single atom containing all probability mass).

Right-continuity of F, combined with the definition of the infimum, yields

$$F(\theta_i^-) \;\le\; \frac{i}{N} \;\le\; F(\theta_i), \qquad i = 1, \dots, N-1. \tag{16}$$

For any $\theta \in [\theta_{i-1}, \theta_i)$, monotonicity of F_n and F ensures that

$$F_n(\theta_{i-1}) \;\le\; F_n(\theta) \;\le\; F_n(\theta_i^-), \qquad F(\theta_{i-1}) \;\le\; F(\theta) \;\le\; F(\theta_i^-),$$

and from (16) the oscillation of F on this interval is bounded by

$$F(\theta_i^-) - F(\theta_{i-1}) \;\le\; \frac{i}{N} - \frac{i-1}{N} \;=\; \frac{1}{N}.$$

Combining,

$$\big|F_n(\theta) - F(\theta)\big| \;\le\; \max\Big(\big|F_n(\theta_{i-1}) - F(\theta_{i-1})\big|,\, \big|F_n(\theta_i^-) - F(\theta_i^-)\big|\Big) \;+\; \frac{1}{N}.$$

Since F is the cumulative density function of a probability measure, its atom set $\{x : F(x) - F(x^-) > 0\}$ is at most countable: for each $k \ge 1$, atoms of mass $\ge 1/k$ are at most k in number, and the atom set is the countable union over k of these finite sets. The function $|F_n - F|$ is càdlàg, and its discontinuities are contained in the (at most countable) union of the atom set of F and the data points $\{x_1, \dots, x_n\}$. Consequently $\sup_{\theta \in \mathbb{R}} |F_n(\theta) - F(\theta)|$ equals its supremum over a countable set and is a well-defined measurable random variable.

Taking the supremum over $\theta \in \mathbb{R}$,

$$\sup_{\theta}\big|F_n(\theta) - F(\theta)\big| \;\le\; \max_{1 \le i \le N} \max\Big(\big|F_n(\theta_{i-1}) - F(\theta_{i-1})\big|,\, \big|F_n(\theta_i^-) - F(\theta_i^-)\big|\Big) \;+\; \frac{1}{N}, \tag{17}$$

since at $\theta = \theta_0 = -\infty$ and $\theta = \theta_N = +\infty$ both F_n and F vanish or equal 1 identically.

Fix $t \in (0, 1]$ and choose $N = \lceil (q+1)/t \rceil$. Since $q > 2$ and $t \le 1$, $N \ge q+1 \ge 3$. Then $1/N \le t/(q+1)$, so $t - 1/N \ge qt/(q+1)$, and $N - 1 \le (q+1)/t$.

By (17), the event $\{\sup_\theta |F_n - F| > t\}$ is contained in the union, over $i \in \{1, \dots, N\}$, of the events $\{|F_n(\theta_{i-1}) - F(\theta_{i-1})| > qt/(q+1)\}$ and $\{|F_n(\theta_i^-) - F(\theta_i^-)| > qt/(q+1)\}$.

Two of these events occur with probability 0 (at $\theta_0 = -\infty$ and at $\theta_N = \infty$). Boole's inequality combined with (15) (each applied with $qt/(q+1)$ in place of t) results in

$$
\begin{aligned}
P\left(\sup_\theta |F_n - F| > t\right) &\le 2(N-1) \cdot \frac{A_q}{n^{q/2}\,(qt/(q+1))^q} \;=\; \frac{2(N-1)\,(q+1)^q\,A_q}{q^q\,n^{q/2}\,t^q} \\
&\le \frac{2\,\big((q+1)/t\big)\,(q+1)^q\,A_q}{q^q\,n^{q/2}\,t^q} \qquad \big(\text{using } N-1 \le (q+1)/t\big) \\
&= \frac{2\,(q+1)^{q+1}\,A_q}{q^q\,n^{q/2}\,t^{q+1}}.
\end{aligned}
$$

Noticing that $(1+q)^{q+1}/q^q \le (q+1)e$, setting $C(q, p, K_0) := 2e(q+1)A_q$ completes the proof, since A_q depends only on q and on the mixing-tail sum, which under (9) is bounded by a quantity depending only on p and K_0. □

According to Theorem 2, generalization of Theorem 1 to dependent data with dependencies conforming with polynomial mixing, could be achieved by replacing exponential tails with the ones in the rhs of (12). Doing so will loosen then confidence bounds. Yet, the overall principle of learnable inference of relevant probabilities would remain.

C Details of building AI ontology calibrators

In the following sections, we provide details on the ontologies for which AI calibrators were built.

C.1 Ontology 1: BF

C.1.1 BF programs-outputs generation

Each BF program consists of commands represented by one of eight symbols, detailed in Table 3.

The initial training data was generated using a genetic algorithm that evolves programs based on their output. This approach enables the generation of diverse program candidates.

We also implement a custom BF interpreter with the following specifications:

1. Program outputs are represented as ASCII codes (values $0 - 255$). The interpreter truncates output sequences exceeding 256 codes.
2. All output values including zeros produced during execution are preserved in the output sequence.
3. Programs entering infinite loops during execution are terminated, and a special token `<InfLoop>` is outputted.

Table 3: BF commands' specifications. Our model uses all commands except input (`,`) and predicts an end-of-sequence token for program termination.

Symbol(s)	Command description
`>` / `<`	Increment / decrement the data pointer by one.
`+` / `-`	Increment / decrement the byte at the data pointer by one.
`.`	Output the byte at the data pointer.
`,`	Accept one byte of input, storing its value in the byte at the data pointer.
`[` / `]`	If the byte at the data pointer is zero / non-zero, then instead of moving the instruction pointer forward to the next command, jump it forward to the command after the matching `]` / `[` command.

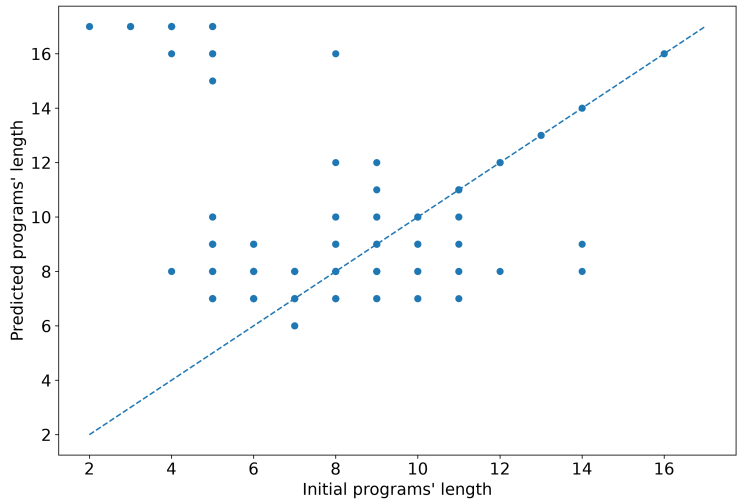


Figure 3: The dependence of initial and predicted programs lengths of new BF programs (that differ from the training data). Among $1,490$ new programs, 198 programs are shorter and $1,061$ are longer than in the initial data, and 231 programs of the same length as in the initial data.

C.1.2 BF PROGRAMS PREDICTION

Model's training and the calibrator application details For model training, we tokenize BF programs by mapping each symbol to a digit and introducing special tokens: beginning-of-sequence ($\texttt{BOS} = 1$), end-of-sequence ($\texttt{EOS} = 2$), and padding ($\texttt{PAD} = 0$). Our programs do not contain the symbol ",,"; however, our model is also expected to generate $\texttt{EOS}$ for each sequence. All programs are padded to length 18 using the $\texttt{PAD}$ token.

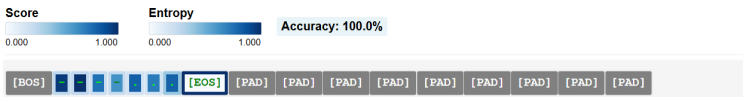
Our Transformer architecture comprises a 12-layer encoder followed by two linear projection layers with interleaved dropout and layer normalization (applied twice), and a 12-layer decoder followed by two linear layers with three dropout-normalization stages. Sinusoidal positional encodings are applied to embeddings of both program tokens and output symbols. The training objective combines token-level cross-entropy loss with a set-level term that aggregates probability mass across all valid programs producing identical outputs. We train our Transformer model autoregressively.

We apply the calibrator to the entire dataset and analyse results for the first $10,000$ program-output pairs. In $2,328$ cases, the model generates programs *identical* to those in the training data. Surprisingly, in $1,490$ cases, it produces functionally equivalent *new* programs that yield identical outputs to the original training programs. We count these as correct predictions since they satisfy the input-output specification. A length comparison of these novel programs is presented in Figure 3.

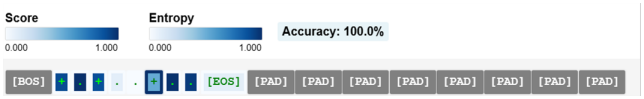
As a result, we obtain $3,818$ correctly generated programs and $6,182$ incorrect ones.

Predicted programs analysis As noted above, our trained Transformer model generates programs that differ from training examples while producing identical outputs. Figure 3 shows the length relationship between initial and predicted programs for these $1,490$ cases. Among them, $1,061$ programs (71.2%) exceed their training counterparts in length (by up to 15 symbols). Notably, we also observe 231 structurally distinct programs of identical length (15.5%), and 198 shorter programs (13.3%) reduced by a maximum of 6 symbols.

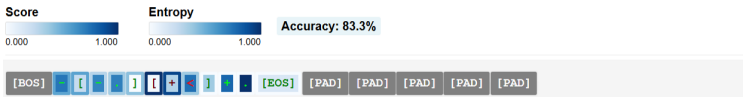
We further analyse the relationship between entropy and the model's internal statistics. First, we visualise entropy and the calibrator's probability distributions for presented programs. Figure 4 shows examples for both correctly and incorrectly predicted programs, demonstrating that confidence and entropy may exhibit an inverse linear relationship (e.g., Figures 4a and 4d) or no consistent correlation (e.g., Figures 4b and 4c).



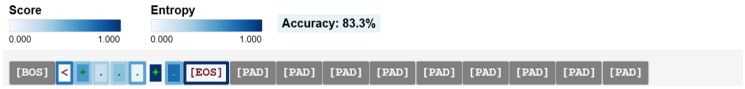
(a) The example of correctly predicted program. Confidence scores and entropy correlate.



(b) The example of correctly predicted program. Confidence scores and entropy do not correlate.



(c) The example of incorrectly predicted program: the initial program is `-[-.][+<]+.` while the predicted program is `-[-.]+]]]+..` Confidence scores and entropy do not correlate.



(d) The example of incorrectly predicted program: the initial program is `<+...+.` while the predicted program is `++...+.-`. Confidence scores and entropy correlate.

Figure 4: A symbol-level analysis of model confidence and uncertainty (entropy) for program predictions. Each token is visualized with triple encoding: (1) symbols' foreground colour indicates prediction correctness (green for correct, red for incorrect symbols), (2) background intensity represents confidence scores, and (3) the border colour encodes predictive entropy.

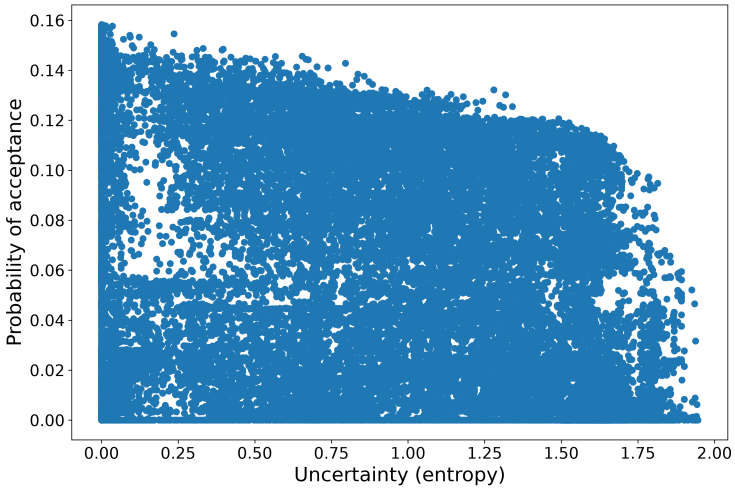


Figure 5: The dependence of the probability acceptance on the entropy. The Pearson correlation equals to -0.37 with p-value < 0.0001.

Second, we compute the Pearson correlation coefficient between entropy and the calibrator's probability across $10,000$ program samples. Figure 5 reveals a weak negative linear correlation ($r = -0.37$, $p < 0.0001$).

We also analyse the calibrator's probability in more detail. Figure 6 characterizes the calibrator's probability distributions. The histogram for incorrectly predicted programs (left) exhibits a sharp unimodal peak at near-zero probability ($p \approx 0$, density ≈ 55), indicating the model's low confidence in erroneous outputs, with only weak, diffuse elevation (density < 15) elsewhere in the range.

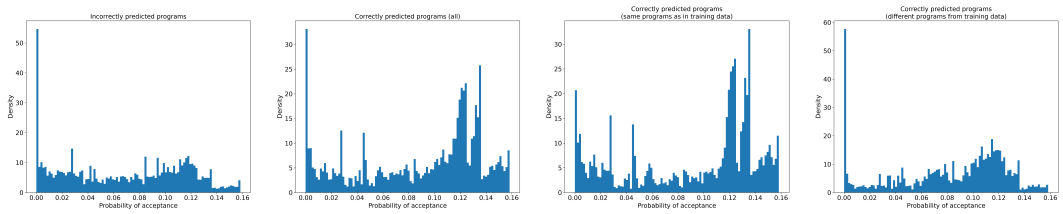


Figure 6: The histograms of calibrator's probabilities.

Correctly predicted programs (all, second panel) instead show a clear bimodal distribution: a peak at $p \approx 0$ (density ≈ 33) and a second mode around $p \approx 0.12 - 0.135$ reaching a comparable density of ≈ 25, indicating two distinct confidence regimes among correct predictions. Decomposing this by memorization status reveals a clear asymmetry. For programs identical to those seen during training (third panel), the high-confidence mode is in fact the dominant feature, peaking at $p \approx 0.135$ with density ≈ 33 and exceeding the near-zero peak (density ≈ 20) by roughly $1.6\times$ – the calibrator is markedly more confident when the correct program matches training data. For correct predictions on programs structurally different from training (right panel), the pattern reverses: the near-zero peak becomes the single largest peak in the entire figure (density ≈ 58), roughly $2.9\times$ its own high-confidence region (density ≈ 20 at $p \approx 0.115$) and the corresponding near-zero peak in the training-identical subset (third panel).

This divergence implies the model develops specialized confidence heuristics for generating functionally equivalent but structurally novel programs. The consistent concentration of all distributions in the low-probability regime ($p < 0.16$) further indicates that high-confidence predictions are rare, aligning with the weak entropy-probability correlation observed in Figure 5.

C.2 Ontology 2: Math problems

To assess the correctness of math problem solutions, we employ two techniques:

1. *the LLM Judge* – motivated by the impracticality of manual annotation at scale (Teo & Shari, 2025; Huber & Guggenberger, 2022). We use Qwen2.5-32B (Team, 2024) as the judge. Each answer (from both the expert and the student) is split into double-newline-delimited fragments ("sentences"), which are assembled into a structured prompt asking the checker to align each sentence from the expert's trace to one or more sentences in the student's trace, assigning a confidence of "high", "medium", or "low" to each correspondence. The full prompt is provided in Appendix C.2.1). Task correctness is then computed as the average confidence score (1, 0.5, or 0, respectively) across all associations and the final verdict is "correct" (label 1) if the result ≥ 0.5 otherwise "incorrect" (label 0), following a similar method to (Kahng et al., 2024).
2. *the ProcessBench annotation* – which reuses the benchmark's original evaluation protocol (Zheng et al., 2025). Answers are split into sentences using the same scheme as above and fed into the ProcessBench evaluator, which identifies the index of the first erroneous sentence in the student's trace, or signals a fully correct solution. We use DeepSeek-V3.2 as the annotator given its strong performance on math tasks (DeepSeek-AI, 2025b). Inference was done locally, on NVidia H100 80GB GPUs, full annotation took six hours per dataset.

C.2.1 Prompt for LLM-based checker of the correctness of math problems' solving

The full prompt for comparing the student's model math problem solving with the expert's via Qwen2.5-32B is the following:

> You are analyzing two chain-of-thought (CoT) reasoning traces for the same math problem. One is from a large model and one is from a small model.

Your task is to map which sentences in the large CoT correspond to which sentences in the small CoT. Some sentences may correspond to multiple sentences, or some may have no clear correspondence.

Question:
{text_of_task_prompt}

Large Model CoT (numbered sentences):
{expert_list_of_sentences}

Small Model CoT (numbered sentences):
{student_list_of_sentences}

Please output a JSON array where each element maps a large_cot sentence index to matching small_cot sentence indices. Use this format:

```
[
  {{
    "large_idx": 0,
    "small_idx": [0, 1],
    "confidence": "high",
    "rationale": "Both introduce the problem and set up variables"
  }},
  {{
    "large_idx": 1,
    "small_idx": [2],
    "confidence": "medium",
    "rationale": "Similar equation setup but with different
    clarity"
  }}
]
```

Confidence levels: "high" (clear correspondence), "medium" (likely but not certain), "low" (weak correspondence)

Output only the JSON array, no additional text.

C.2.2 Results of labeling strategies for math tasks

In this work, we consider only $6,845$ math problems for which we have both labelings: one from an LLM-based checker and one from ProcessBench. The LLM-based checker labeled $6,453$ problems as correctly solved, while ProcessBench labeled only $3,452$. Figure 7 shows that neither labeling yields separable classes under PCA. Nevertheless, we consider the ProcessBench labeling as more reliable, since the LLM-based checker marked almost all problems as correct. Therefore, for medical question answering we use the ProcessBench labeling.

C.3 Ontology 3: medical-related question answering

For medical question answering, we provide the possible addition highlight of six parts of the models' outputs:

1. **Decision** – a clinical, surgical, or scientific action to take or avoid.
2. **Reasoning** – a pivotal logical step that drives the conclusion.
3. **Conclusion** – the final answer or diagnosis, expressed as a concept rather than a letter.
4. **Caution** – a risk, contraindication, warning, or negation that prevents a wrong action.
5. **Mechanism** – a biological, pharmacological, or molecular mechanism on which the answer hinges.
6. **Elimination** – reasoning that rules out a wrong option (critical for NOT/incorrect/except questions).

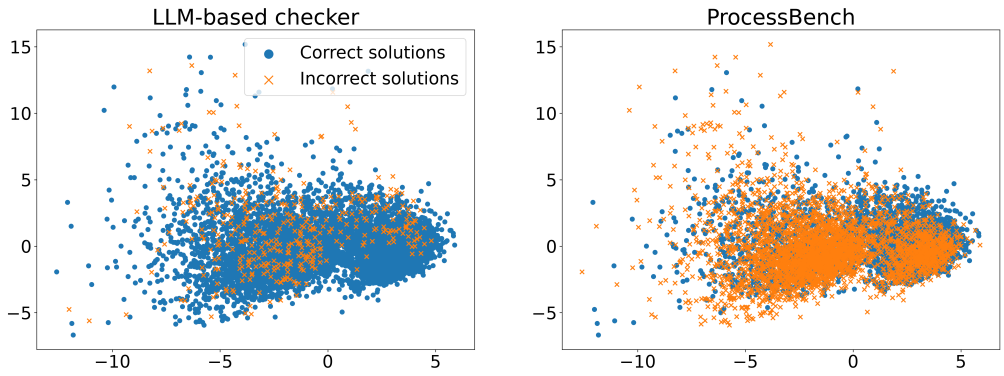


Figure 7: PCA visualization of both labeling strategies.

To produce these highlights, we use MedGemma (`google/medgemma-1.5-4b-it`) with the following prompt:

You are a precise annotation judge for science and medicine questions.

Your task is to read a question and a model-generated answer, then identify spans of text that are SEMANTICALLY PIVOTAL – meaning that changing or removing them would either reverse the meaning, lead to the wrong conclusion, or cause a clinically or scientifically dangerous error.

The questions come from a broad range of ontologies:
- Molecular and cell biology (mechanisms, pathways, organelles, proteins)
- Genetics and genomics (mutations, inheritance, sequencing, gene regulation)
- Physiology and anatomy (organ systems, hormones, homeostasis)
- Neuroscience (synapses, neurotransmitters, brain function)
- Ecology and evolution (selection, population dynamics, symbiosis)
- Clinical medicine (diagnosis, treatment, surgical indications, patient management)
- Pharmacology (drug targets, mechanisms of action, resistance)
- Research methods (PCR, sequencing, microscopy, assays)
- Healthcare systems and policy (EHR, public health, technology)

Semantically pivotal spans include:
- Direction or magnitude words whose reversal flips the answer:
”increase” vs ”decrease”, ”inhibits” vs ”activates”, ”proximal” vs ”distal”
- A negation that inverts the meaning of a claim:
”not”, ”except”, ”unlikely”, ”absent”
- A specific named entity the answer critically depends on:
a drug name, enzyme, gene, pathway, or anatomical structure
- A key quantity, threshold, or relationship: ”above pH 7”, ”half-life of 8 days”, ”binds competitively”
- The clause that eliminates a distractor in NOT/incorrect/except questions:
the reasoning that rules out a wrong option, not just the option letter

Do NOT highlight:
- Bare answer labels in isolation (”answer D”, ”option C”) – highlight the reasoning or concept that justifies the choice instead
- Connector words alone (”because”, ”therefore”, ”however”) – highlight the clause they introduce if that clause is pivotal, not the connector itself
- Background or contextual sentences that do not affect the correctness of the answer if removed

If the answer contains only a bare label with no reasoning, return an empty highlights list and note this in the summary.

Return ONLY a valid JSON object (no markdown fences, no extra text):

```
{
  "highlights": [
    {
      "span":   "<exact substring copied character-for-character
      from the answer>",
      "type":   "<one of: decision | reasoning | conclusion |
      caution | mechanism | elimination>",
      "reason": "<one sentence explaining why changing this span
      would make the answer wrong>"
    }
  ],
  "summary": "<2-3 sentence summary of what is semantically pivotal
  in this answer>"
}
```

Type definitions:
decision - a clinical, surgical, or scientific action to take or avoid
reasoning - a pivotal logical step that drives the conclusion
conclusion - the final answer or diagnosis, expressed as a concept not a letter
caution - a risk, contraindication, warning, or negation that prevents a wrong action
mechanism - a biological, pharmacological, or molecular mechanism the answer hinges on
elimination - reasoning that rules out a wrong option (critical for NOT/incorrect/except questions)

Rules:
* ”span” must be an exact character-for-character substring of the answer text
* Highlight every span that is semantically pivotal – neither more nor less
* Do not add spans to reach a minimum count
* Do not drop spans to stay under a maximum count
* A short answer may have 1-2 highlights; a long reasoning chain may have more
* Spans may overlap; that is fine
* Do NOT include any text outside the JSON object.”

Figure 8 shows an example containing three types of highlights: reasoning (”providing the first step in glucose catabolism”), conclusion (”B: Glycolysis”), and mechanism (”takes place in the cytoplasm” and ”converts glucose into pyruvate”).

These highlights help identify which parts of a reasoning trace deserve closer attention. While a mistake on the example shown here would be inconsequential, this is not the case for many other medical questions, where an incorrect answer could cause real harm. These highlights can also be used in the labeling process: for example, an answer may be marked incorrect if an error appears in any span belonging to one of these six categories.

D Relations of derived indicators for classifiers

D.1 Analysis of indicators and baselines on all ontologies

To analyse the relations between the 15 indicators and six baselines used for training classifiers, we calculate pairwise Spearman correlations, including correlations with the target label for all our domains, labelings, and students. Figure 9 presents feature pairs with a statistically significant (p-value ≤ 0.05) absolute correlation coefficient exceeding 0.6 for all models and ontologies. The calibrator-based probability indicators provide signals largely independent from the entropies- and softmaxes-based indicators. In general, the following indicators provide their unique signals with no correlations with others: $std(p), std(e),\ PCC, SCC,\ MSP, \max(TE), \min(TP)$. In this list, we

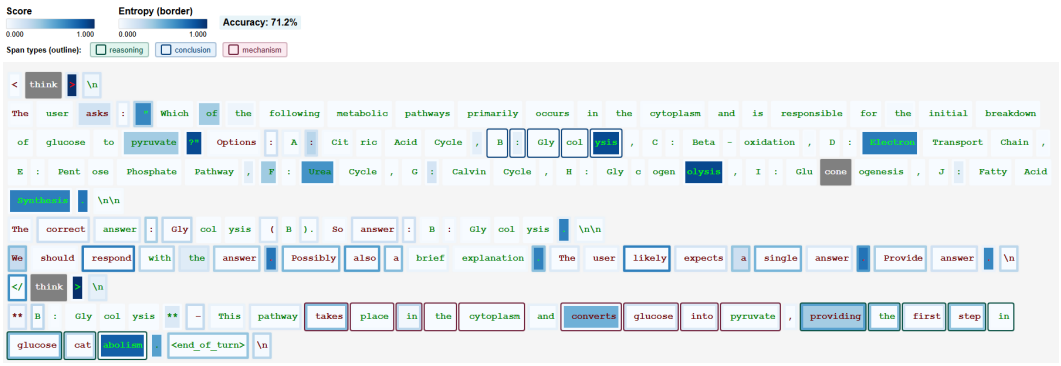


Figure 8: Visualization comparing the expert's and the student's answers to the following question: which of the following metabolic pathways primarily occurs in the cytoplasm and is responsible for the initial breakdown of glucose to pyruvate? A: Citric Acid Cycle; B: Glycolysis; C: Beta-oxidation; D: Electron Transport Chain; E: Pentose Phosphate Pathway; F: Urea Cycle; G: Calvin Cycle; H: Glycogenolysis; I: Gluconeogenesis; J: Fatty Acid Synthesis.

did not include $\bar{p}$ and $\bar{G}_p$ as they correlate with each other, which is expected (will be additionally mentioned below).

The obtained correlations can be divided into the following groups:

1. **Mathematically trivial**: the correlation coefficient is not an empirical finding but a mathematical identity or a consequence of a monotone transformation. The correlated pairs of (E_s, PP_s), (E_p, PP_p), $\left(LM\ PP, \frac{1}{\bar{G}_{TP}}\right)$ are the monotone transformations.

2. **Information theory**: the correlation is fully predicted by information theory. Two distinct sub-patterns appear here. First, *confidence-proxy convergence*: $\left(\bar{s}, \overline{TP}\right)$, $\left(\bar{G}_s, \overline{TP}\right)$, and $\left(\bar{G}_s, \bar{G}_{TP}\right)$. $\bar{s}$ aggregates the probability of the single most likely token at each position, regardless of what token was actually produced, while $\overline{TP}$ aggregates the probability the model assigns to the token it actually generated; the two are identical under greedy decoding and remain strongly correlated otherwise, since sequences where the model's best guess is confident also tend to be sequences where the realised token was highly probable. Second, the *entropy–probability inverse relationship*: $\left(\bar{e}, \frac{1}{\bar{s}}\right)$, $\left(\bar{e}, \frac{1}{\bar{G}_s}\right)$, $\left(\bar{s}, \frac{1}{\bar{G}_e}\right)$, $\left(\bar{G}_s, \frac{1}{\bar{G}_e}\right)$, $\left(\bar{G}_e, \frac{1}{\overline{TP}}\right)$, and $\left(\bar{e}, \frac{1}{\overline{TP}}\right)$. A peaked (low-entropy) distribution necessarily places more mass on its top token, so any entropy-side statistic is inversely related to any probability-side statistic; these six pairs are the same relationship instantiated across the arithmetic/geometric aggregation choices on each side.

3. **Aggregation robustness**: $\left(\bar{e}, \bar{G}_e\right)$, $\left(\bar{s}, \bar{G}_s\right)$, and $\left(\bar{p}, \bar{G}_p\right)$ confirm that switching from the arithmetic to the geometric mean of the same underlying per-token quantity (entropies, softmaxes, or calibrator probabilities) does not change the ranking of instances, i.e. the results are not artefacts of the choice of aggregator.

4. **Bounded Support Artifact**: the correlation arises from the bounded range $(1/K, 1]$ that top-1 softmax values are confined to, not from a deep property of model behavior. Since each top-1 softmax value $s_t \in (1/K, 1]$ is bounded above, sequences that are consistently confident cluster near the ceiling and sequences that are consistently uncertain cluster near the floor $1/K$; in both regimes the values are compressed and $std(s)$ is low, so $\bar{s}$ and $std(s)$ move in opposite directions: $\left(\bar{s}, \frac{1}{std(s)}\right)$. Only an unstable mix of confident and uncertain positions yields a high $std(s)$. The pair $\left(std(s), \frac{1}{\bar{G}_s}\right)$ is not an independent finding: it follows by transitivity from $\left(\bar{s}, \frac{1}{std(s)}\right)$ combined with the aggregation-robustness identity $\bar{G}_s \sim \bar{s}$.

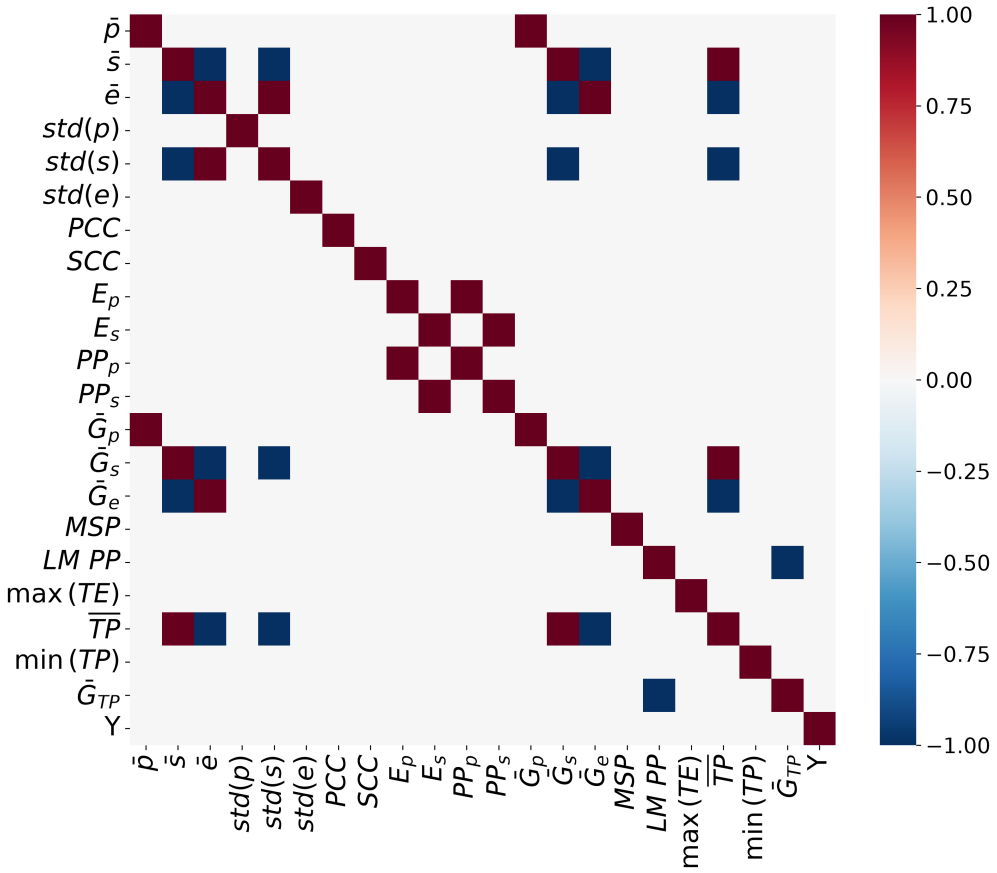


Figure 9: Statistically significant strong Spearman correlations (absolute coefficient > 0.6, p-value ≤ 0.05) between the 15 classifier indicators and the target label, aggregated across all ontologies and models. Only pairs for which all correlations satisfy both criteria are shown. The target labels are presented with Y.

5. **Surprising / Non-Obvious**: within a single token position, a more peaked softmax distribution mechanically implies both a higher top-1 value and a lower entropy – a negative relationship. The non-obvious finding is that, across instances, the sign flips: $(\bar{e}, std(s))$ is *positively* correlated – instances that are uncertain on average (high $\bar{e}$) are also the instances whose top-1 confidence is most *unstable* across positions (high $std(s)$), rather than uniformly low, reflecting instance-level difficulty where some positions remain confident while others do not, inflating the variance of s_t. This aligns with a second, distinct pair, $\left(std(s), \frac{1}{\overline{TP}}\right)$: instances with high positional variability in top-1 confidence also show low average softmaxes on the tokens actually produced, i.e. the model is simultaneously erratic in its best guess *and* less committed to its actual output. Combined, these compose into $\left(\bar{e}, \frac{1}{\overline{TP}}\right)$: average per-token uncertainty positively correlated with low realised-token confidence, mediated by $std(s)$.

In can also be noted that none of the provided indicators correlate with the target label.

D.2 Analysis of indicators including verbalized confidence on medical ontology

Some papers (Kadavath et al., 2022; Lin et al., 2022) report a correlation between generation correctness and the model's self-assessment labels. To verify this claim, we additionally prompt

our students on the medical ontology data to label the correctness of their answers and assess their confidence on a scale from 1 to 10.

For self-correctness assessment, we tried to follow the technique from (Kadavath et al., 2022) and reuse their prompt; however, our models produced outputs whose verdicts were almost impossible to parse. Therefore, for self-correctness assessment we used the following prompt:

> I will give you a question and a possible answer with reasoning traces. Please output whether the possible answer is correct or not. Output only a single word: "correct" or "wrong".
>
> Question:
> {text_of_prompt}
>
> Possible Answer:
> {text_of_student_response}
>
> The possible answer is:

The prompt used for self-confidence assessment is as follows:

> I will give you a question and your answer with reasoning traces. Please output whether you are confident in the provided answer. Output only a single value from 1 to 10, where 1 means you are not confident and 10 means you are completely confident.
>
> Question:
> {text_of_prompt}
>
> Your Answer:
> {text_of_student_response}
>
> Confidence score is:

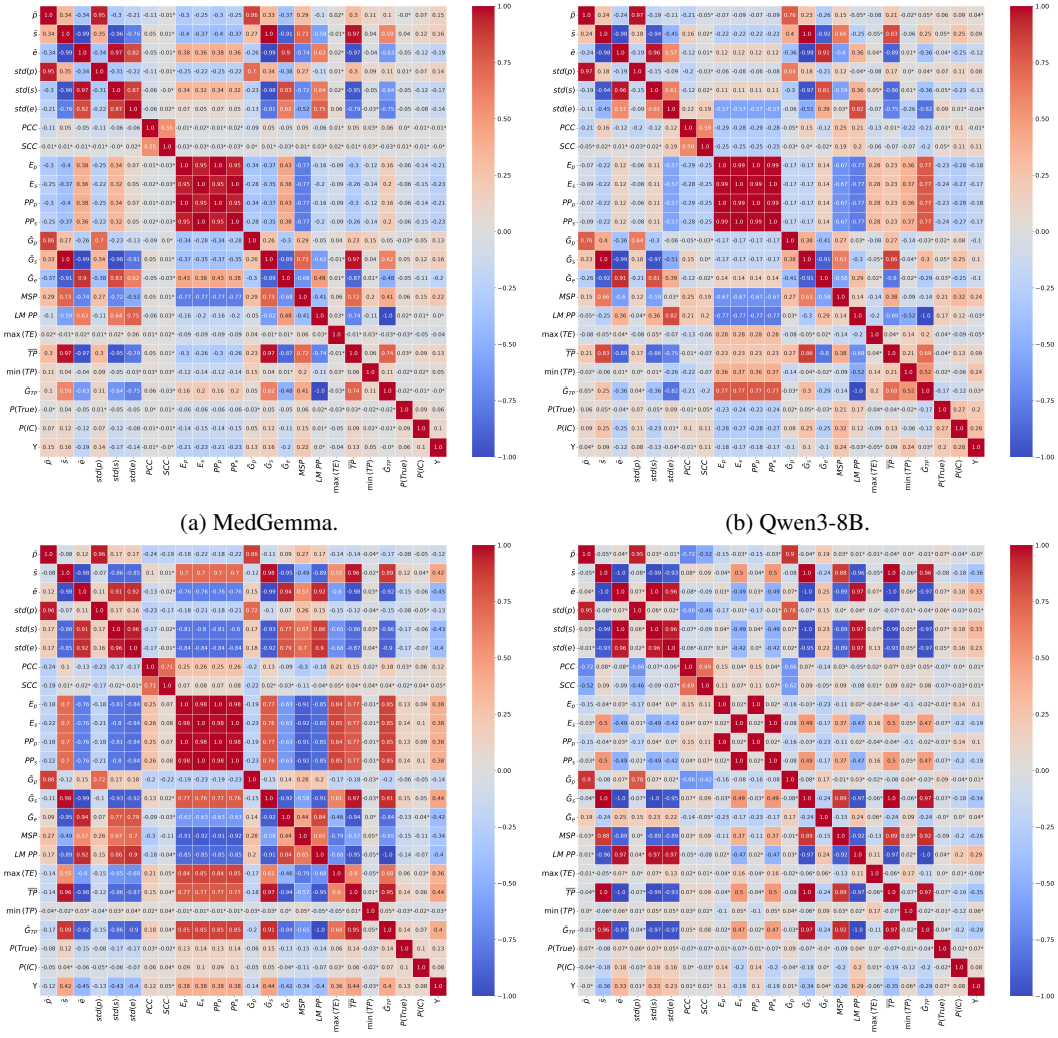
Figure 10 presents the correlation of indicators with baselines and self-assessment labels: assessment of response correctness ($P(True)$) and assessment of confidence in a response ($P(IC)$). For self-correctness and self-confidence assessment, we considered the responses labeled with ProcessBench. As shown in Figure10, only about half of such data have self-assessment labels for the considered students: MedGemma – 58.7% ($3,509$ responses with both self-assessment labels out of $5,978$ labeled with ProcessBench), Qwen3-8B – 40.8% ($1,308$ out of $3,207$), Qwen3-0.6B – 38.6% ($1,415$ out of $3,663$), and Qwen3.5-0.8B – 31.5% (615 out of $1,954$). Nevertheless, we only consider responses from which self-assessment labels can be easily obtained: if a response contained a label but its parsing was difficult, it was not included in our list. Overall, unlike what is reported in other papers, we do not observe strong correlations of either $P(True)$ or $P(IC)$ with Y.

E TECHNICAL DETAILS OF BUILT CLASSIFIERS

In Section 5 we provide the metrics of trained Logistic Regression (with default hyperparameters (with random_state= 123) on each of our 15 indicators, six baselines, and two cases of several indicators usage: all 15 indicators and the first 12 indicators (the set of all indicators without $\bar{G}_{\cdot}$-based). We additionally conduct the training of XGBoost with default hyperparameters (with random_state= 123) and MLP with the architecture presented in Figure 11 on all possible combinations of our 15 indicators. The classifiers were trained on an internal cluster, consisting of NVidia Tesla H200 141GB and NVidia RTX 4090 96GB GPUs. Total training time was around 96 hours for both ablations and the final training run.

For the MLP, we use $n = 1$ block, a hidden dimension of $H = 128$, and a dropout rate of $p = 0.2$.

All classifiers are trained on 20-folds cross-validation with stratification w.r.t. target labels. For XGBoost and MLP, we used weighted BCEWithLogitsLoss loss function (with pos_weight=



(a) MedGemma. (b) Qwen3-8B.

(c) Qwen3-0.6B. (d) Qwen3.5-0.8B.

Figure 10: Spearman correlation matrices for four students on the medical ontology data. ProcessBench labels are denotes as Y.

$N_{\text{zeros}}/N_{\text{ones}}$) for all the ontologies. For MLP, we additionaly choose 20% of training part on each fold for validation and normalise the data. We also use `AdamW` optimizer with learning rate $= 10^{-3}$ and weight decay $= 10^{-4}$, `ReduceLROnPlateau` scheduler maximizing F1 score with patience $= 5$, factor $= 0.5$, `EarlyStopping` with patience $= 7$ and delta $= 10^{-4}$. We train all out MLPs with 300 epochs for each features set.

Figure 12 shows the training process of XGBoost and MLP classifiers on the medical ontology with MedGemma. Although both classifiers achieve comparable ROC AUC scores on the test data, XGBoost exhibits clear overfitting, as its training score is substantially higher and is close to the ideal value of 1. We observe similar behavior across the other ontologies. Therefore, in the paper, we focus on MLP and in Section 5 we present only the results for the best MLPs.

F Transferability of classifiers

For a subset of indicators, we evaluate the transferability of all trained MLP-based classifiers across all ontology–model combinations. Figures 13 and 14 compares the classifiers' transfer across ontologies in two settings: using the set of traditional indicators with one replaced by our $\bar{p}$ and using the

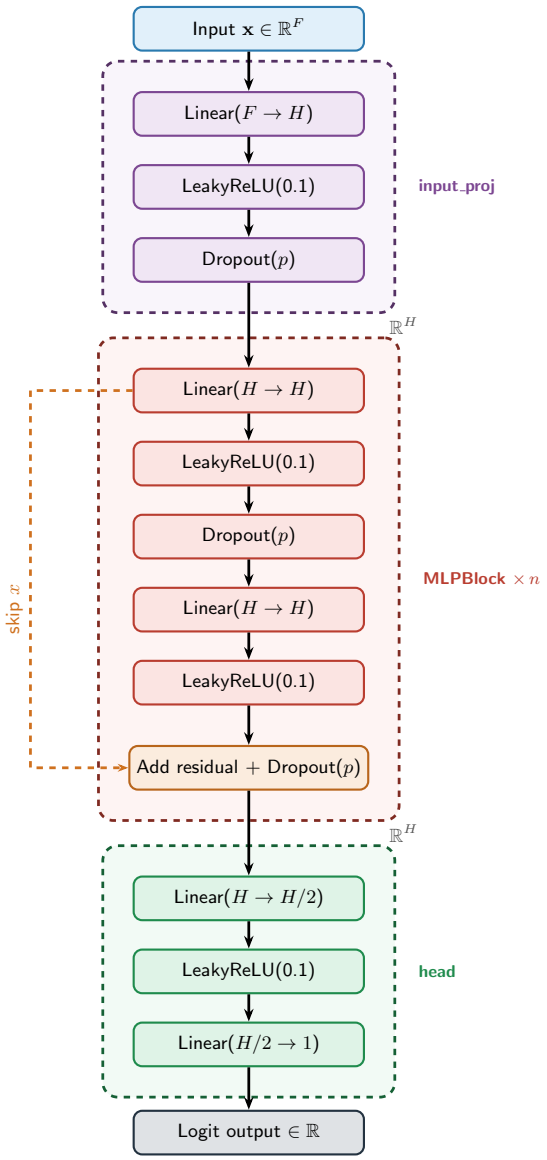
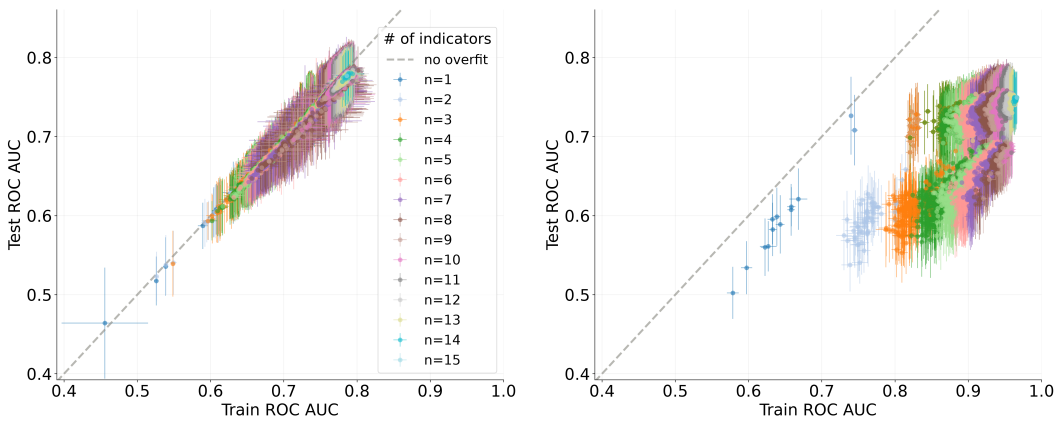


Figure 11: The scheme of the MLP architecture.

traditional indicators only; the same number of indicators is therefore used in both settings. As the figures show, replacing $\bar{G}_s$ with $\bar{p}$ improves the transfer quality in most cases, while in the cases where the traditional indicators perform better, the maximum difference in their favor is only 0.036 ROC AUC (0.052 F1).

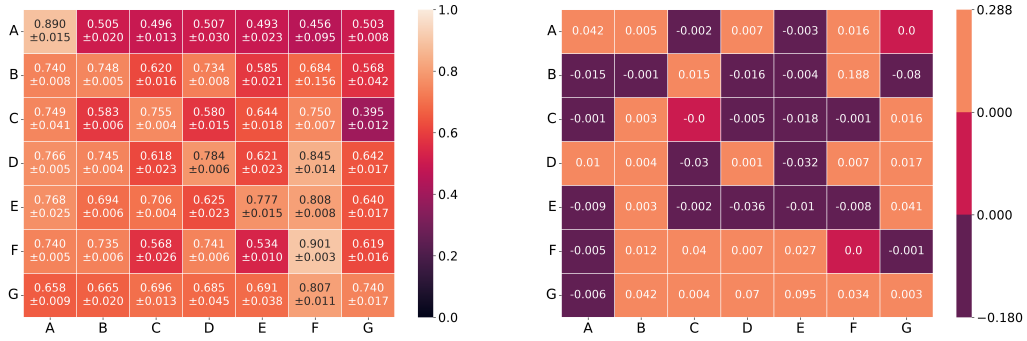
However, using more calibrator-based indicators does not necessarily improve transfer quality. Figure 15 provides the results for classifier transfer using the first 12 features ($\{\bar{p}, \bar{s}, \bar{e}, \quad std(p), std(s), std(e), \quad PCC, SCC, \quad E_p, E_s, \quad PP_p, PP_s\}$), calibrator-only indicators ($\{\bar{p}, std(p), E_p, PP_p, \bar{G}_p\}$), and calibrator-only and mixed indicators ($\{\bar{p}, std(p), PCC, SCC, E_p, PP_p, \bar{G}_p\}$). The results show that the indicator sets provided in Figure 15 have lower transferability than those provided in Figure 13, despite the fact that Figure 13a contains only one calibrator probability.



(a) MLP classifier.

(b) XGBoost classifier.

Figure 12: Training and testing ROC AUC of classifiers trained on MedGemma features.



(a) $\{\bar{\mathbf{p}}, \bar{s}, \bar{e},\ std(s), std(e),\ E_s,\ PP_s,\ \bar{G}_e\}$.

(b) Difference between mean ROC AUC scores of $\{\bar{\mathbf{p}}, \bar{s}, \bar{e},\ std(s), std(e),\ E_s,\ PP_s,\ \bar{G}_e\}$ and mean F1 scores of $\{\bar{\mathbf{G}}_\mathbf{s}, \bar{s}, \bar{e},\ std(s), std(e),\ E_s,\ PP_s,\ \bar{G}_e\}$.

Figure 13: Aggregated transfer results (method IDs as in Table 1). Each of the 20 classifiers from 20-fold cross-validation is transferred to the whole data of each ontology; we report the mean and standard deviation of the **ROC AUC score**. In both heatmaps, rows denote the training ontology and columns the test ontology. **Left:** classifiers using the indicators given in the subcaption. **Right:** difference in mean ROC AUC between classifiers on $\{\bar{\mathbf{p}}\} \cup I$ and on $\{\bar{\mathbf{G}}_\mathbf{s}\} \cup I$ (traditional indicators only), where $I = \{\bar{s}, \bar{e}, std(s), std(e), E_s, PP_s, \bar{G}_e\}$; 25 of 49 (51 %) transfers favor the indicators with our $\bar{p}$.

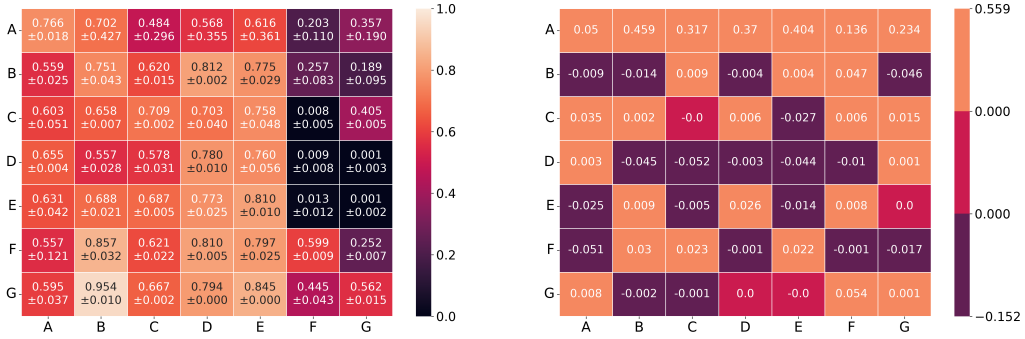
G Licenses on used data, models and software

In the paper the following models, software and licences are used:

1. Qwen-2.5 – Apache 2.0 license;
2. DeepSeek V3.2 – MIT license;
3. MedGemma-2.5-4B – health-ai-developer-foundations license;
4. SGLang inference server – Apache 2.0 license;
5. XGBoost algorithm – Apache 2.0 license;

H Societal impacts

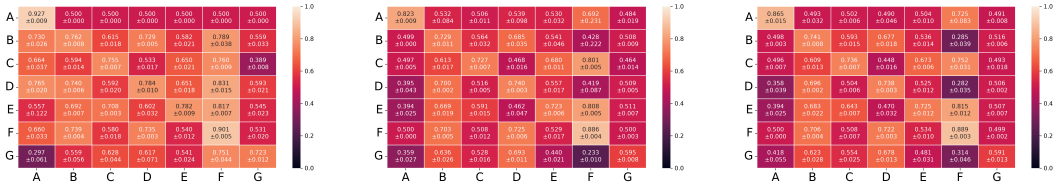
The proposed framework aims to improve the reliability of generative AI systems by providing lightweight indicators of atypical and potentially incorrect outputs, which could be valuable in high-



(a) $\{\bar{\mathbf{p}}, \bar{s}, \bar{e},\ std(s), std(e),\ E_s,\ PP_s,\ \bar{G}_e\}$.

(b) Difference between mean F1 scores of $\{\bar{\mathbf{p}}, \bar{s}, \bar{e},\ std(s), std(e),\ E_s,\ PP_s,\ \bar{G}_e\}$ and mean F1 scores of $\{\bar{\mathbf{G}}_\mathbf{s}, \bar{s}, \bar{e},\ std(s), std(e),\ E_s,\ PP_s,\ \bar{G}_e\}$.

Figure 14: Aggregated transfer results (method IDs as in Table 1). Each of the 20 classifiers from 20-fold cross-validation is transferred to the whole data of each ontology; we report the mean and standard deviation of the **F1 score**. In both heatmaps, rows denote the training ontology and columns the test ontology. **Left:** classifiers using the indicators given in the subcaption. **Right:** difference in mean F1 between classifiers on $\{\bar{\mathbf{p}}\} \cup I$ and on $\{\bar{\mathbf{G}}_\mathbf{s}\} \cup I$ (traditional indicators only), where $I = \{\bar{s}, \bar{e}, std(s), std(e), E_s, PP_s, \bar{G}_e\}$; 26 of 49 (53 %) transfers favor the indicators with our $\bar{p}$.



(a) The first 12 indicators.

(b) Calibrator-only indicators.

(c) Calibrator-only and mixed indicators.

Figure 15: Aggregated transfer results (method IDs as in Table 1). Each of the 20 classifiers from 20-fold cross-validation is transferred to the whole data of each ontology; we report the mean and standard deviation of the **ROC AUC score**. In all heatmaps, rows denote the training ontology and columns the test ontology.

stakes ontologies such as medicine, software engineering and decision support. If used appropriately, such methods may highlight fluent but unreliable generations and help route unreliable cases to stronger verification pipelines. At the same time, the method should not be interpreted as a guarantee of factual correctness: false negatives may induce false confidence, while false positives may suppress useful outputs. These risks are particularly important in medical ontology, where detector scores should serve only as additional evidence rather than as expert judgment. Careful ontology-specific calibration, transparent reporting of detector non-confidence, and validation under distribution shift are necessary before deployment.