ZINA: Multimodal Fine-grained Hallucination Detection and Editing

Supplementary Material

A. Taxonomy Construction

We designed the taxonomy in three steps:

1. Initial Design. Since our focus is on hallucinations in image captioning, we first drew inspiration from standard image captioning evaluation, specifically SPICE [4]. SPICE defines object, attribute, and relation as the core components of image captions. Given their established role in image captioning, we adopted these three as candidate categories. We further referred to existing taxonomies such as POPE [29], AMBER [49], and UniHD [11]. While POPE and AMBER include object and attribute, UniHD additionally introduces text and fact hallucinations. Based on these works, we initially selected five categories.

2. Pilot Annotation Experiment. To check the taxonomy's coverage, we then conducted a pilot annotation experiment of 200 samples by a small group of annotators. Feedback from this experiment revealed frequent errors related to number and grammar. Since grammatical errors are generally excluded from hallucination definitions, we added number as a new candidate.

3. Validation through Large-scale Annotation. To validate the final taxonomy, we performed a crowdsourced annotation task for about 2k samples. As shown in Table 1, even GPT-4o exhibited hallucinations across all six categories, supporting the validity of our taxonomy.

B. Evaluation Metrics

Span-level metrics. By utilizing [22, 56], BERT-F_1 and CLIP-F_1 are computed by setting the similarity function $\mathrm{sim}(\cdot,\cdot)$ to $\mathrm{BERTScore}(\cdot,\cdot)$ and $\mathrm{CLIPScore}(\cdot,\cdot)$, respectively. The precision, recall, and F_1 score are defined as follows:

$$\mathrm{Precision} = \frac{\sum_{i\in N} \mathrm{sim}(\hat{y}^{(i)}_{\mathrm{text}}, y^{(i)}_{\mathrm{text}})}{\sum_{i\in N} \mathbf{1}[\hat{y}^{(i)}_{\mathrm{type}} \neq 0]}, \tag{5}$$

$$\mathrm{Recall} = \frac{\sum_{i\in N} \mathrm{sim}(\hat{y}^{(i)}_{\mathrm{text}}, y^{(i)}_{\mathrm{text}})}{\sum_{i\in N} \mathbf{1}[y^{(i)}_{\mathrm{type}} \neq 0]}, \tag{6}$$

$$\mathrm{F}_1 = 2 \cdot \frac{\mathrm{Precision} \cdot \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}. \tag{7}$$

Following [56], we used DeBERTa embeddings [21] for computing BERTScore[1].

[1] https://huggingface.co/microsoft/deberta-xlarge-mnli

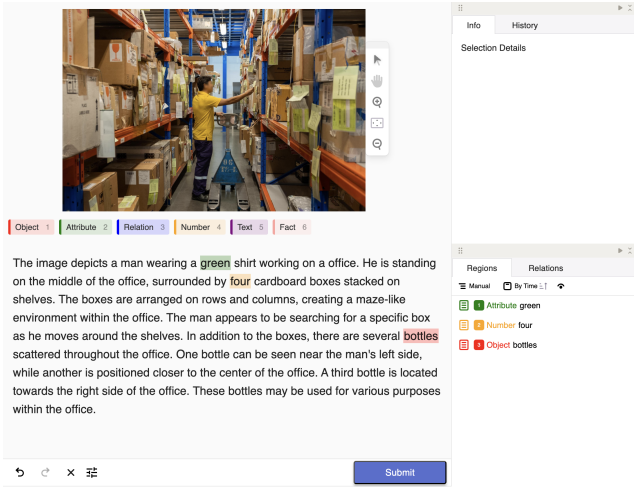

Figure 4. Annotation interface used for the VisionHall dataset.

Sentence-level metrics. We employed CLIP-S [22] and PAC-S [43] for sentence-level evaluation. We selected them because they are standard metrics for image captioning evaluation.

CLIP-S first obtains the embeddings t, v, and r from CLIP [41] for the edited text, image, and reference, respectively, and then computes the mean of the cosine similarities. The score $\mathrm{Score}(t, v, r)$ of CLIP-S is defined as follows:

$$S(x, y) = w \cdot \max\big(\cos(x, y),\, 0\big), \tag{8}$$

$$\mathrm{Score}(t, v, r) = \mathrm{mean}\big(S(t, v),\, S(t, r)\big), \tag{9}$$

where w denotes a rescaling factor of 2.5 [22]. Similarly, PAC-S adopts the same form but uses a CLIP fine-tuned for the task-specific domain. These metrics assign a single score to the edited text, indicating how well it aligns with both the image and the reference.

As sentence-level metrics are standard in hallucination editing evaluation [38, 45], we report CLIP-S and PAC-S alongside BERT-F_1 and CLIP-F_1. CLIP-S and PAC-S rely on CLIP [41], which limits input text to a maximum of 77 tokens. Therefore, previous works [36, 52] used modified variants of these metrics. These variants compute the cosine similarity between each sentence in a paragraph and the image, and then average the values to obtain the final score. Following this approach, we applied the same sentence-wise averaging for CLIP-S and PAC-S.

C. Construction of VisionHall Dataset

Text-editing vs. image-editing. Although modifying text to inject hallucinations is the standard approach [38, 40], some studies (e.g. [8]) generate hallucinations via image editing. These image-editing approaches are problematic because they can introduce image-text discrepancies through uncontrolled visual artifacts (e.g. alterations in non-target regions). In contrast, text-editing methods modify textual descriptions, enabling better-controlled errors; we therefore follow this text-editing approach.

Annotation Process. We employed LabelStudio [2] for collecting the VisionHall dataset. Fig. 4 illustrates the annotation interface. We conducted annotations via a public crowdsourcing platform, where we recruited annotators from a broad online population without imposing restrictions on demographic or geographic background. We recruited annotators and provided payment that was adequate based on the participants' country of residence. We also obtained consent through the task instructions, which explicitly stated that the collected data would be used for research purposes.

The full text of the instructions given to participants is as follows:

Annotation Instructions

You will be given an image along with a caption describing its content. Your task is to identify any parts of the caption that contain hallucinations and label them using one of the six predefined categories:
- Spatial Relation Hallucination
- Errors in spatial relationships (Note: Grammatical mistakes in prepositions do NOT count as hallucinations.)
- For example:
If the caption says "an apple on the table", but the apple is not actually on the table, you should select the word "on".
If the caption says "an apple in to the table", the phrase "in to" is a grammatical error, so you should not select any words.
- Attribute Hallucination <attribute>
- Errors in descriptive attributes, such as adjectives or adverbs.
- For example:
If the caption says "a blue apple", but the apple is not actually blue, you should select the word "blue".
- Object Hallucination <object>
- Errors in naming specific objects or entities.
- For example:
If the caption says "a red apple", but there are no apples and instead a book is present, you should select the word "apple".
- Number Hallucination <number>
- Errors related to numerical values or quantities.
- For example:
If the caption says "two red apples", but there are actually three apples, you should select the word "two".
- Factual Hallucination <fact>
- Errors concerning named entities (e.g., incorrect facts or identities).
- For example:
If the caption says "Steve Jobs with his family", but Steve Jobs is not present and instead it's Bill Gates, you should select "Steve Jobs".
- Text Hallucination <text>
- Errors involving scene text (i.e., text appearing within the image).
- For example:
If the caption says "a sign that says STOP", but the sign actually says "GO", you should select the word "STOP".
To label a word as a hallucination, simply click on it. To change the category of the tag, click on the "Categories" section shown in the interface. You can ignore grammatical mistakes - they are not considered hallucinations.

[2] https://github.com/HumanSignal/label-studio

Metric	Value
Total graphs	108,918
Total nodes	130,968
Avg. connected components	5.45
Avg. nodes	6.55
Avg. edges	2.53
Avg. degree	2.06

Table 6. Structural statistics of the pruned dependency graphs

Ethical considerations. The collected data contains no personally identifiable or offensive content. All data used in this study are publicly available. We also confirmed that the source websites, repositories, and publications include no statements indicating concerns about personal information.

D. Details of Datasets

D.1. VisionHall Dataset

VisionHall comprises 6,854 MLLM-generated descriptions for 4,759 images, collected from 211 annotators, for the detection task. For the editing task, VisionHall contains 20k synthetic samples by our novel graph-based method. Each sample includes an English reference caption, with a total of 986,488 words, an average length of 143.9 words, and a vocabulary size of 21,757. The MLLM-generated descriptions are also in English, totaling 1,189,673 words,

Model	Detection F_1	BERT-F_1	CLIP-F_1	CLIP-S	PAC-S
(i) $d < 2$	37.92	34.48	40.70	65.53	74.09
(ii) $p = 0.25$	42.90	37.59	43.87	65.52	74.06
(iii) ZINA	**45.15**	**44.02**	**50.39**	**66.08**	**74.36**

Table 7. Ablation on graph-pruning strategies. (i) degree-based pruning with threshold $d < 2$ (from Table 6); (ii) random pruning with probability $p = 0.25$; (iii) original ZINA. ZINA outperformed both pruning variants, while degree-based performs the worst.

with an average length of 173.6 words and a vocabulary size of 42,446. The test set for both tasks consists of 400 samples collected from the human annotators. Its size is comparable to that of MHaluBench [11], whose test set for image captioning contains 200 samples. We used the training set to train the model and the test set to evaluate model performance.

As shown in Table 2, for most categories, the span lengths are generally comparable. Regarding tag frequency, we found that the overall distribution was also similar across real and synthetic data. One exception is the Attribute category, which appears less frequently in the synthetic set. This is because the Error Insertion module is intentionally designed to frequently select rarer categories such as Fact or Text, in order to improve coverage across diverse error types. As a result, attribute hallucinations are relatively underrepresented. These analyses suggest that the overall distribution gap remains relatively small in both span length and tag frequency.

Table 6 presents structural statistics of the pruned dependency graphs, including the total number of graphs and nodes, as well as the average number of connected components, nodes, degree, and edges per graph. Each sample contains an average of 5.45 connected components, with each graph consisting of an average of 6.55 nodes and 2.53 edges. On average, each node is connected to 2.06 other nodes, suggesting sufficient structural complexity in the constructed graphs.

D.2. MHaluBench

MHaluBench [11] is a benchmark for evaluating coarse-grained hallucination detection. This dataset provides samples with coarse-grained annotations in both Image-to-Text and Text-to-Image settings. We evaluated the baselines and our proposed method on the "Image-to-Text" subset of MHaluBench, as our task focuses on image captioning. This subset includes 200 samples for the image captioning task and 200 samples for the VQA task. The generated texts are collected using MLLMs such as LLaVA [34].

Evaluation. MHaluBench provides both segment-level and claim-level annotations. For the segment-level task, each sentence in the generated texts was labeled as either hallucinatory or non-hallucinatory. For the claim-level task, claims extracted from these sentences were further annotated with hallucination labels. An entire response is labeled hallucinatory if it contains at least one hallucinatory segment.

Baselines. The authors of MHaluBench used two baselines: a Gemini-based Self-Check and a GPT-based Self-Check. Self-Check [11] detects hallucinations using chain-of-thought reasoning without relying on external knowledge. Following this setup, we employed these two baselines, as well as UniHD, on MHaluBench.

E. Implementation Details

Training. We finetuned Qwen2.5-VL-72B-Instruct[3] using LoRA [24] with a rank of 8. The model was trained for 10 epochs with a batch size of 4. We set the maximum sequence length to 2048 tokens and the initial learning rate to 1.0×10^{-4}. We use the AdamW optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and apply cosine annealing for learning rate scheduling.

We employed LLaMA-Factory [57] to train ZINA on the VisionHall dataset. For training, we adopted gradient accumulation and ZeRO-3 optimization to improve memory efficiency and scalability. All models were trained using 4 NVIDIA H200 SXM GPUs, each with 141 GB of VRAM.

Evaluation. We used VLMEvalKit [15] to ensure reproducibility for the following models: LLaVA-1.5-7B [32], LLaVA-1.5-13B, Qwen2-VL-7B [51], LLaVA-OV-Qwen2-7B [26], LLaVA-NeXT-Qwen-32B [33], LLaVA-OV-Qwen2-72B, LLaMA-3.2-90B-Vision-Instruct [19], Qwen2.5-VL-72B-Instruct [7], and GPT-4o [1]. All experiments were reported based on a single run.

F. Additional Experiments

F.1. Ablation on Data Curation Choices

We conducted experiments to analyze the impact of data curation design choices. Specifically, we compared graph pruning strategies under two conditions: (i) deterministically pruning nodes with degree $d < 2$, based on the average degree shown in Table 6; and (ii) randomly pruning nodes

[3] https://huggingface.co/Qwen/Qwen2.5-VL-72B-Instruct

Method	Two-stage	Detection	Editing		Overall	
		F_1	CLIP-S	PAC-S	BERT-F_1	CLIP-F_1
LLaVA-1.5-7B [32]		0.82	64.01	72.72	0.66	0.93
LLaVA-1.5-7B [32]	✓	3.39	64.41	72.70	3.39	3.39
Qwen2-VL-7B [51]		3.36	64.79	73.01	3.62	4.98
Qwen2-VL-7B [51]	✓	13.13	64.70	73.04	13.79	16.41
LLaVA-OV-Qwen2-7B [26]		3.39	64.06	72.40	3.39	3.39
LLaVA-OV-Qwen2-7B [26]	✓	5.38	64.40	72.73	4.58	5.33
LLaVA-1.5-13B [32]		4.73	64.74	73.02	5.08	6.71
LLaVA-1.5-13B [32]	✓	4.14	64.46	72.72	3.87	4.05
LLaVA-NeXT-Qwen-32B [33]		19.09	65.34	73.47	24.29	31.06
LLaVA-NeXT-Qwen-32B [33]	✓	12.75	64.40	72.69	9.49	12.60
Llama-3.2-90B-Vision-Instruct [19]		16.92	65.28	73.54	14.56	17.62
Llama-3.2-90B-Vision-Instruct [19]	✓	16.74	64.93	73.40	15.30	17.20
Qwen2.5-VL-72B-Instruct [7]		21.31	64.38	72.99	18.85	23.67
Qwen2.5-VL-72B-Instruct [7]	✓	<u>35.21</u>	65.29	73.65	<u>34.06</u>	<u>39.26</u>
LLaVA-OV-Qwen2-72B [26]		25.70	65.74	73.91	20.81	26.81
LLaVA-OV-Qwen2-72B [26]	✓	34.92	65.73	73.03	31.83	36.80
GPT-4o (w/o images) [1]		27.02	65.66	<u>73.99</u>	23.34	27.99
GPT-4o (w/o images) [1]	✓	21.90	<u>65.85</u>	73.10	21.46	25.10
GPT-4o [1]		29.37	65.58	73.86	24.89	30.19
GPT-4o [1]	✓	31.78	65.56	73.83	31.48	35.90
ZINA (Ours)	✓	**45.15**	**66.08**	**74.36**	**44.02**	**50.39**

Table 8. Quantitative comparison with single-stage and two-stage baselines on the VisionHall dataset. **Bold** font indicates the best, and <u>underlined</u> font indicates the second best. Our proposed method outperformed both single-stage and two-stage baselines in both tasks.

with probability $p = 0.25$, where p denotes the probability of selecting a node for removal along with its descendants.

The results are summarized in Table 7. We observed that the degree-based pruning strategy led to worse performance compared to the original ZINA. This is likely due to the reduced complexity of inter-error dependencies, which results in a greater number of independent, unstructured errors.

Similarly, pruning nodes with $p = 0.25$ yielded slightly lower performance than the original setting ($p = 0.5$), likely because the inserted errors became slightly more complex and began to deviate from those observed in real data. These analyses support the appropriateness of our current design choices for data curation.

F.2. Additional Quantitative Comparison

Two-Stage Variants of Baselines. Although it is standard to compare system-level models with single-step MLLMs [11, 38], we also compare ZINA with two-stage variants of the baseline MLLMs, in addition to the original MLLMs. These variants are modified versions of the baselines used in Table 3, converted into the same two-stage pipeline as ZINA.

Table 8 presents the quantitative results of these two-stage comparisons. The results show that our method even outperformed the two-stage baselines in both tasks. Some MLLMs, such as LLaVA-OV-Qwen2-7B and Qwen2.5-VL-72B-Instruct, improved performance when converted to a two-stage setup. In contrast, models such as GPT-4o and LLaVA-NeXT-Qwen-32B exhibit degraded performance in two-stages. Together with Table 5, these results indicate that the performance gains primarily arise from the combination of semi-synthetic data and the two-step strategy, rather than from either component in isolation.

Method	Detection	Editing		Overall	
	F_1	CLIP-S	PAC-S	BERT-F_1	CLIP-F_1
LLaVA1.5-13B†	7.70	63.91	73.03	7.68	7.42
Qwen2.5-VL-72B†	24.00	64.95	73.06	19.95	24.18
LLaVA-OV-Qwen2-72B†	29.68	65.80	73.97	23.32	27.21
ZINA (Ours)	**45.15**	**66.08**	**74.36**	**44.02**	**50.39**

Table 9. Additional experiments. † represents fine-tuned versions.

Finetuned Baselines. Table 9 shows a quantitative comparison between ZINA and the finetuned baselines. The results show that ZINA outperformed the finetuned baselines, indicating that the performance improvements mainly stem from the combination of synthetic data and the two-step generation strategy, rather than either component in isolation.

F.3. Quantitative Comparison with HalLocalizer

To evaluate the proposed method on an out-of-domain dataset, we assess ZINA on the HalLoc [40] dataset. Table 10 shows that ZINA outperformed HalLocalizer in 3 out of 4 categories. This result indicates generalization beyond VisionHall, despite the fundamental differences in dataset construction, i.e., HalLoc injects token-level errors via hallucinated QA, whereas VisionHall inserts span-level errors with a graph-based approach.

Method	Object	Attribute	Relationship	Scene
$\text{HalLocalizer}_{\text{best}}$	0.68	0.64	**0.71**	0.76
ZINA (Ours)	**0.82**	**0.71**	0.64	**0.92**

Table 10. Results of ZINA on HalLoc. Each column represents the F_1 score for the respective category.

G. Licenses and Intended Use

ZINA is released under the BSD 3-Clause License, and the VisionHall dataset is released under the CC BY-NC 4.0 License. Table 11 summarizes the licenses of the models and datasets used in this study.

All existing artifacts used in this study were utilized in a manner consistent with their intended use. For the artifacts we created, we define their intended use as general academic and research use, which is compatible with the original access conditions of the datasets and models employed in this study.

Model / Dataset	License
DCI dataset [47]	CC BY-NC 4.0
MHaluBench [11]	MIT
InstructBLIP [13]	Research-only (non-commercial)
InternVL [12]	MIT
LLaVA-NeXT [33]	Apache 2.0
LLaVA-1.5 [32]	Apache 2.0
Multimodal-GPT [18]	Apache 2.0
Qwen-VL-Chat [6]	Tongyi Qianwen
ShareGPT4V [10]	Apache 2.0
BLIP-2 [28]	BSD 3-Clause
GIT [50]	MIT
Long-CLIP [55]	Apache 2.0
Qwen2.5-VL [7]	Apache 2.0
BERTScore [56]	MIT
CLIPScore [22]	MIT
PAC-S [43]	MIT
ZINA	BSD 3-Clause
VisionHall	CC BY-NC 4.0

Table 11. Licenses of models and datasets used in this study.

H. Prompts

H.1. Baselines

The full prompts used for baselines are described below. These prompts are designed based on prior work [38]. In pilot experiments, we observed that MLLMs occasionally misused the `<relation>` tag to indicate grammatical relationships rather than spatial relationships between objects, and applied the `<fact>` tag to entire sentences instead of to specific named entities. To address these issues, we adopt `<spatial_relation>` in place of `<relation>`, and `<named_entities_fact>` in place of `<fact>`.

Full Prompt for Baselines

Given a passage with errors of image captions, identify any <spatial_relation>, <attribute>, <object>, <number>, <named_entities_fact>, <text> errors in the passage and add edits for errors.
If there are no errors, return the passage with no tags.
Any changes to the original passage should be marked in <> tags.

Below are the error definitions followed by examples of what you need to follow.
Definitions:
Types of Errors and Definitions:
1. **<spatial_relation> (Spatial Relation Error)**: Incorrect spatial relationship between entities or concepts. (Only those involving spatial relationships are permitted, not grammatical relationship errors.)
- Example: "an apple on the table" → "an apple <spatial_relation>under</spatial_relation> the table"
2. **<attribute> (Attribute Error)**: Incorrect attributes such as adjectives or adverbs describing objects or concepts.
- Example: "Red sky" → "<attribute>Blue</attribute> sky"
3. **<object> (Object Error)**: Incorrect specific objects or entities.
- Example: "There is a chair on the table." → "There is a <object>book</object> on the table."
4. **<number> (Number Error)**: Incorrect quantities or numerical values.
- Example: "3 cats" → "<number>5</number> cats"
5. **<named_entities_fact> (Factual Error)**: Error related to named entities. This tag should not be applied to anything other than named entities. For example, "John F. Kennedy Center" is acceptable, but "center" is not.

- Example: "The image is of the John F. Kennedy Center." → "The image is of the <named_entities_fact>white house</named_entities_fact>."
6. **<text> (Text Error)**: Incorrect scene texts (text that appears in an image).
- Example: "A car is parked under a sign that says 'Restaurant'." → "A car is parked under a sign that says <text>'Hotel'</text>."
Example
Passage: [few-shot-example]
Reference: [few-shot-example]
Edited: [few-shot-example]
Task
Now detect errors and include edits in the following passage like done in the example above.
Include error tags <> for ANYTHING YOU CHANGE IN THE ORIGINAL PASSAGE.
Passage: [Original]
Reference: [Reference]
Edited:

H.2. Detection and Reviewer MLLMs

The full prompts used for the detection MLLM $\mathcal{M}_{\text{det}}$ and the reviewer MLLM $\mathcal{M}_{\text{rev}}$ are described below. These prompts are also designed based on prior work [38].

Full Prompt for $\mathcal{M}_{\text{det}}$

You are given an image, a generated caption (Original), and a human reference caption (Reference).
Your task is to detect any single word in the Original that is NOT supported by the image
(hallucinations) and label each of them with exactly one tag from:
• object – wrong or missing object
• spatial_relation – wrong spatial or positional term
• attribute – wrong adjective or adverb
• number – wrong quantity
• text – incorrect visible text
• named_entities_fact – incorrect named entity
Return ONLY a comma-and-slash separated list of "word, tag" pairs, e.g.
three, number / apples, object
If no hallucinations exist, return exactly:
none
———————- FEW-SHOT EXAMPLES ———————-
1. number
Original : "There are three cats."
Reference: "Two cats are on the sofa."
Output : three, number
2. spatial_relation
Original : "An apple on the table."
Reference: "An apple is under the table."
Output : on, spatial_relation
3. attribute
Original : "Red sky over the mountains."
Reference: "Blue sky over the mountains."
Output : Red, attribute
4. object
Original : "There is a chair on the table."
Reference: "There is a book on the table."
Output : chair, object
5. named_entities_fact
Original : "The image shows the John F. Kennedy Center."
Reference: "The image shows the White House."
Output : John F. Kennedy Center, named_entities_fact
6. text
Original : "A sign says 'Restaurant'."
Reference: "A sign says 'Hotel'."
Output : Restaurant, text
———— NOW PROCESS THIS SAMPLE ————
Original:
[Original]
Reference:
[Reference]
Output:

Full Prompt for $\mathcal{M}_{\text{rev}}$

You are given an image, the same reference caption, and an "Original" caption
in which candidate hallucination words have been wrapped with XML tags:
<object> . . . </object>
<spatial_relation> . . . </spatial_relation>
<attribute> . . . </attribute>
<number> . . . </number>
<text> . . . </text>
<named_entities_fact> . . . </named_entities_fact>
For EACH tagged word decide:
– If it must be corrected, return the corrected word.
– If it is already correct, return the original word unchanged.
Return ONLY a result "tagged_segment: replacement", e.g.
<object>chair</object>: book
———————- FEW-SHOT EXAMPLES ———————-
1. number: wrong or missing object
Original : There are <number>three</number> cats.
Reference: Two cats are on the sofa.
Output : <number>three</number>: two
2. spatial_relation: wrong spatial or positional term
Original : An apple <spatial_relation>on</spatial_relation> the table.

Reference: An apple is under the table.
Output : <spatial_relation>on</spatial_relation>: under
3. attribute: wrong adjective or adverb
Original : <attribute>Red</attribute> sky over the mountains.
Reference: Blue sky over the mountains.
Output : <attribute>Red</attribute>: Blue
4. object: wrong object
Original : There is a <object>chair</object> on the table.
Reference: There is a book on the table.
Output : <object>chair</object>: book
5. named_entities_fact: incorrect named entity
Original : The image shows the <named_entities_fact>John F. Kennedy Center</named_entities_fact>.
Reference: The image shows the White House.
Output : <named_entities_fact>John F. Kennedy Center</named_entities_fact>: White House
6. text: incorrect visible text
Original : A sign says <text>'Restaurant'</text>.
Reference: A sign says 'Hotel'.
Output : <text>'Restaurant'</text>: 'Hotel'
———— NOW PROCESS THIS SAMPLE ————
Original:
[Original]
Reference:
[Reference]
Instructions:
- Only look at segments already wrapped in XML tags in the Original (‘<object>...</object>‘, ‘<spatial_relation>...</spatial_relation>‘, etc.).
- Do **not** add any new tags.
- Decide for **each** existing tag whether it needs correction, but then choose **only one** tagged segment to report (the first or most obvious error).
- Output **exactly one** line in the form:
‘<tag>word</tag>: corrected_word‘
—and nothing else.

H.3. Synthetic Training Data Curation

The full prompts used for the Error Insertion (EI) module are shown below:

Full Prompt for Inserting Errors

Objective:
Insert six types of errors (as defined below) into the provided candidate sentence (cand) while following the guidelines. Use the provided references (refs) as needed.
Instructions:
- **Modify Only Existing Text:**
- Only change words or phrases that already appear in the candidate sentence (cand).
- Do not add any extra words, phrases, or information that are not in cand.
- The overall content must remain exactly as in the original cand aside from the inserted error tags.
- **Maintain Consistency:**
- When replacing a word, update every reference to that word in the sentence to preserve consistency.
- *Example:* Changing “three apples” to “seven apples” requires updating all mentions of “three apples” in the sentence.
- If you replace one term (e.g., “train station” with “bus stop”), ensure that all instances of “train station” are changed accordingly.
- **Scene Text Errors:**
- Insert text errors only when there is an existing reference to scene text in the original sentence.
- **Error Annotation Format:**
- Use the following XML-like format to mark errors:
- ‘<object original="original object name" id="E1">wrong object name</object>‘
- ‘<attribute original="original attribute" id="E2" parent="E1">wrong attribute</attribute>‘
- **Rules for Error Tags:**
- Each error must include:
- The type of error.
- A sequential ID.
- The original text.
- A parent dependency (if applicable).
- Once an object is mislabeled, all subsequent references to it must consistently reflect the same error.
- If subsequent text depends on an existing error, use the same parent ID to denote this dependency.
- Do not nest error tags.
- Do not use the same ID for both a current element and its parent.
- After applying modifications, ensure there are no duplicate IDs.
- Modifications must be applied to whole words only (do not split a word into parts). For example, changing "beach" by replacing "bea" with "city" and "ch" with "desert" is not allowed.
- Do not insert tags that result in semantically equivalent replacements. For example, changing "a" to "<number original="a" id="E1">one</number>" is not allowed.
- **Types of Errors:**
1. **spatial_relation (Spatial Relation Error):**
- Modify only prepositions that indicate spatial relationships (no grammatical errors allowed).
- *Example:*
‘"an apple on the table"‘ →
‘"an apple <spatial_relation original="on" id="E1">under</spatial_relation> the table"‘
2. **attribute (Attribute Error):**
- Change descriptive attributes such as adjectives or adverbs.
- *Example:*
‘"Red sky"‘ →
‘"<attribute original="Red" id="E1">Blue</attribute> sky"‘
3. **object (Object Error):**
- Replace names of generic objects or entities that are common nouns (i.e., not proper names).
- *Example:*
‘"There is a chair on the table."‘ →
‘"There is a <object original="chair" id="E1">book</object> on the table."‘

- Note: For any proper noun or official name (e.g., "John F. Kennedy Center" or "Fuji"), use the fact error as described next.
4. **fact (Factual Error):**
- Replace proper, named entities with incorrect names. Apply this error type only to entities that are proper names (e.g., formal institution names, landmarks, or official titles).
- *Example:*
‘"The image is of the John F. Kennedy Center."‘ →
‘"The image is of the <named_entities_fact original="John F. Kennedy Center" id="E1">white house</named_entities_fact>."‘
- Important: For any non-proper, generic object or entity (even if it might appear to be an entity), use the object error type instead.
5. **number (Number Error):**
- Change numerical values or quantities.
- *Example:*
‘"3 cats"‘ →
‘"<number original="3" id="E1">5</number> cats"‘
- If you change a singular quantity to plural (or vice versa), make sure the associated noun reflects the correct number (i.e., use plural form for plural numbers and singular form for singular numbers). No annotation tags are needed for singular/plural noun form adjustments.
- *Example:*
‘"a cat"‘ →
‘"<number original="a" id="E1">5</number> cats"‘
6. **text (Text Error):**
- Change scene text (i.e., text that appears within the image).
- *Example:*
‘"A car is parked under a sign that says ’Restaurant’."‘ →
‘"A car is parked under a sign that says <text original="’Restaurant’" id="E1">’Hotel’</text>."‘
- **Output Format:**
- Provide the modified sentence along with error annotations in the following JSON format:
“‘json
{
"original": "Two apples in front of a cat. The apples are Fuji.",
"modified": "<number original="Two" id="E1">nine</number> <object original="apples" id="E2">bananas</object> <spatial_relation original="in front of" id="E3">on</spatial_relation> <object original="a cat" id="E4">a table</object>. The <object original="apples" id="E5" parent="E2">bananas</object> are <named_entities_fact original="Fuji" id="E6">Cavendish</named_entities_fact>."
}
“‘
- **Important:**
- Only modify words that exist in the original cand.
- Do not add any additional words or phrases.
- **Input:**
- **Original Sentence (cand):**
cand
- **Reference (refs):**
refs

Full Prompt for Checking and Revising

You are a quality control assistant tasked with evaluating the output ‘generated_xml‘ produced by the LLM (which is the modified version of the provided ‘original‘ candidate sentence). Your objective is to ensure that ‘generated_xml‘ fully complies with all specified guidelines. Please review the output using the checklist below and then provide a JSON-formatted report detailing your findings. If modifications are necessary, include the revised output.
Checklist:
1. **Original Sentence Integrity:**
- Verify that the value of the ‘"original"‘ key exactly matches the provided candidate sentence.
- Ensure that no modifications, additions, or extra words/phrases have been introduced in the ‘"original"‘ value.
2. **Modified Sentence Requirements:**
- Confirm that the ‘"modified"‘ value is the candidate sentence with error annotations inserted.
- Check that only words or phrases from the candidate sentence have been modified—no new content should be added.
- *Note:* The meaning of the ‘"modified"‘ value may differ from the original.
3. **Error Annotation Types:**
- Confirm that only the following six error types are used: **spatial_relation, attribute, object, number, named_entities_fact, text**.
- Verify that each error annotation exactly matches one of these types and is applied only in the appropriate context (for example, text errors are used only for scene text).
4. **Error Tag Format and Consistency:**
- Each error annotation must use the XML-like tag format:
“‘xml
<error_type original="original text" id="E#">modified text</error_type>
“‘
- Ensure that:
- The tag name is exactly one of the specified error types.
- The ‘original‘ attribute contains the exact original word or phrase from the candidate sentence.
- The ‘id‘ attribute is sequential (e.g., E1, E2, . . .) with no duplicates.
- For dependent errors, the child error tag must include the correct ‘parent‘ attribute referencing the appropriate error ID.
- Error tags should only be applied to whole words (no splitting of words), and no error tag should be nested within another.
- **Whitespace Requirement:** Since tags are treated like words, ensure there is a space immediately after a closing tag. For example, ‘<attribute>blue</attribute> <object>book</object>‘ is correct, whereas ‘<attribute>blue</attribute><object>book</object>‘ is not acceptable.
5. **Consistency in Modifications:**
- Verify that if a word is modified in one location, every occurrence of that word in the sentence is modified consistently.
- Ensure that the same error tag (and, if applicable, the same ‘parent‘ attribute) is used for all repeated instances of that word.
- When one modification depends on another error, ensure that a proper tree structure is maintained using the ‘parent‘ attribute.
6. **No Semantically Equivalent Replacements:**
- Ensure that error annotations do not result in semantically equivalent replacements (e.g., do not replace “a” with a number tag representing “one”).
7. **Singular/Plural Adjustments:**
- If changing a singular quantity to plural (or vice versa), ensure that the associated noun reflects the correct number (i.e., use plural form for plural numbers and singular form for singular numbers).
- When modifying the noun’s number, the change must be performed via a number error tag applied to the noun, with the number error tag for the quantity serving as the parent. If the noun is already modified by an object (or another) tag, the existing tag takes priority but the ‘parent‘ attribute should be added.
- **Example:**
- ‘a cat‘ → ‘<number original="a" id="E1">three</number> <number original="cat" id="E2" parent="E1">cats</number>‘
- ‘a cat‘ → ‘<number original="a" id="E1">three</number> <object original="cat" id="E2" parent="E1">dogs</object>‘

8. **No Article Swapping:**
- Modifications that swap articles (e.g., changing "a" to "the" or vice versa) are not permitted.
9. **No Unjustified Modifications:**
- Do not allow changes that cannot be determined solely from the original word. For example, altering ‘two bible ’Guide’‘ to ‘two <named_entities_fact>quran</named_entities_fact> <text>bible</text>‘ is unacceptable, as it is unrealistic to deduce the appropriate modification for "bible" from the candidate sentence alone.
10. **No Matching Words Between Tagged and Untagged Text:**
- Ensure that the words within the error tags do not exactly match the corresponding words in the original, untagged text. For instance, transforming "To the left" into ‘<spatial_relation>To the right</spatial_relation>‘ is incorrect because the words "To" and "the" are repeated. The proper transformation is to only enclose the modified word: "To the left" should become "To the <spatial_relation>right</spatial_relation>".
11. **Application of Error Annotations for Generic Objects or Entities:**
- Fundamental Principle: For generic objects or entities expressed by common nouns (i.e., non-proper nouns), do not use the named_entities_fact tag. Always use the object tag.
- named_entities_fact (Factual Error): This tag should only be used for proper names such as formal institution names, famous landmarks, or official titles. It is only applicable when an incorrect proper name needs to replace the original.
- object (Object Error): For common objects or entities (non-proper nouns), use this tag instead. In cases where the word does not represent a proper noun, the object tag must be used rather than named_entities_fact.
—
Output Format:
After your evaluation, please produce a JSON object with the following keys:
- ‘"need_update"‘: A boolean indicating whether the output requires modifications.
- ‘"updated_xml"‘: The revised output (if modifications are needed) or a copy of the compliant output.
- ‘"reason"‘: A detailed explanation listing any detected violations with clear references to the specific rule(s) broken, or a statement confirming full compliance if no issues are found.
Example:
- ‘original‘: a red apple
- ‘generated_xml‘: "<number original="a">3</number> <attribute original="red" >blue</attribute> <object original="apple">books</object>"
“‘json
{
"need_update": true,
"updated_xml": "<number original="a" id="E1">3</number> <attribute original="red" id="E2">blue</attribute> <object original="apple" id="E3" parent="E1">books</object>",
"reason": "yyy"
}
“‘
Input Data:
- **‘original‘:**
{original}
- **‘generated_xml‘:**
{output}