SEMANTIC UNCERTAINTY QUANTIFICATION NEEDS FACTUAL EQUIVALENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Semantic uncertainty quantification for large language models rests on a common template: sample several answers, measure how much they agree, and treat disagreement as uncertainty. We first formalize this template as two separate roles: an operator that compares two answers, and an aggregator that combines all pairwise comparisons into a scalar. Existing methods differ almost entirely in how they aggregate, while taking the operator off the shelf, typically an NLI model or a generic sentence encoder. We show that this reliance on off-the-shelf operators is the primary bottleneck of semantic UQ: they do not accurately measure factual equivalence of multiple answers to the same question. We resolve this with a deliberately simple recipe: a single encoder trained contrastively to isolate the targeted fact, utilizing synthetic data generated by an LLM and dataset both disjoint from all evaluation settings. Integrating the resulting operator into existing methods improves performance on 120 of 126 evaluation settings (95%) spanning 18 model–dataset combinations across language and vision-language models. The best variant reaches 0.76 mean AUROC against 0.68 for the strongest baseline, while replacing the quadratic cross-encoder comparisons of entailment-based operators with one encoder pass per answer. The uniformity of the improvement supports the view that the operator, not the aggregator, is the limiting factor. The same operator also improves single generation token-level estimators: the norm it assigns to each token measures how much that token bears on the answer, and reweighting token log-likelihoods accordingly sharpens the estimate.

1 INTRODUCTION

Despite their overwhelming popularity, Large Language Models (LLMs) and Large Vision-Language Models (LVLMs) are still prone to *hallucinations*: they generate fluent, confident answers that are not grounded in their input Ji et al. (2023); Liu et al. (2024a). Knowing when an output should not be relied upon has therefore become essential, which is the role of uncertainty quantification (UQ) Hernandez-Lobato & Adams (2015); Gal & Ghahramani (2016).

Early token-level methods operate directly on autoregressive token probabilities Malinin & Gales (2021); Aichberger et al. (2024). However, these measures conflate lexical variation with semantic variation. Given the question *"What is the capital of France?"*, a model may output *"The capital of France is Paris."* (A1), *"The capital of France is Madrid."* (A2), or *"It's Paris."* (A3) (Fig. 1, top row). A1 and A3 agree on the fact, yet receive different token-level scores because they differ in wording and structure. A model that is entirely certain of the correct fact can thus appear highly uncertain simply due to lexical diversity.

Semantic UQ methods resolve this by evaluating generations in a space of meaning rather than vocabulary Farquhar et al. (2024). As illustrated in Fig. 1, they share a common template: sample N answers, compare them pairwise, and aggregate the resulting comparison structure into a scalar uncertainty score. We formalize this template as $U = F(\boldsymbol{S})$, where an *operator* s scores pairwise semantic equivalence to form an $N \times N$ matrix $\boldsymbol{S}$, and an *aggregator* F summarizes its structure. Prior work has focused almost exclusively on designing sophisticated aggregators F: taking the entropy of semantic clusters (SE, CAE) Farquhar et al. (2024), computing von Neumann entropy over graph Laplacians (KLE) Nikitin et al. (2024), or analyzing the spectrum of a similarity matrix (COS) Chen et al. (2024). In every case, the operator s is treated as a secondary design choice. Prior

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.

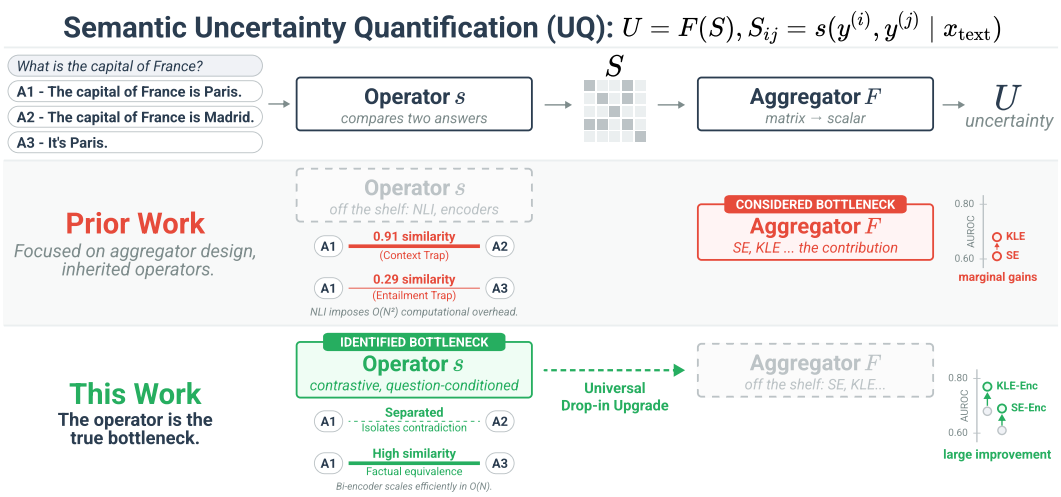

Figure 1: **The Operator Bottleneck in Semantic UQ.** Semantic uncertainty methods decompose into a pairwise comparison operator (s) and a structural aggregator (F). While prior work focuses on designing sophisticated aggregators, they inherit off-the-shelf operators that suffer from misaligned training objectives or severe scaling bottlenecks. We identify the operator as the true bottleneck and introduce a contrastively trained, question-conditioned bi-encoder. Serving as a universal drop-in replacement, our learned operator universally improves the performance of existing aggregators at a fraction of the computational cost of NLI baselines.

work reports that switching between off-the-shelf operators yields limited improvements Farquhar et al. (2024), which we argue occurs not because the choice is unimportant, but because none of these operators is structurally suited for the task.

This reliance on off-the-shelf operators is the primary bottleneck of semantic UQ. As illustrated in the "Prior Work" row of Fig. 1, when existing operators evaluate generated text, they suffer from severe distribution shifts and misaligned objectives. External sentence encoders evaluate general topical overlap, blinding them to targeted factual contradictions (e.g., scoring *"The capital is Paris"* and *"The capital is Madrid"* as highly similar). Internal hidden states are optimized for autoregressive next-token prediction, which induces a strong recency bias that prioritizes shared syntactic suffixes over factual denotation. Finally, Natural Language Inference (NLI) cross-encoders successfully incorporate the question but frame the comparison as strict directional entailment. This fundamental misalignment causes them to artificially fracture meaning clusters when identical facts are expressed with natural, asymmetric verbosity. Furthermore, evaluating entailment requires joint pairwise inference, imposing an $O(N^2)$ computational bottleneck. No existing choice evaluates targeted factual equivalence conditioned on the question, aligns natively with the generative distribution of the LLM, and scales efficiently.

In this work, we show that the operator, rather than the aggregator, is the true bottleneck of semantic UQ because off-the-shelf metrics fail to measure factual equivalence (Fig. 1, bottom row). Instead of designing a new aggregator, we treat the equivalence operator as a first-class object of study. To bypass the flaws of human-annotated NLP datasets, we extract a native metric space from the language model itself. When an LLM samples multiple responses, it might produce both correct and incorrect answers. This *predictive variability* provides a source of weak supervision: correct answers collapse into a single semantic equivalence class, while incorrect answers diverge.

By utilizing this structural property, we introduce a question-conditioned bi-encoder trained via a contrastive triplet objective. Trained offline without any semantic annotations, on a dataset disjoint from all evaluation settings, our encoder learns to look through lexical overlap and isolate factual divergences. During inference, it produces a mathematically symmetric similarity matrix $\boldsymbol{S}$ using only $O(N)$ forward passes, serving as a universal, drop-in replacement for any existing aggregator.

Contributions. **(1)** We formalize semantic UQ as the composition of a comparison operator and an aggregator ($U = F(\boldsymbol{S})$), isolating the off-the-shelf operator as the primary bottleneck limiting

current methods. **(2)** We introduce a framework to extract a native semantic geometry using weak supervision from the LLM's own predictive variability, training a question-conditioned equivalence bi-encoder without manual semantic labels. **(3)** We demonstrate that our operator acts as a universal upgrade: replacing the default operators in SE, CAE, KLE, and COS improves performance on 120 of 126 evaluation settings (95%) spanning 18 model–dataset combinations with both LLMs and LVLMs, raising the mean AUROC from 0.68 to 0.76 while bypassing the $O(N^2)$ scaling bottleneck of entailment baselines. **(4)** We show that the benefits extend beyond semantic aggregation: the geometric norm of our bi-encoder assigns an intrinsic importance score to individual tokens, yielding a lightweight reweighting mechanism that strictly improves single-generation token-level methods.

2 Related Work

Non-semantic Uncertainty Quantification in LLMs and LVLMs. Non-semantic UQ methods estimate uncertainty without explicitly comparing the meaning of several generations. Some approaches ask the model to report its own confidence, directly or in natural language Kadavath et al. (2022); Lin et al. (2022). Others use information from a single generation, such as token probabilities, likelihood-based scores, or internal model representations Malinin & Gales (2021); Murray & Chiang (2018); Ren et al. (2023); Azaria & Mitchell (2023); Li et al. (2023). Another line of work perturbs the input or the decoding process and measures how much the prediction changes Gao et al. (2024); Zhang et al. (2025); Li et al. (2026).

Semantic Uncertainty Quantification in LLMs and LVLMs. The closest works to ours sample several responses and measure their agreement in semantic space Kuhn et al. (2023); Farquhar et al. (2024); Lin et al. (2024). Recent methods improve this idea by using local semantic density Qiu & Miikkulainen (2024), pairwise semantic similarities Nguyen et al. (2025), semantic embeddings Grewal et al. (2024), geometric representations of uncertainty Li et al. (2026), or Gaussian-process models for LVLMs Hoche et al. (2025). Other work studies how to reduce the sampling cost of semantic UQ Park & Cho (2026), while recent analyses stress that semantic disagreement should not be confused directly with reliability Hoche et al. (2026). Our work is complementary: rather than proposing a new way to aggregate semantic disagreement, we focus on the comparison operator itself. Existing methods typically rely on NLI models, internal LLM features, or general-purpose sentence embeddings, whereas we learn a dedicated question-conditioned semantic encoder directly from LLM generations.

3 The Operator Bottleneck in Semantic UQ

Semantic UQ methods estimate uncertainty by measuring agreement across multiple model generations rather than relying on token-level probabilities. While the literature presents a diverse array of estimators, we observe that they all share a common structural decomposition.

3.1 Formalizing Semantic UQ: an Operator Followed by an Aggregator

Given a textual query $\boldsymbol{x}_{\text{text}}$ and an autoregressive language model $p_{\boldsymbol{\theta}}$, we sample N responses $\boldsymbol{y}^{(i)} \sim p_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{x}_{\text{text}})$. Every semantic estimator proceeds in two stages. First, a pairwise comparison *operator* s evaluates how far two generations express the same answer, populating a comparison matrix $\boldsymbol{S} \in \mathbb{R}^{N \times N}$:

$$\boldsymbol{S}_{ij} = s\big(\boldsymbol{y}^{(i)}, \boldsymbol{y}^{(j)} \mid \boldsymbol{x}_{\text{text}}\big). \tag{1}$$

Second, an *aggregator* F summarizes this matrix into an uncertainty score $U = F(\boldsymbol{S})$.

Prior work has focused predominantly on the design of the aggregator F. Semantic Entropy (SE) and Cluster-Assignment Entropy (CAE) Kuhn et al. (2023); Farquhar et al. (2024) reduce $\boldsymbol{S}$ to a discrete partition and compute the Shannon entropy over the resulting clusters. Kernel Language Entropy (KLE) Nikitin et al. (2024) treats $\boldsymbol{S}$ as an adjacency matrix and computes the von Neumann entropy of its graph Laplacian. COS Chen et al. (2024) computes the spectral entropy of $\boldsymbol{S}$ directly.

Conversely, the operator s is treated as a secondary design choice, inherited off-the-shelf from components built for other tasks. SE, CAE, and KLE rely on Natural Language Inference (NLI) cross-encoders trained to predict directional entailment on human-written datasets (MNLI).

COS relies on cosine similarities over either internal autoregressive hidden states (COS-int) or external general-purpose sentence encoders (COS-ext). A detailed formalization of these comparison matrices and their integration into existing baselines is provided in App. A.

3.2 Diagnostic Benchmark: How Existing Operators Fail

This design choice also creates a computational bottleneck: NLI cross-encoders compare answers pairwise, requiring $O(N^2)$ expensive forward passes to build $\boldsymbol{S}$. More importantly, these borrowed representations are not designed for the predictive variability of generative models. We therefore build a diagnostic benchmark to study the main off-the-shelf operators. Each experiment below is built from 1,000 pairs drawn from sampled and greedy generations of the same pipeline as our training data (Sec. 4); reported scores are averages over these pairs (details in App. F).

The Context Trap (COS-ext). External sentence encoders mainly capture overall semantic similarity and do not directly use the question. We compare a correct answer with a rewritten version where only the key fact is changed. Although the two answers contradict each other, external encoder gives a very high mean cosine similarity of 0.91. This shows that lexical and topical overlap can hide the factual difference that UQ should detect.

The Autoregressive Bias (COS-int). Internal LLM representations are optimized for next-token prediction rather than answer equivalence. For contradictory pairs with a shared suffix, such as *"Paris is the capital."* and *"Madrid is the capital."*, internal representations give a high mean similarity of 0.87. Hence, the shared local context can dominate the representation and hide the factual disagreement that should be detected by UQ measures.

The Entailment Trap (NLI Cross-Encoders). NLI models are trained for directional entailment, not for deciding whether two answers express the same fact. To illustrate this, we compare pairs of correct generations with very different lengths, such as *"It is Paris."* and *"The capital city of France is indeed Paris."*. Although both answers mean the same thing, NLI cross-encoders give a low mean similarity of 0.29. This shows that entailment can split semantically equivalent answers simply because they differ in wording or verbosity.

The Distribution Mismatch (OOD Syntax). Beyond architectural flaws, supervised NLI metric spaces suffer from a fundamental domain shift. Operators trained on human-annotated text (MNLI) expect well-formed, grammatically complete sentences. To quantify this, we isolate 1,000 pairs comparing the LLM's greedy generated correct sentence (e.g., *"The capital of France is Paris."*) against the dataset's raw ground-truth entity (e.g., *"Paris"*). NLI cross-encoders give a low mean similarity of only 0.27. This shows that NLI can confuse differences in form with differences in meaning, motivating an operator trained directly on LLM generations.

3.3 Desiderata for an Ideal Operator

These diagnostic failures dictate the exact requirements for a semantic UQ operator. We define an ideal operator $s^\star$ as one that satisfies three core desiderata:

(D1) Invariance to Generative Syntax: To resolve the COS-int autoregressive bias and the NLI distribution mismatch, the metric space must be robust to the specific distributional artifacts of $p_{\boldsymbol{\theta}}$. It must evaluate semantic meaning regardless of asymmetric verbosity, single-word fragments, or shared causal suffixes.

(D2) Question-Conditioned Factual Equivalence: To resolve the COS-ext context trap and the NLI entailment trap, $s^\star$ must be explicitly anchored to $\boldsymbol{x}_{\text{text}}$. It must not measure general topical similarity or directional logic, but specifically evaluate whether two generations denote the exact same factual answer to the given query.

(D3) Computational Scalability: The operator must isolate sequence representations independently to construct the similarity matrix in $O(N)$ transformer passes. Relying on cross-encoders to evaluate every pair incurs an $O(N^2)$ bottleneck that renders UQ prohibitively expensive at scale.

Satisfying these desiderata requires abandoning off-the-shelf metric spaces and external NLP datasets. In Section 4, we demonstrate how to extract an operator fulfilling all these properties by leveraging the LLM's own generative distribution.

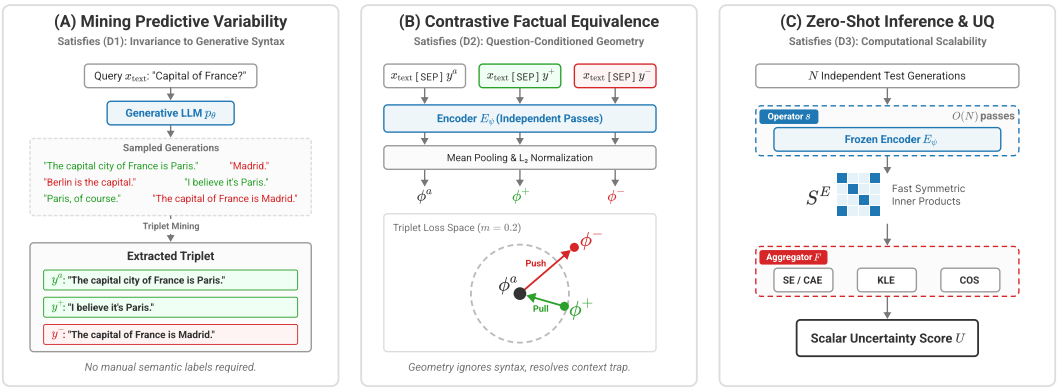

Figure 2: Learning Semantic Geometry from Predictive Variability. **(A) Mining Predictive Variability (Section 4.2):** We extract positive and negative semantic triplets by evaluating repeated LLM generations for factual correctness, naturally capturing the model's native syntax distribution. **(B) Contrastive Factual Equivalence (Sections 4.1 and 4.3):** A shared bi-encoder contextualizes the answers against the query. The contrastive margin separates hard negatives (incorrect answers to the same question) from valid paraphrases. **(C) Zero-Shot Inference & UQ (Section 5):** The frozen encoder generates symmetric $O(N)$ comparison matrices that drop seamlessly into existing aggregators, while its geometric norm enables zero-cost token reweighting.

4 Learning Semantic Geometry from Predictive Variability

To satisfy the desiderata derived in Section 3, we introduce a framework that trains a comparison operator directly on the relation it is meant to express. We extract a native semantic geometry by exploiting the predictive variability of the target language model $p_{\boldsymbol{\theta}}$ (Fig. 2). Specifically, we initialize a pre-trained bi-encoder (Section 4.1), construct a weakly supervised training dataset of semantic triplets by mining the LLM's own generations (Section 4.2), and fine-tune the encoder using a contrastive objective to isolate factual equivalence (Section 4.3).

4.1 The Architecture: Question-Conditioned Bi-Encoding

We begin by defining the trainable module and its inference architecture. To comply with **(D3)** (Computational Scalability), we avoid the $O(N^2)$ cross-attention bottleneck by adopting a bi-encoder framework. Our method is agnostic to the specific base model; we initialize a pre-trained transformer encoder $E_{\boldsymbol{\psi}}$ (e.g., DeBERTa He et al. (2020)), which we will fine-tune to independently contextualize each generation against the query.

Specifically, $E_{\boldsymbol{\psi}}$ takes the question $\boldsymbol{x}_{\text{text}}$ and a single generated response $\boldsymbol{y}^{(i)}$, joined by a separator token, and maps the sequence to a normalized dense vector $\boldsymbol{\phi}_i$:

$$\boldsymbol{\phi}_i = \frac{\bar{\boldsymbol{h}}_i}{\|\bar{\boldsymbol{h}}_i\|_2}, \qquad \bar{\boldsymbol{h}}_i = \frac{1}{T'} \sum_{t=1}^{T'} \boldsymbol{h}_t \in \mathbb{R}^d, \tag{2}$$

where $\boldsymbol{h}_t$ are the contextualized hidden states of the final layer over $E_{\boldsymbol{\psi}}\big(\boldsymbol{x}_{\text{text}}\ \texttt{[SEP]}\ \boldsymbol{y}^{(i)}\big)$ and T' is the length of the concatenated string.

The semantic operator $s_{\boldsymbol{\psi}}$ is defined as the inner product of two such embeddings:

$$s_{\boldsymbol{\psi}}\big(\boldsymbol{y}^{(i)}, \boldsymbol{y}^{(j)} \mid \boldsymbol{x}_{\text{text}}\big) = \langle \boldsymbol{\phi}_i, \boldsymbol{\phi}_j \rangle. \tag{3}$$

By keeping the question in the context window, this structural choice fulfills the first requirement of **(D2)** (Question-Conditioning), anchoring the comparison to the specific point at issue. Furthermore, because the transformer evaluates each generation only once, populating the symmetric matrix $\boldsymbol{S}$ requires exactly N forward passes followed by fast vector dot-products, reducing the operational cost from $O(N^2)$ to $O(N)$.

4.2 The Training Data: Weak Supervision via Predictive Variability

Satisfying **(D1)** (Invariance to Generative Syntax) requires E_{ψ} to be robust to the specific conversational artifacts, varying verbosity, and single-word fragments native to the generative distribution. However, obtaining manual pairwise semantic annotations for raw LLM generations would be prohibitively expensive and defeat the purpose of an automated metric.

Instead, we exploit a structural property of language models: when a query is sampled repeatedly from $p_{\boldsymbol{\theta}}$, the model produces a mix of correct and incorrect answers. We demonstrate that this predictive variability provides a natural source of weak supervision (Fig. 2A). Let $z^{\star}$ denote the semantic equivalence class of the correct factual answer, and let $g_{\boldsymbol{x}}(\boldsymbol{y})$ denote the underlying semantic class of a generated response. We define a binary correctness variable C_i:

$$C_i = \mathbb{1}\big[g_{\boldsymbol{x}}(\boldsymbol{y}^{(i)}) = z^{\star}\big]. \tag{4}$$

We rely on an **assumption**: all correct responses to the same query, regardless of their length or phrasing, belong to the same semantic equivalence class. Therefore, correctness naturally induces exact pairwise semantic relations whenever at least one generation is correct:

$$C_i = C_j = 1 \quad \Longrightarrow \quad g_{\boldsymbol{x}}(\boldsymbol{y}^{(i)}) = g_{\boldsymbol{x}}(\boldsymbol{y}^{(j)}), \tag{5}$$

$$C_i \neq C_j \quad \Longrightarrow \quad g_{\boldsymbol{x}}(\boldsymbol{y}^{(i)}) \neq g_{\boldsymbol{x}}(\boldsymbol{y}^{(j)}). \tag{6}$$

This formulation never requires an explicit supervision signal of the form *"$\boldsymbol{y}^{(i)}$ and $\boldsymbol{y}^{(j)}$ mean the same thing."* Instead, mining the outputs of $p_{\boldsymbol{\theta}}$ automatically yields positive semantic pairs $(\boldsymbol{y}_k^a, \boldsymbol{y}_k^+)$ comprising two correct generations, and negative semantic pairs $(\boldsymbol{y}_k^a, \boldsymbol{y}_k^-)$ comprising one correct and one incorrect generation. Because these pairs are drawn directly from $p_{\boldsymbol{\theta}}$, the metric space natively learns to map across the exact syntax and verbosity distributions encountered during inference.

To construct the training data, we sample multiple responses from `Llama-3.1-8B` on the NQ-OPEN dataset. An LLM-as-a-judge (`Qwen2.5-32B-Instruct`) independently labels each generation as correct or incorrect against the reference text (see App. D).

4.3 The Objective: Contrastive Factual Equivalence

While the architecture provides the necessary question-conditioning, fulfilling the remainder of **(D2)** (Factual Equivalence) requires explicitly optimizing E_{ψ} to measure targeted factual agreement rather than general topical similarity. We fine-tune the base encoder to organize the embedding space strictly according to the correctness-induced triplets (Fig. 2B).

For each triplet index k in a batch of size B, containing an anchor, a positive, and a negative text sequence, we extract their respective normalized embeddings $(\boldsymbol{\phi}_k^a, \boldsymbol{\phi}_k^+, \boldsymbol{\phi}_k^-)$ using Eq. 2, and minimize the contrastive triplet loss:

$$\mathcal{L}_{\text{triplet}} = \frac{1}{B}\sum_{k=1}^{B}\Big[\big\langle \boldsymbol{\phi}_k^a, \boldsymbol{\phi}_k^- \big\rangle - \big\langle \boldsymbol{\phi}_k^a, \boldsymbol{\phi}_k^+ \big\rangle + m\Big]_+, \tag{7}$$

where $m = 0.2$ is the margin. In high-dimensional spaces, unrelated text representations naturally concentrate near orthogonality ($\langle \boldsymbol{\phi}_k^a, \boldsymbol{\phi}_k^- \rangle \approx 0$). A margin of $m = 0.2$ provides sufficient geometric separation to isolate factual contradictions, while remaining highly permissive of valid syntactic variations and varying verbosity within the positive pairs.

This triplet structure inherently defines a distribution of *hard negatives*. Because the negative sequence $\boldsymbol{y}_k^-$ is sampled as an incorrect response to the exact same question as the anchor $\boldsymbol{y}_k^a$, it shares the overall topic and frequently mirrors the syntactic structure, differing only in the tokens conveying the factual claim. Separating these representations cannot be achieved through general topical similarity; the encoder is forced to place its geometric weight exclusively on the specific factual denotation answering the query, perfectly satisfying **(D2)**.

Crucially, this pipeline requires absolutely no semantic equivalence annotations or UQ confidence labels. While the training dataloader leverages binary correctness labels to construct triplets, the

geometric objective forces E_ψ to learn a generalized function of *equivalence*, not absolute truth. Consequently, at inference time, the frozen operator s_ψ successfully groups generations that agree, regardless of whether they agree on the correct answer. As detailed in App. B.1, it successfully clusters sets of identical factual hallucinations, which is exactly the structure required for downstream semantic UQ aggregators.

5 A Universal Drop-In for Semantic UQ

Because the semantic UQ template $U = F(\boldsymbol{S})$ decouples the metric space from the structural summary, the operator learned in Section 4 can be substituted into existing estimators without modifying their aggregators. We denote the symmetric comparison matrix produced by our operator as $\boldsymbol{S}^{\mathrm{E}}$, where $\boldsymbol{S}^{\mathrm{E}}_{ij} = \langle \boldsymbol{\phi}_i, \boldsymbol{\phi}_j \rangle$, and mark each upgraded variant with the subscript Enc.

In every case, the original aggregator F is applied exactly as published. The only change is the matrix it receives, isolating the performance gains entirely to the resolution of the operator bottleneck.

5.1 Upgrading Semantic Aggregators

Natural language inference is fundamentally directional, meaning existing semantic aggregators must enforce bidirectional entailment post-hoc to approximate equivalence. By generating symmetric inner-product matrices directly, our operator eliminates redundant directed evaluations and guarantees that $\boldsymbol{S}^{\mathrm{E}}$ is a valid positive semidefinite Gram matrix, natively satisfying the spectral assumptions required by downstream graph and eigenvalue methods.

Semantic Entropy (SE) and Cluster-Assignment Entropy (CAE) both consume a binary adjacency matrix to partition generations into meaning clusters. We replace their off-the-shelf NLI matrix by thresholding $\boldsymbol{S}^{\mathrm{E}}$: two generations are assigned to the same cluster when $\boldsymbol{S}^{\mathrm{E}}_{ij} > \tau$, with a fixed threshold $\tau = 0.5$. The resulting partition is fed to the original entropy aggregators unchanged, yielding $\mathrm{SE}_{\mathrm{Enc}}$ and $\mathrm{CAE}_{\mathrm{Enc}}$.

Kernel Language Entropy (KLE) consumes a continuous similarity matrix, traditionally built from soft NLI entailment probabilities. We pass the continuous matrix $\boldsymbol{S}^{\mathrm{E}}$ in its place and apply the identical graph Laplacian, heat/Matérn kernel diffusion, and von Neumann entropy computations, yielding $\mathrm{KLE\text{-}Heat}_{\mathrm{Enc}}$ and $\mathrm{KLE\text{-}Matern}_{\mathrm{Enc}}$.

Eigenvalue-based scores (COS) already operate on cosine similarities, previously derived from question-agnostic external encoders or prefix-biased internal hidden states. We replace those flawed representations with our question-conditioned embeddings $\boldsymbol{\phi}_i$ and apply the original spectral entropy computation to yield $\mathrm{COS}_{\mathrm{Enc}}$.

5.2 Lightweight Token-Level Reweighting

The geometric space the operator constructs also carries information at a finer granularity, from which a measure of token importance can be read without any additional training.

Writing each contextual representation as a magnitude and a direction, the pooling of Eq. 2 becomes a weighted sum of unit directions, $\boldsymbol{\phi} \propto \sum_t \|\boldsymbol{h}_t\|_2 \, \bar{\boldsymbol{h}}_t$, in which each token is weighted by its norm. Since the operator compares generations solely through the normalized vector $\boldsymbol{\phi}$, only the directions carry meaning, and the norms determine how much each token steers that direction. This allocation is a free byproduct of the triplet loss: separating an incorrect answer from a correct one requires placing norm on the tokens that carry the factual claim and withholding it from those that vary freely across paraphrases. We therefore read $\|\boldsymbol{h}_t\|_2$ as a measure of semantic importance.

This upgrades the uniform averaging of token-level entropy (TE). Normalizing the norms of the generation's tokens into weights over those positions, discarding the query and special tokens, gives

$$w_t = \frac{\|\boldsymbol{h}_t\|_2}{\sum_{t'=1}^{T} \|\boldsymbol{h}_{t'}\|_2}, \qquad U_{\mathrm{TE}_{\mathrm{Enc}}}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}) = \sum_{t=1}^{T} w_t \, H_t(\boldsymbol{x}, \boldsymbol{y}_{<t}, \boldsymbol{\theta}), \qquad \text{with } H_t \text{ as in Eq. (A.2)} \tag{8}$$

Like TE, this requires a single generation, but down-weights structurally necessary tokens that carry no answer content. Because the encoder and the generating model use different tokenizers, we apply a sequence alignment to map the norms onto the generative vocabulary before computing the sum.

6 EXPERIMENTS

6.1 SETUP

Models and Datasets. We evaluate on six models spanning both modalities. For text-only question answering we use `Qwen2.5-7B` Bai et al. (2025), `Phi-3.5-mini` Abdin et al. (2024) and `Mistral-7B` Jiang et al. (2023); for visual question answering we use `idefics2-8B` Laurençon et al. (2023), `Qwen2.5-VL-7B` Bai et al. (2025) and `llava-1.5-7B` Liu et al. (2024b). None of these models is involved in the construction of the training set of Sec. 4, which is built from generations of `Llama-3.1-8B` alone. We evaluate on six datasets, again covering both settings. The visual question answering benchmarks are **ADVQA** Li et al. (2021), **OKVQA** Marino et al. (2019) and **VizWiz** Gurari et al. (2018); the textual ones are **HotpotQA** Yang et al. (2018), **TriviaQA** Joshi et al. (2017) and **WebQuestions** Berant et al. (2013). Each model is evaluated on the three datasets of its modality, giving the 18 model–dataset combinations reported throughout. The exact splits can be found in App. E. **NQ-OPEN** Lee et al. (2019), used to train the operator, appears in none of them.

Generation. For every question we sample $N = 20$ answers with temperature 1.0, top-$p = 0.9$, top-$k = 50$, and additionally record the greedy generation together with its token-level distributions, which the token-level estimators require. Additional details can be found in App. E.

Correctness labels. Following prior work Farquhar et al. (2024), correctness is assessed on the greedy generation: an LLM-as-a-judge, here `Llama-3.1-70B-Instruct`, compares it to the reference answer and returns a binary label, which serves as the target against which every uncertainty score is evaluated. An alternative convention assigns the label by majority vote over the sampled generations; this is an equally reasonable choice, and App. G.3 reports the same experiments under it, with unchanged conclusions.

Metrics. We report AUROC and ECE. AUROC measures how well an uncertainty score ranks incorrect generations above correct ones, with 0.5 corresponding to random ordering and higher values better; it is invariant to any monotone rescaling of the score, and so evaluates discrimination alone. ECE measures how far predicted confidences are from empirical correctness frequencies, averaged over confidence bins, and is lower when the two agree. Since most of the estimators considered here output an unbounded score rather than a probability, ECE is computed after mapping scores to confidences with a transformation fitted on a held-out split, identically for every method.

6.2 MAIN RESULTS

Tab. 1 reports every estimator in its published form and with our operator substituted, averaged over the 18 model–dataset combinations and broken down by dataset and by model.

Replacing the comparison operator improves every semantic estimator. Substituting the learned operator raises the mean AUROC for every baseline estimator across the 18 model–dataset combinations: CAE improves from 0.612 to 0.724, SE from 0.610 to 0.726, KLE-Heat from 0.677 to 0.756, KLE-Matern from 0.682 to 0.756, and COS from 0.679 (for the stronger variant) to 0.759. Overall 120 of 126 comparisons (95%), obtained by pairing each of the 7 estimators with its operator-substituted counterpart across all 18 model–dataset combinations spanning LLMs and LVLMs, show an improvement. Per-cell results with bootstrapped CIs are detailed in Tabs. G.1 and G.4.

The operator is substantially cheaper. Building $\boldsymbol{S}$ with the conventional NLI operator costs 1.1–1.2,s per question; with ours, it takes 0.03–0.04,s, representing roughly a $30\times$ reduction, since the comparison no longer requires a forward pass per pair. COS-int is the only operator cheaper still, at the cost of white-box access and yielding the weakest performance in the table.

The question matters. Tab. 1 reports $\mathrm{COS}_{\mathrm{Enc}}$ with the question as part of the encoder input, as described in Sec. 4. Retraining the same operator on answer strings alone, removing $\boldsymbol{x}_{\mathrm{text}}$ from the input of E_ψ, drops the mean AUROC from 0.759 to 0.742. The gap is smaller than the gain over the conventional operators, but it holds in every one of the 18 model–dataset combinations (Tab. G.2):

Table 1: **Main results.** AUROC (↑) and ECE (↓) for each estimator in its published form (*base*) and with our operator substituted (*shaded*), averaged over the 18 model–dataset combinations and reported by dataset (top) and by model (bottom). Δ is the change in mean AUROC over all columns.

Method	ADVQA AUROC	ADVQA ECE	OKVQA AUROC	OKVQA ECE	VizWiz AUROC	VizWiz ECE	HotpotQA AUROC	HotpotQA ECE	TriviaQA AUROC	TriviaQA ECE	WebQuestions AUROC	WebQuestions ECE	Δ	post-hoc compute
CAE	.637	.116	.643	.229	.647	.199	.673	.235	.583	.212	.490	.176		1.11 s
CAE_{Enc}	**.663**	**.081**	**.733**	**.194**	**.674**	**.159**	**.722**	**.215**	**.840**	**.135**	**.716**	**.055**	+.113	**0.03 s**
SE	.637	.114	.643	.228	.644	.198	.674	.235	.577	.212	.488	.175		1.18 s
SE_{Enc}	**.666**	**.080**	**.733**	**.193**	**.675**	**.159**	**.725**	**.214**	**.840**	**.133**	**.719**	**.058**	+.116	**0.03 s**
KLE-Heat	.675	.080	.678	**.187**	.690	**.161**	.719	**.208**	.747	**.127**	.555	.112		1.21 s
$\text{KLE-Heat}_{\text{Enc}}$	**.683**	**.077**	**.763**	.203	**.732**	.169	**.763**	.220	**.855**	.140	**.739**	**.045**	+.079	**0.04 s**
KLE-Matern	**.688**	**.070**	.697	**.202**	.712	.178	.715	.226	.729	.149	.548	.138		1.23 s
$\text{KLE-Matern}_{\text{Enc}}$	.684	.074	**.763**	.205	**.733**	**.173**	**.763**	**.223**	**.855**	**.141**	**.739**	**.044**	+.075	**0.04 s**
COS-int	.577	.117	.621	**.203**	.641	**.171**	.720	.228	.658	.139	.602	.144		**0.01 s**
COS-ext	.651	.094	.673	.205	.705	.179	.707	.230	.785	**.129**	.554	.135		0.03 s
COS_{Enc}	**.694**	**.072**	**.758**	.208	**.741**	.178	**.765**	**.225**	**.852**	.141	**.741**	**.043**	+.079	0.03 s
TE	.608	.099	.594	.205	.645	.179	**.738**	.221	.669	.142	.589	.121		**0.01 s**
TE_{Enc}	**.636**	**.079**	**.641**	**.199**	**.668**	**.171**	.699	**.216**	**.769**	**.119**	**.655**	**.074**	+.038	0.02 s

Method	idefics2-8B AUROC	idefics2-8B ECE	Qwen2.5-VL-7B AUROC	Qwen2.5-VL-7B ECE	llava-1.5-7B AUROC	llava-1.5-7B ECE	Qwen2.5-7B AUROC	Qwen2.5-7B ECE	Phi-3.5-mini AUROC	Phi-3.5-mini ECE	Mistral-7B AUROC	Mistral-7B ECE	Δ	post-hoc compute
CAE	.654	.192	.626	.196	.647	.156	.639	.179	.517	.232	.590	.213		1.11 s
CAE_{Enc}	**.678**	**.159**	**.717**	**.144**	**.675**	**.131**	**.769**	**.132**	**.764**	**.130**	**.744**	**.142**	+.112	**0.03 s**
SE	.654	.189	.624	.195	.645	.155	.638	.178	.515	.231	.586	.213		1.18 s
SE_{Enc}	**.679**	**.158**	**.717**	**.144**	**.678**	**.130**	**.770**	**.131**	**.767**	**.130**	**.747**	**.143**	+.116	**0.03 s**
KLE-Heat	.687	.162	.672	**.145**	.684	**.121**	.698	.138	.629	.154	.693	.155		1.21 s
$\text{KLE-Heat}_{\text{Enc}}$	**.723**	**.161**	**.740**	.155	**.715**	.134	**.792**	**.127**	**.792**	**.132**	**.774**	**.145**	+.079	**0.04 s**
KLE-Matern	.707	.165	.681	.156	.709	**.130**	.693	.148	.611	.183	.689	.182		1.23 s
$\text{KLE-Matern}_{\text{Enc}}$	**.724**	**.163**	**.740**	**.155**	**.716**	.133	**.791**	**.128**	**.792**	**.134**	**.774**	**.147**	+.075	**0.04 s**
COS-int	.592	.168	.625	.185	.623	.139	.699	.168	.605	.204	.685	**.138**		**0.01 s**
COS-ext	.669	.174	.693	.162	.667	.142	.688	.157	.704	.137	.653	.199		0.03 s
COS_{Enc}	**.730**	**.165**	**.740**	**.159**	**.723**	**.133**	**.790**	**.124**	**.793**	**.135**	**.775**	.150	+.080	0.03 s
TE	.615	.175	.613	.172	.619	.136	.716	.142	.659	.146	.621	.195		**0.01 s**
TE_{Enc}	**.650**	**.164**	**.644**	**.154**	**.651**	**.131**	**.741**	**.123**	**.714**	**.125**	**.668**	**.161**	+.038	0.02 s

Note. Compute is the post-hoc time to build $\boldsymbol{S}$ for one question; excluding the generation shared by all methods we compare against.

conditioning is a real, if secondary, part of the improvement, and consistent with **(D2)**, which no operator defined on answer pairs alone can satisfy.

The token-level estimator also improves. The pattern holds even for the cheapest estimator: reweighting TE with our operator raises the mean AUROC from 0.640 to 0.678, using a single generation and one extra encoder pass. This shows the operator is useful beyond semantic UQ.

Additional experiments on the necessity of hard negatives, sensitivity to the clustering threshold, other hyperparameters, and further ablations are provided in App. B.

7 Conclusion

We formalized semantic uncertainty quantification as the composition of an operator that decides whether two generations answer a question the same way, and an aggregator that turns the resulting comparisons into a scalar. We observed that every method in this family proposes the latter while inheriting the former from a model built for another task. We introduced a framework for training such an operator on the relation it is meant to express, utilizing a contrastive objective and synthetic data disjoint from every evaluation setting. Substituted into SE, CAE, KLE, and COS, it improves them across 18 model–dataset combinations, requiring only N network evaluations instead of $O(N^2)$. The uniformity of this improvement substantiates our core hypothesis: a single change to the shared component improves every baseline estimator built on it.

The operator is trained on a small synthetic set with a single configuration, and might benefit from more data and a finer-grained objective. We left it there deliberately: the claim we set out to test is that the operator is the limiting component, not that ours is the best one, and our aim was not to introduce a new supervised uncertainty method. That a recipe this minimal already improves every estimator is the point, and it leaves considerable room above the estimates reported here.

AI USE STATEMENT

We used generative AI tools solely to assist with the writing of this manuscript: refining sentence-level phrasing and style, and formatting tables. Separately, an LLM is used within the method itself, as a generator and a judge for constructing the training data described in Sec. 4; this is a documented and integral part of the proposed pipeline rather than an authoring aid, and is evaluated as such throughout the paper. All research ideas, the formalism, experimental design, code, results, and their interpretation are the authors' own and were not suggested, generated, or verified by AI tools. We have reviewed and take full responsibility for every sentence and table in this paper, including any produced with AI assistance.

REPRODUCIBILITY STATEMENT

All details needed to reproduce our results are given in the paper. The construction of the training set, including the source dataset and split, the generating model, sampling parameters and triplet mining, is described in App. C, together with the training hyperparameters of the operator. The correctness-labelling protocol and the judges used for training and evaluation are specified in App. D. The evaluated models, dataset splits, prompts and decoding parameters are listed in App. E, which also gives the operators and hyperparameters used for every baseline. The construction of the diagnostic benchmark of Sec. 3.2 is detailed in App. F, and per-cell results with bootstrapped confidence intervals are reported in App. G. All models, encoders and datasets used are publicly available. Code, trained operator weights and the generated training data will be released upon acceptance.

REFERENCES

Samir Abdaljalil, Hasan Kurban, Parichit Sharma, Erchin Serpedin, and Rachad Atat. Sindex: Semantic inconsistency index for hallucination detection in llms. *arXiv preprint arXiv:2503.05980*, 2025.

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, et al. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024.

Lukas Aichberger, Kajetan Schweighofer, and Sepp Hochreiter. Rethinking uncertainty estimation in natural language generation. *arXiv preprint arXiv:2412.15176*, 2024.

Amos Azaria and Tom Mitchell. The internal state of an LLM knows when it's lying. In *EMNLP*, 2023.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1533–1544, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D13-1160.

Jakub Binkowski, Denis Janiak, Albert Sawczyn, Bogdan Gabrys, and Tomasz Jan Kajdanowicz. Hallucination detection in LLMs using spectral features of attention maps. In *EMNLP*, 2025.

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. In *ICLR*, 2024.

Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. Detecting hallucinations in large language models using semantic entropy. *Nature*, 2024.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016.

Xiang Gao, Jiaxin Zhang, Lalla Mouatadid, and Kamalika Das. SPUQ: Perturbation-based uncertainty quantification for large language models. In *EACL*, 2024.

Yashvir S Grewal, Edwin V Bonilla, and Thang D Bui. Improving uncertainty quantification in large language models via semantic embeddings. *arXiv preprint arXiv:2410.22685*, 2024.

Danna Gurari, Qing Li, Abigale J Stangl, Anhong Guo, Chi Lin, Kristen Grauman, Jiebo Luo, and Jeffrey P Bigham. Vizwiz grand challenge: Answering visual questions from blind people. In *CVPR*, 2018.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*, 2020.

Jose Miguel Hernandez-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *ICML*, 2015.

Joseph Hoche, Andrei Bursuc, David Brellmann, Gilles Louppe, Pavel Izmailov, Angela Yao, and Gianni Franchi. Improving semantic uncertainty quantification in lvlms with semantic gaussian processes. *arXiv preprint arXiv:2512.14177*, 2025.

Joseph Hoche, Maxime Corlay, David Brellmann, Andrei Bursuc, Pavel Izmailov, Angela Yao, and Gianni Franchi. Position: Semantic Uncertainty Measures Disagreement, Not Reliability. working paper or preprint, May 2026. URL `https://hal.science/hal-05633506`.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *CS*, 2023.

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

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*, 2017.

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.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *ICLR*, 2023.

Hugo Laurençon, Lucile Saulnier, Léo Tronchon, Stas Bekman, Amanpreet Singh, Anton Lozhkov, Thomas Wang, Siddharth Karamcheti, Alexander Rush, Douwe Kiela, et al. Obelics: An open web-scale filtered dataset of interleaved image-text documents. *Advances in Neural Information Processing Systems*, 36:71683–71702, 2023.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 6086–6096, 2019.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. In *NeurIPS*, 2023.

Linjie Li, Jie Lei, Zhe Gan, and Jingjing Liu. Adversarial vqa: A new benchmark for evaluating the robustness of vqa models. In *ICCV*, 2021.

Xiaomin Li, Zhou Yu, Ziji Zhang, Yingying Zhuang, Swair Shah, Narayanan Sadagopan, and Anurag Beniwal. Semantic volume: Quantifying and detecting both external and internal uncertainty in llms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 31751–31759, 2026.

Stephanie Lin, Jacob Hilton, and Owain Evans. Teaching models to express their uncertainty in words. *TMLR*, 2022.

Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. Generating with confidence: Uncertainty quantification for black-box large language models. *TMLR*, 2024.

Hanchao Liu, Wenyuan Xue, Yifei Chen, Dapeng Chen, Xiutian Zhao, Ke Wang, Liping Hou, Rongjun Li, and Wei Peng. A survey on hallucination in large vision-language models. *arXiv preprint arXiv:2402.00253*, 2024a.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *CVPR*, 2024b.

Andrey Malinin and Mark Gales. Uncertainty estimation in autoregressive structured prediction. In *ICLR*, 2021.

Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. Ok-vqa: A visual question answering benchmark requiring external knowledge. In *CVPR*, 2019.

Kenton Murray and David Chiang. Correcting length bias in neural machine translation. In *MT*, 2018.

Dang Nguyen, Ali Payani, and Baharan Mirzasoleiman. Beyond semantic entropy: Boosting LLM uncertainty quantification with pairwise semantic similarity. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 4530–4540, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.234. URL https://aclanthology.org/2025.findings-acl.234/.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. In *Findings of the association for computational linguistics: ACL 2022*, pp. 1864–1874, 2022.

Alexander Nikitin, Jannik Kossen, Yarin Gal, and Pekka Marttinen. Kernel language entropy: Fine-grained uncertainty quantification for llms from semantic similarities. In *NeurIPS*, 2024.

Ji Won Park and Kyunghyun Cho. Efficient semantic uncertainty quantification in language models via diversity-steered sampling. *Advances in Neural Information Processing Systems*, 38:32154–32183, 2026.

Xin Qiu and Risto Miikkulainen. Semantic density: Uncertainty quantification for large language models through confidence measurement in semantic space. In *NeurIPS*, 2024.

Jie Ren, Jiaming Luo, Yao Zhao, Kundan Krishna, Mohammad Saleh, Balaji Lakshminarayanan, and Peter J Liu. Out-of-distribution detection and selective generation for conditional language models. In *ICLR*, 2023.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2369–2380, 2018.

Tunyu Zhang, Haizhou Shi, Yibin Wang, Hengyi Wang, Xiaoxiao He, Zhuowei Li, Haoxian Chen, Ligong Han, Kai Xu, Huan Zhang, et al. Token-level uncertainty estimation for large language model reasoning. *arXiv preprint arXiv:2505.11737*, 2025.

A Operators and aggregators

A.1 Preliminaries

Given an input $\boldsymbol{x}$, consisting of a textual query $\boldsymbol{x}_{\text{text}}$ possibly accompanied by an image $\boldsymbol{x}_{\text{im}}$, a model parameterized by $\boldsymbol{\theta}$ autoregressively generates an output response sequence of tokens $\boldsymbol{y} = \{\boldsymbol{y}_t\}_{t=1}^{T}$, where $\boldsymbol{y}_t$ denotes the $t^{\text{-th}}$ output token and T is the length of the output sequence. The probability of the generated sequence $\boldsymbol{y}$ is the joint probability of tokens, defined as the product of conditional token probabilities:

$$p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}) = \prod_{t=1}^{T} p(\boldsymbol{y}_t \mid \boldsymbol{x}, \boldsymbol{y}_{<t}, \boldsymbol{\theta}), \quad \text{where } \boldsymbol{y}_{<t} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{t-1}\}. \tag{A.1}$$

Token Entropy. TE is the simplest uncertainty estimate, it operates directly on the predictive distributions produced during decoding, and requires a single generation. At step t, the model defines a distribution over the vocabulary $\mathcal{V}$, whose Shannon entropy

$$H_t(\boldsymbol{x}, \boldsymbol{y}_{<t}, \boldsymbol{\theta}) = -\sum_{v \in \mathcal{V}} p(v \mid \boldsymbol{x}, \boldsymbol{y}_{<t}, \boldsymbol{\theta}) \log p(v \mid \boldsymbol{x}, \boldsymbol{y}_{<t}, \boldsymbol{\theta}) \tag{A.2}$$

measures how dispersed the model is over the possible continuations at that position. The token-level uncertainty of a generation is obtained by averaging these entropies over the generated sequence:

$$U_{\text{TE}}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{T} \sum_{t=1}^{T} H_t(\boldsymbol{x}, \boldsymbol{y}_{<t}, \boldsymbol{\theta}). \tag{A.3}$$

This estimate is cheap, requiring neither sampling nor comparison between generations, but it is computed entirely in token space. Every position contributes equally to the average, although the positions differ widely in how much they bear on the answer: a model may hesitate between several equally acceptable phrasings while being certain of the answer itself, and such hesitation raises U_{TE} exactly as hesitation over the answer would. Semantic methods compare generations by meaning rather than by token sequence.

This section reviews these methods and isolates the component they share. We first introduce the comparison matrix every such method is built on and the forms it takes in the literature (Sec. A.2), then present the estimators themselves (Sec. A.3), and formalize their decomposition (Sec. A.4).

A.2 The comparison matrix

Each semantic estimator starts from a pairwise comparison of the sampled generations $\mathcal{Y} := \{\boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(N)}\}$, collected in a matrix

$$\boldsymbol{S} \in \mathbb{R}^{N \times N}, \qquad \boldsymbol{S}_{ij} = s\big(\boldsymbol{y}^{(i)}, \boldsymbol{y}^{(j)} \mid \boldsymbol{x}_{\text{text}}\big), \tag{A.4}$$

where the *operator* s scores how far two generations express the same answer to $\boldsymbol{x}_{\text{text}}$. Two instantiations of s account for every method considered here.

Cosine similarity ($\boldsymbol{S}^{\cos}$). Each generation is mapped to a normalized embedding $\boldsymbol{\phi}_i \in \mathbb{R}^d$ and compared by inner product, $\boldsymbol{S}^{\cos}_{ij} = \langle \boldsymbol{\phi}_i, \boldsymbol{\phi}_j \rangle$. The embeddings are taken either from the hidden states of the generating model (COS-int) Chen et al. (2024); Binkowski et al. (2025) or from an external sentence encoder (COS-ext) Grewal et al. (2024); Abdaljalil et al. (2025).

Bidirectional entailment ($\boldsymbol{S}^{\text{nli}}$). Each generation is concatenated with $\boldsymbol{x}_{\text{text}}$ and the pair is passed to a pretrained NLI model, which predicts one of entailment, neutral or contradiction. The operator records whether entailment is predicted in both directions,

$$\boldsymbol{S}^{\text{nli}}_{ij} = \mathbf{1}\big[\boldsymbol{x}_{\text{text}} \wedge \boldsymbol{y}^{(i)} \models \boldsymbol{y}^{(j)} \text{ and } \boldsymbol{x}_{\text{text}} \wedge \boldsymbol{y}^{(j)} \models \boldsymbol{y}^{(i)}\big] \in \{0, 1\}. \tag{A.5}$$

Some methods instead retain the entailment probabilities underlying these predictions, giving a graded matrix $\boldsymbol{S}^{\text{nli}}_{\text{soft}} \in [0, 1]^{N \times N}$ that distinguishes pairs the hard decision treats alike.

In both cases the matrix is inherited from a component developed for another purpose. The estimators below draw on one or the other and differ in what they compute from it.

A.3 SEMANTIC ESTIMATORS

Semantic clustering. SE and CAE both reduce the comparison to a partition of $\mathcal{Y}$ into semantically consistent clusters $\mathcal{C} := \{\mathcal{C}_k\}_{k=1}^{K}$, such that sequences within a cluster share the same meaning. They consume $\boldsymbol{S}^{\text{nli}}$: clusters are formed by comparing each new generation against the existing ones and creating a new cluster when no bidirectional entailment is found Kuhn et al. (2023); Farquhar et al. (2024).

Semantic Entropy. SE weights each cluster by the normalized sequence probabilities of its members,

$$p(\mathcal{C}_k \mid \boldsymbol{x}, \boldsymbol{\theta}) = \sum_{\boldsymbol{y} \in \mathcal{C}_k} \tilde{p}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}), \qquad \tilde{p}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}) = \frac{p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta})}{\sum_{i=1}^{N} p(\boldsymbol{y}^{(i)} \mid \boldsymbol{x}, \boldsymbol{\theta})}, \tag{A.6}$$

and returns the entropy of the resulting cluster distribution:

$$H_{\text{SE}}(\mathcal{Y} \mid \boldsymbol{x}, \boldsymbol{\theta}) = -\sum_{k=1}^{K} p(\mathcal{C}_k \mid \boldsymbol{x}, \boldsymbol{\theta}) \log p(\mathcal{C}_k \mid \boldsymbol{x}, \boldsymbol{\theta}). \tag{A.7}$$

A low value indicates that the responses concentrate on a single meaning, a high value that probability mass is dispersed across distinct meanings.

Cluster-Assignment Entropy. CAE uses the same partition but replaces the probability mass of each cluster by its empirical frequency, removing the need for access to sequence probabilities Farquhar et al. (2024):

$$H_{\text{CAE}}(\mathcal{Y} \mid \boldsymbol{x}) = -\sum_{k=1}^{K} \frac{|\mathcal{C}_k|}{N} \log \frac{|\mathcal{C}_k|}{N}. \tag{A.8}$$

Kernel Language Entropy. Rather than reducing the comparisons to a partition, KLE turns them into a weighted semantic graph over the generations, taking the graded matrix as edge weights, $\boldsymbol{W} = \boldsymbol{S}^{\text{nli}}_{\text{soft}}$ Nikitin et al. (2024). From its Laplacian $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$, where $\boldsymbol{D}$ is the diagonal degree matrix $\boldsymbol{D}_{ii} = \sum_j \boldsymbol{W}_{ij}$, a graph kernel is obtained as the solution of a diffusion equation, $\boldsymbol{K}_t = e^{-t\boldsymbol{L}}$ for the heat kernel (KLE-Heat) or $\boldsymbol{K}_{\nu\kappa} = (2\nu/\kappa^2\boldsymbol{I} + \boldsymbol{L})^{-\nu}$ for the Matérn kernel (KLE-Matern); after normalization to unit trace, the uncertainty is its von Neumann entropy,

$$H_{\text{KLE}}(\mathcal{Y} \mid \boldsymbol{x}) = -\operatorname{Tr}\big(\boldsymbol{K} \log \boldsymbol{K}\big). \tag{A.9}$$

Because the graph carries graded rather than binary weights, KLE distinguishes generations that are related without being equivalent, which hard clustering cannot; it is in fact a strict generalization of SE, which it recovers for a block-diagonal kernel.

Eigenvalue-based scores. COS-int and COS-ext consume $\boldsymbol{S}^{\cos}$ and summarize its spectrum directly, differing only in the embeddings it is built from. Being a Gram matrix of normalized vectors, $\boldsymbol{S}^{\cos}$ is positive semidefinite with trace N, so its eigenvalues $\lambda_1 \geq \cdots \geq \lambda_N$ normalized by their sum form a distribution over the orthogonal directions of the embedding space. The uncertainty is the von Neumann entropy of that distribution Chen et al. (2024):

$$U_{\cos}(\mathcal{Y} \mid \boldsymbol{x}) = -\sum_{i=1}^{N} \tilde{\lambda}_i \log \tilde{\lambda}_i, \qquad \tilde{\lambda}_i = \frac{\lambda_i}{\sum_{j=1}^{N} \lambda_j} = \frac{\lambda_i}{N}. \tag{A.10}$$

A spectrum dominated by one eigenvalue indicates that the generations concentrate on a single direction of meaning, a flat spectrum that they are dispersed across several.

A.4 DECOMPOSING EXISTING SEMANTIC UQ METHODS INTO AN OPERATOR AND AN AGGREGATOR

A common view. Although semantic UQ methods look different, most of them follow the same two-step structure. Given a set of sampled generations $\mathcal{Y} = \{\boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(N)}\}$, they first compare the generations pairwise through an *operator* s, producing

$$\boldsymbol{S}_{ij} = s\left(\boldsymbol{y}^{(i)}, \boldsymbol{y}^{(j)} \mid \boldsymbol{x}_{\text{text}}\right). \tag{A.11}$$

They then apply an *aggregator* F to these pairwise relations,

$$\boxed{U = F(\boldsymbol{S}).} \tag{A.12}$$

For methods that also use sequence probabilities, we write more generally

$$U = F\left(\boldsymbol{S}, \left\{p(\boldsymbol{y}^{(i)} \mid \boldsymbol{x}, \boldsymbol{\theta})\right\}_{i=1}^{N}\right). \tag{A.13}$$

The distinction is simple: the operator s decides *how two answers are related*, while the aggregator F decides *how all these relations are combined into one uncertainty score*. Below, we make both s and F explicit for each estimator.

A.4.1 Semantic Entropy

SE: NLI operator followed by probability-weighted cluster entropy. Semantic Entropy first compares the generations using a binary NLI relation. Its operator is

$$s_{\mathrm{SE}} = s_{\mathrm{NLI}}, \tag{A.14}$$

which produces the binary matrix $\boldsymbol{S}^{\mathrm{nli}}$.

Let

$$\mathcal{C}(\boldsymbol{S}) = \Pi(\boldsymbol{S}) = \{\mathcal{C}_1(\boldsymbol{S}), \ldots, \mathcal{C}_{K(\boldsymbol{S})}(\boldsymbol{S})\} \tag{A.15}$$

denote the semantic partition obtained by applying the clustering rule Π to the matrix $\boldsymbol{S}$. For each cluster, define

$$q_k(\boldsymbol{S}) = \sum_{\boldsymbol{y}^{(i)} \in \mathcal{C}_k(\boldsymbol{S})} \tilde{p}_i, \qquad \tilde{p}_i = \frac{p(\boldsymbol{y}^{(i)} \mid \boldsymbol{x}, \boldsymbol{\theta})}{\sum_{j=1}^{N} p(\boldsymbol{y}^{(j)} \mid \boldsymbol{x}, \boldsymbol{\theta})}. \tag{A.16}$$

The SE aggregator is therefore the function

$$\boxed{F_{\mathrm{SE}}\left(\boldsymbol{S}, \{\tilde{p}_i\}_{i=1}^{N}\right) := -\sum_{k=1}^{K(\boldsymbol{S})} q_k(\boldsymbol{S}) \log q_k(\boldsymbol{S}).} \tag{A.17}$$

Thus,

$$U_{\mathrm{SE}} = F_{\mathrm{SE}}\left(\boldsymbol{S}^{\mathrm{nli}}, \{\tilde{p}_i\}_{i=1}^{N}\right). \tag{A.18}$$

The operator determines which responses belong to the same semantic cluster, while F_{SE} measures how the sequence probability mass is distributed across these clusters.

A.4.2 Cluster-Assignment Entropy

CAE: the same semantic operator with a different aggregator. CAE uses the same binary NLI matrix and the same clustering rule,

$$\mathcal{C}(\boldsymbol{S}) = \Pi(\boldsymbol{S}). \tag{A.19}$$

The difference from SE comes after the clusters have been obtained. CAE assigns to each cluster its empirical frequency,

$$\widehat{q}_k(\boldsymbol{S}) = \frac{|\mathcal{C}_k(\boldsymbol{S})|}{N}. \tag{A.20}$$

Its aggregator is therefore

$$\boxed{F_{\mathrm{CAE}}(\boldsymbol{S}) := -\sum_{k=1}^{K(\boldsymbol{S})} \widehat{q}_k(\boldsymbol{S}) \log \widehat{q}_k(\boldsymbol{S}).} \tag{A.21}$$

Hence,

$$U_{\mathrm{CAE}} = F_{\mathrm{CAE}}\left(\boldsymbol{S}^{\mathrm{nli}}\right). \tag{A.22}$$

SE and CAE therefore use the same semantic operator and clustering rule. They differ only in the final aggregation: SE uses sequence probability mass, while CAE uses empirical cluster frequency.

A.4.3 KERNEL LANGUAGE ENTROPY

KLE: continuous NLI operator followed by graph aggregation. KLE does not first reduce semantic relations to binary clusters. Instead, its operator produces a continuous semantic matrix

$$\boldsymbol{S} = \boldsymbol{S}^{\text{nli}}_{\text{soft}}, \tag{A.23}$$

which is interpreted as the weighted adjacency matrix

$$\boldsymbol{W}(\boldsymbol{S}) = \boldsymbol{S}. \tag{A.24}$$

From this matrix, define

$$\boldsymbol{D}(\boldsymbol{S})_{ii} = \sum_j \boldsymbol{W}(\boldsymbol{S})_{ij}, \qquad \boldsymbol{L}(\boldsymbol{S}) = \boldsymbol{D}(\boldsymbol{S}) - \boldsymbol{W}(\boldsymbol{S}). \tag{A.25}$$

For KLE-Heat, the kernel is

$$\boldsymbol{K}_{\text{Heat}}(\boldsymbol{S}) = \exp\left(-t\boldsymbol{L}(\boldsymbol{S})\right), \tag{A.26}$$

while for KLE-Matern,

$$\boldsymbol{K}_{\text{Matern}}(\boldsymbol{S}) = \left(\frac{2\nu}{\kappa^2}\boldsymbol{I} + \boldsymbol{L}(\boldsymbol{S})\right)^{-\nu}. \tag{A.27}$$

For either kernel, define its trace-normalized form

$$\bar{\boldsymbol{K}}(\boldsymbol{S}) = \frac{\boldsymbol{K}(\boldsymbol{S})}{\text{Tr}(\boldsymbol{K}(\boldsymbol{S}))}. \tag{A.28}$$

The KLE aggregator is then

$$\boxed{F_{\text{KLE}}(\boldsymbol{S}; \boldsymbol{K}) := -\text{Tr}\left(\bar{\boldsymbol{K}}(\boldsymbol{S}) \log \bar{\boldsymbol{K}}(\boldsymbol{S})\right).} \tag{A.29}$$

More explicitly,

$$F_{\text{Heat}}(\boldsymbol{S}) := -\text{Tr}\left(\bar{\boldsymbol{K}}_{\text{Heat}}(\boldsymbol{S}) \log \bar{\boldsymbol{K}}_{\text{Heat}}(\boldsymbol{S})\right), \tag{A.30}$$

$$F_{\text{Matern}}(\boldsymbol{S}) := -\text{Tr}\left(\bar{\boldsymbol{K}}_{\text{Matern}}(\boldsymbol{S}) \log \bar{\boldsymbol{K}}_{\text{Matern}}(\boldsymbol{S})\right). \tag{A.31}$$

Therefore,

$$U_{\text{KLE-Heat}} = F_{\text{Heat}}\left(\boldsymbol{S}^{\text{nli}}_{\text{soft}}\right), \tag{A.32}$$

$$U_{\text{KLE-Matern}} = F_{\text{Matern}}\left(\boldsymbol{S}^{\text{nli}}_{\text{soft}}\right). \tag{A.33}$$

The two KLE variants therefore use the same semantic operator. They differ only in the graph kernel used inside the aggregator.

A.4.4 COS

COS: cosine operator followed by spectral aggregation. COS builds its comparison matrix from cosine similarities between answer representations,

$$\boldsymbol{S}^{\cos}_{ij} = \langle \boldsymbol{\phi}_i, \boldsymbol{\phi}_j \rangle. \tag{A.34}$$

COS-int and COS-ext differ only in the source of $\boldsymbol{\phi}_i$: COS-int uses representations from the generating model, whereas COS-ext uses an external sentence encoder.

For any positive semidefinite similarity matrix $\boldsymbol{S}$, let

$$\lambda_1(\boldsymbol{S}), \ldots, \lambda_N(\boldsymbol{S}) \tag{A.35}$$

be its eigenvalues and define

$$\widetilde{\lambda}_i(\boldsymbol{S}) = \frac{\lambda_i(\boldsymbol{S})}{\sum_j \lambda_j(\boldsymbol{S})} = \frac{\lambda_i(\boldsymbol{S})}{\text{Tr}(\boldsymbol{S})}. \tag{A.36}$$

The COS aggregator is

$$\boxed{F_{\cos}(\boldsymbol{S}) := -\sum_{i=1}^{N} \widetilde{\lambda}_i(\boldsymbol{S}) \log \widetilde{\lambda}_i(\boldsymbol{S}).} \tag{A.37}$$

Hence,

$$U_{\cos} = F_{\cos}\left(\boldsymbol{S}^{\cos}\right). \tag{A.38}$$

Since the embeddings are normalized, $\text{Tr}(\boldsymbol{S}^{\cos}) = N$. The operator therefore defines the geometry of the generations, while $F_{\cos}$ measures how this geometry is spread across its spectral directions.

B Additional Experiments

B.1 Clustering of Incorrect Answers

As described in Section 4.3, we use answer correctness as a proxy for semantic equivalence when building training triplets. Two answers to the same question form a positive pair if both are correct, and a negative pair if one is correct and the other is not. Pairs of two incorrect answers therefore never appear as positives during training. This raises a natural concern: the encoder might only learn to group correct answers together, and fail to group incorrect answers that share the same meaning.

We argue that this concern does not hold, because correctness enters the pipeline only through the construction of triplets. The encoder never receives a correctness label as input and is never trained to predict one. The objective in Eq. 7 depends only on inner products between answer embeddings, so what the encoder learns is a notion of *equivalence* between answers, not of *truth*. For a given question, all correct answers share the same denotation, which makes correctness a convenient and reliable source of equivalent pairs. The notion of equivalence the encoder learns from these pairs is not tied to the correct answer, and it transfers to answers that agree on the same incorrect content.

Tab. B.1 illustrates this behavior with two pairs of paraphrases answering the same question: one pair is correct and the other is incorrect. The encoder assigns nearly identical similarities to both pairs (0.975 and 0.980). It therefore groups paraphrases based on their shared meaning, regardless of whether that meaning is correct. This is precisely the behavior required by downstream semantic UQ aggregators: when a model consistently produces the same hallucination, the corresponding samples should fall into a single cluster, yielding low semantic entropy that reflects the model's (misplaced) confidence.

Table B.1: Similarity assigned by our encoder to pairs of semantically equivalent answers, for correct and incorrect content. Question: *"What is the capital of France?"*

Correctness	Answer 1	Answer 2	Similarity
Both correct	The capital of France is Paris.	Paris is the capital of France.	.975
Both incorrect	The capital of France is Madrid.	Madrid is the capital of France.	.980

B.2 Sensitivity to base encoder

A natural question is whether the gains come from the particular encoder we fine-tune or from the training framework itself. We therefore repeat the procedure of Sec. 4 with six public sentence encoders, covering different model sizes and pretraining strategies. For every backbone, we keep the training data, objective, margin, and number of epochs unchanged, and evaluate the resulting $\mathrm{COS}_{\mathrm{Enc}}$.

Tab. B.2 reports the mean AUROC over the 18 model–dataset combinations. The difference between the six trained backbones is small, only 0.009, and even the weakest trained encoder outperforms the best encoder before fine-tuning. Moreover, larger models are not consistently better, and the ranking of the encoders changes after training. These results suggest that the main gain does not come from a particular backbone or from simply improving an already strong representation. Instead, our training adds a question-conditioned notion of answer equivalence that transfers across different encoder architectures.

B.3 Negatives are necessary, not incidental

The objective in Eq. (7) has two goals: bring equivalent answers closer and push different answers apart. This raises a simple question: *are negative examples really needed, and does their choice matter?*

In our setting, the negatives are particularly informative. They are not unrelated sentences, but incorrect answers to the *same* question, generated by the same model. The anchor and the negative therefore share the same topic and often a similar structure, while differing mainly in the answer

Table B.2: **The framework is insensitive to the backbone.** Mean AUROC of $\mathrm{COS}_{\mathrm{Enc}}$ over the 18 model–dataset combinations, for six sentence encoders used as the starting point of the training procedure of Sec. 4, with all other settings held fixed.

Backbone	AUROC
`sentence-t5-large`	.759
`all-roberta-large-v1`	.759
`all-mpnet-base-v2`	.756
`multi-qa-mpnet-base-dot-v1`	.755
`bge-base-en-v1.5`	.752
`gtr-t5-large`	.750

Table B.3: **Hard negatives matter, not just negatives.** AUROC of $\mathrm{COS}_{\mathrm{Enc}}$ for the untrained encoder and under three training regimes: no negatives (cosine-similarity loss on positive pairs only), random negatives (drawn from a different question), and hard negatives (drawn from the same question, used throughout this paper), averaged over models (top) and over datasets (bottom).

	ADVQA	OKVQA	VizWiz	HotpotQA	TriviaQA	WebQuestions	Mean
hard negatives (ours)	**.694**	**.758**	**.741**	**.765**	**.852**	**.741**	**.759**
random negatives	.666	.690	.697	.738	.823	.618	.705
untrained encoder	.651	.673	.705	.707	.785	.554	.679
no negatives	.585	.616	.610	.687	.760	.640	.650
	`idefics2-8B`	`Qwen2.5-VL-7B`	`llava-1.5-7B`	`Qwen2.5-7B`	`Phi-3.5-mini`	`Mistral-7B`	**Mean**
hard negatives (ours)	**.730**	**.740**	**.723**	**.790**	**.793**	**.775**	**.759**
random negatives	.685	.690	.679	.738	.734	.707	.705
untrained encoder	.669	.693	.667	.688	.704	.653	.679
no negatives	.596	.628	.588	.711	.740	.636	.650

itself. To separate them, the encoder cannot rely only on topic, style, or length. It must focus on the part of the response that carries the answer. We therefore refer to these examples as *hard negatives*.

To study their role, we compare four settings using the same encoder and training questions. The first is the *untrained encoder*. The second uses *only positive pairs, with no negative term*. The third uses *random negatives, sampled from answers to different questions*. The last uses *the hard negatives of our method, sampled from incorrect answers to the same question as the anchor*.

Tab. B.3 shows a clear difference. The mean AUROC is 0.679 for the untrained encoder, 0.650 without negatives, 0.705 with random negatives, and 0.759 with hard negatives. Training only on positive pairs even performs worse than the untrained encoder, suggesting that pulling examples together without any repulsive term can lead to a less informative representation. Random negatives improve the result, showing that pushing different examples apart is important. However, they remain clearly below hard negatives: answers from different questions can often be separated using topic alone. In contrast, same-question negatives share the topic and differ mainly in the answer, forcing the encoder to learn factual differences. This explains why our training uses negatives from the same question rather than random responses from the full training set.

B.4 Sensitivity to Training Hyperparameters

The operator is trained with a margin of 0.2 and a learning rate of 2×10^{-5}, fixed once and used in all main experiments. To test whether our results depend strongly on these choices, we vary the margin and learning rate separately while keeping all other settings unchanged.

Tab. B.4 reports the AUROC of $\mathrm{COS}_{\mathrm{Enc}}$ for the different configurations. The results vary only slightly, from 0.749 to 0.765. Even the weakest setting still improves over the untrained encoder by 0.070. The hyperparameters used in the main paper are also not the best-performing ones: using a margin of 0.1 improves AUROC by 0.006. We nevertheless keep the original configuration, since tuning the hyperparameters on the evaluation settings would make the comparison less fair.

Table B.4: **Small variations across hyperparameters.** AUROC of $\mathrm{COS}_{\mathrm{Enc}}$, averaged over the 18 model–dataset combinations, when the margin and learning rate are varied around the configuration used throughout (shaded). All other settings are held fixed.

Margin	Learning rate	AUROC
0.10	2×10^{-5}	.765
0.15	2×10^{-5}	.758
0.20	2×10^{-5}	.759
0.30	2×10^{-5}	.757
0.50	2×10^{-5}	.749
0.20	1×10^{-5}	.759
0.20	3×10^{-5}	.761
0.20	5×10^{-5}	.758

B.5 Token-Level Localization of the Ground Truth

In Sec. 5.2, we interpret the token norm $\|\boldsymbol{h}_t\|_2$ as a measure of how much token t contributes to the semantic representation of the answer. This interpretation leads to a simple prediction: if large norms really identify the most important parts of the answer, then the highest-weighted token should often lie on the ground-truth answer itself. We test this directly.

We use `Mistral-7B` on the **TriviaQA** val set. We keep only greedy generations that are labelled correct and contain exactly one annotated ground-truth alias as a substring. This leaves 232 examples out of 1,000. For each example, we compute the token weights w_t on the generated answer, conditioned on the question, and match the ground-truth text span to the encoder tokens using character offsets.

The token with the largest weight falls inside the ground-truth span in 167 out of 232 cases, corresponding to 72%. For comparison, a token drawn uniformly at random from the answer would fall inside the ground-truth span only 3.9% of the time. This corresponds to an improvement of roughly $18\times$ over chance. Importantly, the encoder receives no token-level supervision during training: the triplet objective of Eq. 7 only compares complete answers. The result therefore suggests that the learned representation naturally localizes the part of the response that carries the factual answer.

Figure B.1 illustrates this behavior on three generations for the same question. In each case, most of the weight is placed on the answer token, even when its position changes: it may appear at the end of the sentence, at the beginning, or after an unrelated introductory clause. In contrast, tokens describing the topic of the question receive little weight. For example, words such as `France` and `capital` are relevant to the topic, but they are shared by both correct and incorrect answers and therefore do not help distinguish them.

This behavior is consistent with our training objective. A hard negative answers the same question as the anchor and often shares most of its wording, while differing mainly in the factual answer. To separate the two, the encoder must therefore focus on the part of the sentence that changes the answer, rather than on words that only describe the general topic.

B.6 Sensitivity to the Clustering Threshold

SE and CAE require a binary matrix, so the continuous $\boldsymbol{S}^{\mathrm{E}}$ must be thresholded before the aggregator can be applied. We use $\tau = 0.5$ as a natural threshold.

Tab. B.5 reports $\mathrm{CAE}_{\mathrm{Enc}}$ as τ is varied. Performance rises monotonically over the range tested, from 0.656 at $\tau = 0.3$ to 0.751 at $\tau = 0.7$, and $\tau = 0.5$ is therefore not the best available setting: $\tau = 0.7$ is 0.027 higher. The sweep has not reached a maximum, so a still stricter threshold might do better again.

We report the sweep rather than adopt its best point. The framework of Sec. 4 is trained entirely outside the uncertainty pipeline, with no UQ supervision and on data disjoint from every evaluation setting, and selecting τ against the evaluation results would introduce exactly the dependence the

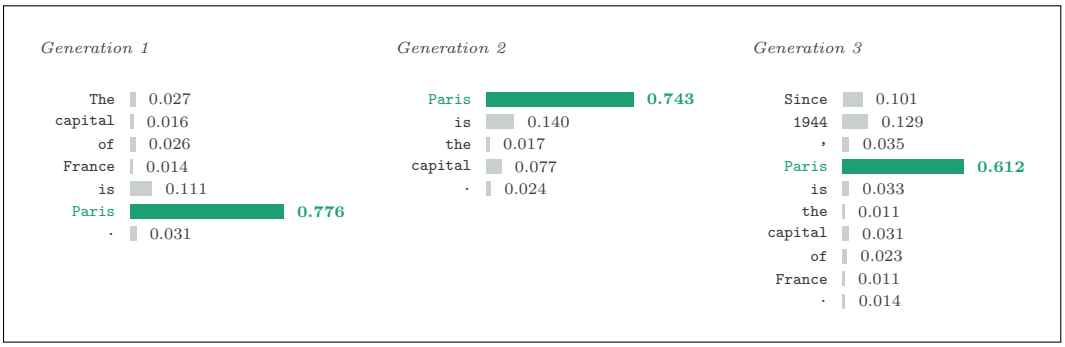

Figure B.1: **Token weights concentrate on the answer span.** Weights w_t from Eq. 8 for three generations answering *"What is the capital of France?"*. The answer token carries the majority of the weight wherever it appears in the sentence, while the tokens naming the question's subject receive almost none.

rest of the design avoids. The numbers in Tab. 1 are therefore obtained with the principled value, and the gains reported there understate what a tuned threshold would give.

Table B.5: **Sensitivity to the clustering threshold.** AUROC and ECE of $\mathrm{CAE}_{\mathrm{Enc}}$ over the 18 model–dataset combinations, as the threshold τ applied to $\boldsymbol{S}^{\mathrm{E}}$ is varied. The value used throughout the paper is shaded.

	AUROC	ECE
$\tau = 0.3$	.656	.139
$\tau = 0.4$	.693	.138
$\tau = 0.5$	.724	.140
$\tau = 0.6$	.743	.137
$\tau = 0.7$	.751	.141

C Training the operator

C.1 Training Hyperparameters

We fine-tune a `sentence-t5-large` encoder using triplet loss (see Eq. 7) with cosine distance and a margin of 0.2, training with a batch size of 32 and a learning rate of 2×10^{-5} under a linear warmup/decay schedule. We train for up to 30 epochs, with early stopping after 3 epochs without improvement in validation triplet accuracy.

C.2 Construction of the training set

Generation and labelling. We use the complete **NQ-OPEN** val split (3,160 questions). For each we sample $N = 10$ responses from `Llama-3.1-8B-Instruct` with temperature 1.0, top-p 0.9 and top-k 50, matching the sampling regime used at evaluation so that the operator is trained on the kind of text it will later compare. Each generation is labelled correct or incorrect against the reference by the procedure of App. D, partitioning the 10 generations of each question into a correct subset $\mathcal{P}$ and an incorrect subset $\mathcal{N}$. No model or dataset used in Sec. 6 appears at any point in this pipeline.

Triplets. A question is retained only when $|\mathcal{P}| \geq 2$ and $|\mathcal{N}| \geq 1$, leaving 1,720 of the 3,160. For each retained question a triplet is drawn uniformly as

$$\boldsymbol{y}^a \sim \mathcal{P}, \qquad \boldsymbol{y}^+ \sim \mathcal{P} \setminus \{\boldsymbol{y}^a\}, \qquad \boldsymbol{y}^- \sim \mathcal{N},$$

with all three sharing the same question, which is prepended to each before encoding. Both the anchor and the positive come from $\mathcal{P}$: two answers can be incorrect without being incorrect in the

same way, so a pair drawn from $\mathcal{N}$ would not reliably be equivalent, and using such pairs as positives would teach the encoder to pull distinct errors together. Triplets are resampled at every epoch, so each question contributes a different pair each time it is visited.

D LLM AS A JUDGE

We rely on an LLM-as-a-judge procedure at several stages of our pipeline. This section clarifies exactly how and where it is used.

Judging protocol. Given a question, an answer generated by the model under study, and the ground-truth answer, we prompt a larger external LLM to decide whether the generated answer matches the ground truth. The judge is never asked whether the generated answer is correct in its own opinion. Its role is restricted to comparing the generated answer with the reference, which accounts for paraphrases, formatting differences, or partial verbosity that exact-match or token-overlap metrics would miss. This text-to-text comparison is a simple task for a large instruction-tuned model. We use this procedure to annotate the training set.

Assigning a correctness label to each question. Evaluating uncertainty quantification (UQ) methods requires a single binary label per question, and two strategies are common in the literature. In the first, a greedy answer is generated alongside the sampled answers used to compute the uncertainty measures, and this greedy answer is labelled with the judging protocol above. In the second, each of the $N = 20$ sampled answers is labelled individually, and the question receives the majority label among them. Both strategies are reasonable. We adopt the greedy-answer strategy for our main results because it is the more direct of the two.

Separate judges for training and evaluation. To rule out any dependency between the labels used for training and those used for evaluation, we use a different judge for each split: Qwen2.5-32B-Instruct annotates the training set, and Llama-3.1-70B-Instruct annotates the test set. We have no specific reason to expect such a dependency, but using distinct judges removes it by construction.

Robustness to the labeling strategy. The performance of our method does not depend on the aggregation strategy. The main results table reports scores computed with greedy-answer labels, and App. G.3 compares results under both strategies, showing that the conclusions are consistent.

E EVALUATION PROTOCOL

E.1 MODELS, DATASETS AND SPLITS

We evaluate on six datasets, again covering both settings, each on a fixed subset of an official split. The visual question answering benchmarks are **ADVQA** (Li et al., 2021) (last 2,200 questions of the validation set), **OKVQA** (Marino et al., 2019) (first 5,000 of the validation set) and **VizWiz** (Gurari et al., 2018) (last 4,400 of the training set); the textual ones are **HotpotQA** (Yang et al., 2018) (first 5,000 of the validation set), **TriviaQA** (Joshi et al., 2017) (first 1,000 of the validation set) and **WebQuestions** (Berant et al., 2013) (first 2,000 of the test set). Each model is evaluated on the three datasets of its modality, giving the 18 model–dataset combinations reported throughout.

E.2 PROMPTS AND DECODING

For every question, we sample $N = 20$ answers with nucleus sampling (temperature 1.0, top-$p = 0.9$, top-$k = 50$). Each model is prompted with the original question of the dataset, preceded by a short instruction: "Answer the question about this image." for visual benchmarks and "Answer the following question." for textual ones. We deliberately impose no constraint on the length or form of the answer, so that the sampled generations retain their natural variability in phrasing and verbosity, which is precisely what the operator must see through.

E.3 ESTIMATOR HYPERPARAMETERS

SE, CAE and KLE use the entailment model `microsoft/deberta-v2-xlarge-mnli` He et al. (2020) as their operator. For KLE, we report the heat-kernel variant with $t = 0.3$ and the Matérn variant with $\kappa = 1.0$ and $\nu = 1.0$, the hyperparameters that gave the best results for this baseline. COS-int embeds each generated answer using the last-layer hidden states of the generating model itself, and COS-ext uses `sentence-t5-large` Ni et al. (2022), the pretrained sentence encoder from which our operator is fine-tuned. COS-ext and $\mathrm{COS}_{\mathrm{Enc}}$ therefore share the same architecture and initialization, and differ only in our training.

F LIMITS OF EXISTING OPERATORS

F.1 THE CONTEXT TRAP (COS-EXT)

To test whether a general-purpose encoder separates answers that differ only on the fact that matters, we construct pairs that are near-identical in phrasing but factually opposed. The pipeline, applied to the **NQ-OPEN** train split, is:

1. **Generate.** For each query, take the model's greedy decoded answer together with the list of ground-truth reference strings.
2. **Filter.** Retain the entry only if the greedy answer contains an exact substring match to at least one reference, so that every retained example is verifiably correct rather than partial or paraphrased.
3. **Corrupt.** Prompt `Llama-3.1-8B-Instruct` with the question, the matched ground truth and the greedy answer, asking it to replace the correct fact with a plausible but different incorrect one, preserving sentence structure and style.

This yields $1{,}000$ pairs that agree in topic, length, and phrasing and disagree only on the fact determining correctness:

Question	"Who wrote the book The Origin of Species?"
Greedy (correct)	The Origin of Species was written by **Charles Darwin** and published in 1859.
Rewritten (incorrect)	The Origin of Species was written by **Alfred Wallace** and published in 1859.

On these pairs, the base encoder gives a mean cosine similarity of 0.91: it does not separate them. The same encoder after our training gives 0.45.

F.2 THE AUTOREGRESSIVE BIAS (COS-INT)

Hidden-state representations are read at the final answer token, so they are shaped most by what the sequence ends on. To test whether this displaces the fact that determines correctness, we construct pairs that end identically and disagree earlier. The pipeline, applied to the **NQ-OPEN** train split, is:

1. **Sample and label.** For each query, take the 10 sampled answers and their per-sample correctness labels, and split them into a correct and an incorrect subset.
2. **Match on suffix.** Search for a correct/incorrect pair sharing their last four words, so that the two answers converge on an identical closing phrase while disagreeing on the fact stated earlier in the sentence.
3. **Extract.** Teacher-force each answer as the continuation of the question in a single forward pass with `Llama-3.1-8B-Instruct`, and take the final-layer hidden state at the last answer token.

Natural samples rarely end on the same short phrase while diverging in correctness, so the criterion is restrictive: 491 queries yield such a pair.

Question	"Who played the Joker in The Dark Knight"
Correct	Heath Ledger **played the role in the film.**
Incorrect	Jack Nicholson **played the role in the film.**

The answers differ only in their opening span, which carries the entire answer, and agree on everything the representation reads last. Cosine similarity between the two hidden states is 0.869 on average: the representation is dominated by the shared suffix rather than by the content preceding it. Our operator, on the same pairs, gives 0.350.

F.3 THE ENTAILMENT TRAP (NLI CROSS-ENCODERS)

Entailment is a directional relation, and two answers can convey the same fact while differing in how much else they say. To test whether this fractures valid semantic clusters, we construct pairs that are equivalent by construction but maximally apart in verbosity. The pipeline, applied to the **NQ-OPEN** train split, is:

1. **Sample and label.** For each query, take the 20 sampled answers and their per-sample correctness labels.
2. **Keep the correct ones.** Discard every answer not labelled correct, so that each remaining answer is independently verified as conveying the reference fact.
3. **Pair the extremes.** Pair the shortest correct answer with the longest, giving one pair per query spanning the range of natural response verbosity.

This yields 1,000 pairs in which a concise entity-only answer is matched against a verbose paraphrase of the same fact.

Question	"Who wrote the book The Origin of Species?"
Shortest	**Charles Darwin**.
Longest	The Origin of Species was written by **Charles Darwin**, who published it in 1859.

Averaging the predicted entailment probability in both directions, the NLI cross-encoder scores these pairs at 0.291, although every pair is factually equivalent by construction: the additional correct detail of the longer answer blocks entailment in one direction, and the objective penalizes verbosity rather than tracking agreement. Our operator gives 0.801 on the same pairs.

F.4 THE DISTRIBUTION MISMATCH (OOD SYNTAX)

Beyond architectural mismatches, supervised metric spaces suffer a domain shift: operators trained on human-annotated corpora expect well-formed sentences, while QA references are often bare entities. To quantify this, we compare each generated sentence against the reference string it matches. The pipeline, applied to the **NQ-OPEN** train split, is:

1. **Generate.** For each query, take the model's greedy decoded answer.
2. **Retrieve the reference.** Take the dataset's raw ground-truth string, which is typically a single entity rather than a sentence.
3. **Pair.** Keep the pair when the generated answer is labelled correct, so that every pair is equivalent by construction while differing in syntactic form.

This yields 1,000 pairs of a complete sentence against a bare entity.

Question	"What is the capital of France?"
Generated	The capital of France is **Paris**.
Reference	**Paris**

Both conventional operators collapse on these pairs: the NLI cross-encoder scores them at 0.265 and the base sentence encoder at 0.362, although each pair is a trivially correct match. Both conflate syntactic completeness with semantic overlap, and so fail on precisely the fragmented strings that QA datasets use as references. Our operator reaches 0.574 on the same pairs.

G COMPLETE RESULTS

G.1 PER-CELL RESULTS

Tab. G.1 and Tab. G.4 report every estimator's per-model AUROC and ECE with bootstrapped 95% confidence intervals, for the textual and visual benchmarks, respectively, underlying the averages of Tab. 1.

G.2 QUESTION-CONDITIONING ABLATION

The operator of Sec. 4 reads the question alongside each generation, which desiderata **(D2)** requires. To isolate that choice, we repeat the entire procedure without it: the encoder is trained on answer strings alone, and evaluated the same way, so the comparison removes the question from both stages rather than only from inference. Everything else is unchanged.

Tab. G.2 reports the result for $\mathrm{COS}_{\mathrm{Enc}}$. Mean AUROC falls from 0.759 to 0.742. The gap is smaller than the gain over the conventional operators, and we do not claim conditioning as the main source of the improvement, but it is present in almost every one of the 18 model–dataset combinations, ranging from $+0.006$ to $+0.026$. Its consistency across settings that share no model, no dataset and no modality is what makes it interpretable as an effect of the conditioning rather than of any particular evaluation.

G.3 GREEDY VS. MAJORITY-VOTE LABELS

As described in App. D, each question can be assigned a correctness label in two ways: by judging a single greedy answer, which is what we use in the main results, or by taking the majority label over the $N = 20$ sampled answers. Tab. G.3 reports the AUROC of every estimator under both strategies, together with the gain our variant brings over its base counterpart. Absolute scores are slightly higher with majority-vote labels for all methods, but the conclusions do not change. In every block, our variant outperforms its base counterpart under both strategies; the ranking within each block is identical, and the gains are of similar magnitude, slightly larger under majority vote.

Table G.1: **Per-cell results on the textual benchmarks.** AUROC (↑) and ECE (↓) with bootstrapped 95% confidence intervals, for each estimator in its published form (*base*) and with our operator substituted (*shaded*), reported for each of the three text-only models.

	Qwen2.5-7B		Phi-3.5-mini		Mistral-7B	
Method	AUROC	ECE	AUROC	ECE	AUROC	ECE
HotpotQA						
CAE	.683 ±.070	.234 ±.050	.651 ±.021	.259 ±.014	.685 ±.016	.212 ±.012
CAE_{Enc}	**.724** ±.067	**.216** ±.051	**.722** ±.019	**.235** ±.014	**.719** ±.015	**.193** ±.012
SE	.684 ±.067	.234 ±.046	.653 ±.020	.258 ±.014	.686 ±.015	.212 ±.013
SE_{Enc}	**.726** ±.064	**.215** ±.050	**.726** ±.021	**.235** ±.014	**.722** ±.016	**.193** ±.012
KLE-Heat	.709 ±.066	**.207** ±.050	.714 ±.019	**.231** ±.014	.733 ±.015	**.186** ±.013
$\text{KLE-Heat}_{\text{Enc}}$	**.761** ±.061	.223 ±.050	**.762** ±.019	.240 ±.015	**.765** ±.014	.198 ±.012
KLE-Matern	.704 ±.064	**.225** ±.049	.703 ±.020	**.249** ±.014	.738 ±.015	**.203** ±.012
$\text{KLE-Matern}_{\text{Enc}}$	**.760** ±.060	.226 ±.046	**.762** ±.019	.243 ±.014	**.766** ±.014	.201 ±.012
COS-int	.752 ±.058	**.221** ±.050	.688 ±.034	.253 ±.025	.719 ±.014	.210 ±.012
COS-ext	.709 ±.063	.229 ±.050	.702 ±.020	**.252** ±.014	.709 ±.016	**.208** ±.012
COS_{Enc}	**.761** ±.060	.228 ±.051	**.760** ±.020	.246 ±.014	**.775** ±.013	**.203** ±.012
TE	**.785** ±.057	.219 ±.049	**.704** ±.020	.245 ±.014	**.725** ±.016	.200 ±.013
TE_{Enc}	.730 ±.062	**.215** ±.049	.688 ±.020	**.240** ±.014	.679 ±.016	**.194** ±.012
TriviaQA						
CAE	.677 ±.036	.165 ±.032	.462 ±.034	.232 ±.031	.611 ±.038	.240 ±.027
CAE_{Enc}	**.847** ±.025	**.126** ±.024	**.841** ±.025	**.087** ±.022	**.832** ±.030	**.191** ±.027
SE	.672 ±.033	.164 ±.029	.458 ±.038	.232 ±.030	.602 ±.039	.240 ±.027
SE_{Enc}	**.848** ±.025	**.121** ±.023	**.843** ±.026	**.088** ±.022	**.831** ±.030	**.190** ±.028
KLE-Heat	.789 ±.029	**.108** ±.026	.644 ±.038	**.084** ±.028	.808 ±.033	**.189** ±.026
$\text{KLE-Heat}_{\text{Enc}}$	**.860** ±.023	.122 ±.024	**.862** ±.022	.100 ±.022	**.845** ±.027	.196 ±.026
KLE-Matern	.777 ±.030	**.114** ±.026	.615 ±.035	**.137** ±.029	.795 ±.035	**.197** ±.028
$\text{KLE-Matern}_{\text{Enc}}$	**.859** ±.022	.124 ±.025	**.861** ±.023	.101 ±.022	**.844** ±.027	.199 ±.027
COS-int	.736 ±.031	.121 ±.027	.580 ±.034	.157 ±.032	.658 ±.022	.139 ±.027
COS-ext	**.790** ±.029	**.117** ±.026	**.808** ±.028	**.074** ±.025	**.756** ±.034	**.197** ±.027
COS_{Enc}	**.855** ±.023	.118 ±.026	**.863** ±.022	.104 ±.024	**.838** ±.028	.200 ±.027
TE	.720 ±.033	.114 ±.028	.659 ±.034	.095 ±.028	.628 ±.039	.215 ±.027
TE_{Enc}	**.800** ±.028	**.103** ±.026	**.774** ±.028	**.061** ±.024	**.734** ±.034	**.193** ±.030
WebQuestions						
CAE	.558 ±.025	.137 ±.022	.438 ±.025	.206 ±.021	.474 ±.026	.186 ±.022
CAE_{Enc}	**.737** ±.021	**.056** ±.016	**.729** ±.021	**.067** ±.019	**.681** ±.023	**.042** ±.019
SE	.557 ±.025	.136 ±.022	.436 ±.025	.202 ±.021	.470 ±.025	.186 ±.022
SE_{Enc}	**.736** ±.022	**.058** ±.017	**.733** ±.021	**.067** ±.020	**.689** ±.025	**.047** ±.019
KLE-Heat	.597 ±.025	.099 ±.021	.530 ±.027	.148 ±.021	.538 ±.024	.089 ±.021
$\text{KLE-Heat}_{\text{Enc}}$	**.755** ±.021	**.035** ±.017	**.751** ±.019	**.058** ±.018	**.712** ±.022	**.041** ±.016
KLE-Matern	.598 ±.025	.106 ±.019	.513 ±.025	.162 ±.021	.534 ±.024	.147 ±.022
$\text{KLE-Matern}_{\text{Enc}}$	**.755** ±.020	**.033** ±.018	**.751** ±.021	**.058** ±.021	**.711** ±.022	**.041** ±.016
COS-int	.610 ±.025	.164 ±.022	.546 ±.025	.202 ±.022	.651 ±.023	.066 ±.019
COS-ext	.565 ±.025	.125 ±.020	.603 ±.025	.087 ±.018	.493 ±.024	.193 ±.022
COS_{Enc}	**.753** ±.019	**.028** ±.016	**.757** ±.020	**.055** ±.020	**.713** ±.021	**.046** ±.017
TE	.641 ±.024	.094 ±.021	.615 ±.023	.098 ±.019	.511 ±.025	.171 ±.021
TE_{Enc}	**.694** ±.023	**.051** ±.019	**.680** ±.022	**.075** ±.020	**.592** ±.026	**.097** ±.021

Table G.2: **Conditioning on the question helps in every setting.** AUROC of $\mathrm{COS}_{\mathrm{Enc}}$ with and without the question, by dataset (top) and by model (bottom). In the answer-only variant, the question is removed from both training and evaluation.

	ADVQA	**OKVQA**	**VizWiz**	**HotpotQA**	**TriviaQA**	**WebQuestions**	Mean
with question	**.694**	**.758**	**.741**	**.765**	**.852**	**.741**	**.759**
answer only	.668	.735	.730	.759	.842	.719	.742
	`idefics2-8B`	`Qwen2.5-VL-7B`	`llava-1.5-7B`	`Qwen2.5-7B`	`Phi-3.5-mini`	`Mistral-7B`	Mean
with question	**.730**	**.740**	**.723**	**.790**	**.793**	**.775**	**.759**
answer only	.711	.725	.697	.776	.782	.761	.742

Table G.3: **AUROC under greedy and majority-vote correctness labels.** Scores are averaged over the 18 model–dataset combinations. Δ is the AUROC gain of our variant over its base counterpart (over COS-ext for the COS block). The last column indicates whether the ranking within the block is the same under both labelling strategies. Best value per block in bold.

	Greedy		Majority vote		
Method	AUROC	Δ	AUROC	Δ	Order preserved
CAE	.612		.629		✓
$\mathrm{CAE}_{\mathrm{Enc}}$	**.724**	+.112	**.759**	+.130	
SE	.610		.626		✓
$\mathrm{SE}_{\mathrm{Enc}}$	**.726**	+.116	**.761**	+.135	
KLE-Heat	.677		.706		✓
$\mathrm{KLE\text{-}Heat}_{\mathrm{Enc}}$	**.756**	+.079	**.794**	+.088	
KLE-Matern	.682		.713		✓
$\mathrm{KLE\text{-}Matern}_{\mathrm{Enc}}$	**.756**	+.074	**.794**	+.081	
COS-int	.635		.653		
COS-ext	.679		.704		✓
$\mathrm{COS}_{\mathrm{Enc}}$	**.759**	+.080	**.797**	+.093	
TE	.640		.645		✓
$\mathrm{TE}_{\mathrm{Enc}}$	**.678**	+.038	**.690**	+.045	

Table G.4: **Per-cell results on the visual benchmarks.** AUROC (↑) and ECE (↓) with bootstrapped 95% confidence intervals, for each estimator in its published form (*base*) and with our operator substituted (*shaded*), reported for each of the three vision-language models.

	idefics2-8B		Qwen2.5-VL-7B		llava-1.5-7B	
Method	AUROC	ECE	AUROC	ECE	AUROC	ECE
ADVQA						
CAE	.644 ±.021	.115 ±.018	.631 ±.024	.127 ±.020	.635 ±.022	.107 ±.019
CAE_{Enc}	**.649** ±.022	**.085** ±.019	**.698** ±.021	**.059** ±.019	**.642** ±.022	**.099** ±.020
SE	.646 ±.022	.111 ±.019	.629 ±.025	.129 ±.020	.636 ±.022	.103 ±.018
SE_{Enc}	**.652** ±.023	**.083** ±.020	**.702** ±.022	**.060** ±.019	**.645** ±.023	**.098** ±.020
KLE-Heat	.670 ±.023	.093 ±.021	.675 ±.022	.074 ±.019	**.679** ±.022	**.073** ±.018
$\text{KLE-Heat}_{\text{Enc}}$	**.678** ±.020	**.071** ±.018	**.712** ±.020	**.073** ±.019	.659 ±.021	.088 ±.018
KLE-Matern	**.686** ±.021	.073 ±.017	.690 ±.021	.070 ±.017	**.688** ±.020	**.067** ±.017
$\text{KLE-Matern}_{\text{Enc}}$	.679 ±.022	**.070** ±.017	**.712** ±.021	**.070** ±.018	.660 ±.023	.081 ±.020
COS-int	.571 ±.024	.121 ±.021	.582 ±.024	.131 ±.020	.579 ±.023	.101 ±.020
COS-ext	.641 ±.022	.094 ±.018	.684 ±.022	.087 ±.020	.629 ±.024	.101 ±.020
COS_{Enc}	**.694** ±.022	**.071** ±.020	**.707** ±.021	**.075** ±.018	**.682** ±.022	**.071** ±.018
TE	.601 ±.025	.103 ±.020	.612 ±.023	.115 ±.020	.611 ±.023	**.080** ±.021
TE_{Enc}	**.636** ±.022	**.082** ±.019	**.635** ±.023	**.074** ±.020	**.637** ±.024	**.080** ±.020
OKVQA						
CAE	.678 ±.016	.240 ±.012	.597 ±.017	.226 ±.013	.654 ±.016	.220 ±.012
CAE_{Enc}	**.724** ±.015	**.210** ±.012	**.749** ±.015	**.179** ±.013	**.725** ±.015	**.192** ±.013
SE	.677 ±.016	.239 ±.013	.598 ±.017	.223 ±.013	.652 ±.017	.221 ±.012
SE_{Enc}	**.723** ±.016	**.210** ±.012	**.749** ±.015	**.179** ±.012	**.727** ±.016	**.191** ±.012
KLE-Heat	.699 ±.015	**.206** ±.012	.652 ±.016	**.169** ±.013	.684 ±.016	**.186** ±.013
$\text{KLE-Heat}_{\text{Enc}}$	**.765** ±.014	.219 ±.012	**.764** ±.014	.189 ±.013	**.761** ±.014	.201 ±.013
KLE-Matern	.723 ±.016	**.220** ±.012	.654 ±.016	**.185** ±.012	.715 ±.015	**.201** ±.013
$\text{KLE-Matern}_{\text{Enc}}$	**.764** ±.014	.222 ±.012	**.763** ±.014	.190 ±.013	**.760** ±.014	.203 ±.012
COS-int	.592 ±.017	**.199** ±.013	.626 ±.016	.210 ±.013	.646 ±.017	**.201** ±.012
COS-ext	.674 ±.017	.224 ±.012	.677 ±.016	**.187** ±.013	.668 ±.015	.202 ±.013
COS_{Enc}	**.759** ±.014	.224 ±.012	**.763** ±.014	.193 ±.013	**.751** ±.014	.207 ±.013
TE	.613 ±.017	.223 ±.012	.566 ±.017	.186 ±.012	.602 ±.016	.206 ±.012
TE_{Enc}	**.660** ±.016	**.216** ±.012	**.610** ±.016	**.184** ±.012	**.653** ±.016	**.198** ±.012
VizWiz						
CAE	.641 ±.017	.219 ±.013	.649 ±.017	.235 ±.013	.653 ±.015	.142 ±.013
CAE_{Enc}	**.660** ±.017	**.183** ±.013	**.703** ±.017	**.194** ±.013	**.657** ±.017	**.101** ±.015
SE	.639 ±.017	.218 ±.014	.646 ±.017	.233 ±.013	.647 ±.016	.141 ±.014
SE_{Enc}	**.662** ±.018	**.182** ±.013	**.701** ±.017	**.193** ±.014	**.662** ±.017	**.101** ±.014
KLE-Heat	.693 ±.017	**.187** ±.013	.689 ±.016	**.191** ±.014	.689 ±.015	**.104** ±.013
$\text{KLE-Heat}_{\text{Enc}}$	**.728** ±.016	.193 ±.013	**.744** ±.016	.204 ±.014	**.725** ±.015	.112 ±.015
KLE-Matern	.711 ±.016	.202 ±.013	.700 ±.016	.211 ±.014	.725 ±.015	.122 ±.014
$\text{KLE-Matern}_{\text{Enc}}$	**.729** ±.016	**.197** ±.013	**.744** ±.015	**.207** ±.013	**.727** ±.015	**.116** ±.015
COS-int	.612 ±.017	**.183** ±.013	.665 ±.018	.215 ±.013	.645 ±.016	**.115** ±.014
COS-ext	.693 ±.018	.203 ±.013	.717 ±.017	.212 ±.014	.703 ±.015	.121 ±.014
COS_{Enc}	**.738** ±.016	.201 ±.013	**.749** ±.016	**.211** ±.013	**.736** ±.014	.121 ±.015
TE	.632 ±.017	.199 ±.013	.660 ±.018	.215 ±.013	.644 ±.016	.122 ±.012
TE_{Enc}	**.654** ±.017	**.194** ±.014	**.688** ±.017	**.204** ±.014	**.662** ±.016	**.116** ±.014