Introducing Verification Task of Set Consistency with Set-Consistency Energy Networks

Mooho Song, Hye Ryung Son, Jay-Yoon Lee*
Seoul National University
{anmh9161, hyeryung.son, lee.jayyoon}@snu.ac.kr

Abstract

Examining logical inconsistencies among multiple statements (such as collections of sentences or question-answer pairs) is a crucial challenge in machine learning, particularly for ensuring the safety and reliability of models. Traditional methods that rely on 1:1 pairwise comparisons often fail to capture inconsistencies that only emerge when more than two statements are evaluated collectively. To address this gap, we introduce the task of *set-consistency verification*, an extension of natural language inference (NLI) that assesses the logical coherence of entire sets rather than isolated pairs. Building on this task, we present the *Set-Consistency Energy Network* (SC-Energy), a novel model that employs a margin-based loss to learn the compatibility among a collection of statements. Our approach not only efficiently verifies inconsistencies and pinpoints the specific statements responsible for logical contradictions, but also significantly outperforms existing methods, including prompting-based LLM models. Furthermore, we release two new datasets: Set-LConVQA and Set-SNLI for set-consistency verification task.

1 Introduction

In machine learning (ML), examining whether multiple statements exhibit logical consistency is a crucial challenge, especially when evaluating models for their safe usage. Examples from document summarization and question answering provide intuitive reason for such evaluations. When a large language model (LLM) generates a summary of a document, inconsistencies may arise between the original document and the summary. Similarly, in question-answering tasks, ML models do not inherently guarantee consistent responses to semantically related questions. Consider the following question-answering example from (Tandon et al., 2019). If a model answers "Yes" to the question "Does CO_2 increase the population of polar bears?", it should not also answer "Yes" to the contradictory question "Does CO_2 decrease the population of polar bears?". Logical consistency among these pairs, and more broadly within a set, is required for reliability and underscores the need for effective inconsistency detection methods.

Few studies have addressed related problems, particularly in the domain of factual inconsistency detection. These studies primarily focus on identifying factual discrepancies between a document and its summary. However, detecting logical inconsistencies among multiple statements remains limited with these methods.

First, some studies do not incorporate a holistic view of consistency across multiple statements. For example, (Xu et al., 2024; Wang et al., 2020; Fabbri et al., 2022) generate questions based on summaries and then obtain answers from the original document and summaries simultaneously, assessing consistency between them. However, arbitrary question generations do not guarantee the detection of all subtle logical inconsistencies, making it challenging to capture a comprehensive view of consistency for multiple statements.

Second, even approaches that aim for a holistic view face limitations when verifying consistency among multiple statements. (Yang et al., 2024; Falke et al., 2019) evaluate factual consistency based on (NLI), typically by comparing the entailment score for all combinations of pairs of sentences (or chunk of sentences) present in a document and its summary. Although this pairwise comparing method considers all sentences in a document, it has inherent shortcomings as following:

- **Limited Scope of Inconsistency Detection:** Existing Method can only detect inconsistencies when two statements directly conflict each other. In other words, logical inconsistencies that emerge only when multiple statements are considered collectively may go undetected. For ex-

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 33346–33366

ample, consider the following three statements:

- *"Either the train arrives at 8 AM or it arrives at 9 AM."* (p or q)
- *"The train does not arrive at 8 AM."* ($\neg p$)
- *"The train does not arrive at 9 AM."* ($\neg q$)

While no two statements contradict each other, an inconsistency becomes evident when all three are considered together.

- **Combinatorial Complexity**: In large text corpora such as articles, combinatorial complexity of pairwise comparisons across all statements incur heavy computational costs.

- **Overly Sensitive Classification:** When examining inconsistencies in exhaustive pairwise comparisons, if even one of them detects an inconsistency, the entire document is deemed inconsistent in a naive way. As the number of statements increases, the probability of detecting at least one inconsistency also increases, rendering pairwise classifications impractical for a larger set.

To overcome these limitations, we propose the **Set-Consistency Energy Network** (SC-Energy), an approach that detects the logical inconsistencies across the entire statements within a set. SC-Energy treats a collection of natural language statements(*e.g.*, sentences or input-output pairs) as a set and employs a margin-based loss to learn the compatibility among them. Unlike traditional research related to energy networks that process single inputs (Belanger and McCallum, 2016; LeCun et al., 2006; Tu et al., 2019; Lee et al., 2022) , SC-Energy is designed to capture the compatibility across multiple data points. Even with a relatively compact architecture such as RoBERTa-base (Liu, 2019), our model outperforms LLMs such as GPT-4o, o3-mini from OpenAI[1] in effectively detecting inconsistencies.

Our key contributions are as follows:

- We introduce the task of *set-consistency verification*, which extends natural language inference beyond pairwise comparisons to assess the logical coherence of multiple statements. We show that LLMs lack such verification capability and highlight the importance of the task.

- We release two refactored datasets, Set-LConVQA and Set-SNLI, for *set-consistency verification* and *locate* tasks to facilitate further research in this area.

- We show that learning an energy space capable of distinguishing subtle differences in consistency across entire sets of statements, rather than binary classification of an isolated set, is crucial. This contrast is particularly highlighted in the *locate* task, which requires pinpointing the specific statements responsible for logical contradictions.

2 Related Works

Factual Inconsistency Detection Several studies have focused on identifying factual discrepancies in summarization tasks. In particular, research such as (Luo et al., 2023; Xu et al., 2024; Yang et al., 2024; Fabbri et al., 2022; Laban et al., 2022) has examined methods to detect inconsistencies between a document and its summary. These approaches typically involve decomposing the text into smaller units, generating common questions to compare source context and summary, or performing pairwise entailment-score comparisons for all combinations of individual sentences (or, chunk of sentences) from document and summary.

Structured Energy Networks Several studies have explored energy-based models for structured prediction tasks. Belanger and McCallum introduced Structured Prediction Energy Networks (SPENs), which use energy functions to model dependencies between output components in structured tasks (Belanger and McCallum, 2016). Tu et al. enhanced the joint training of inference networks and SPENs with a compound objective, leading to more effective optimization (Tu et al., 2019). Lee et al. proposed SEAL, a framework where an energy network is used as a loss function to guide the training of a separate neural network (Lee et al., 2022).

A characteristic feature of these studies is that they inject learning signals to contrast over points in the $(\mathcal{X} \times \mathcal{Y})$ space to train the energy network. Our work expands this concept by learning the energy surface in the $(\mathcal{X} \times \mathcal{Y})^*$ space, *i.e.*, over arbitrary number of input-output pairs (as discussed in Section 3).

3 Set-Consistency Energy Networks

3.1 Definitions

Several definitions are listed as follows.

[1]https://openai.com/

Data Consistency	Set-LConVQA Example	Set-SNLI Example
Consistent Set (S_C)	{("question": "what color is desk?", "answer": "brown"), ("question": "is desk brown?", "answer": "yes"), ("question": "is desk pink?", "answer": "**no**")}	{"If a couple walk hand in hand down a street, then a couple is walking together.", $(p \rightarrow q)$ "No couple is walking together.", $(\neg q)$ "**No couple walks hand in hand down a street.**" $(\neg p)$}
Inconsistent Set (S_I)	{("question": "what color is desk?", "answer": "brown"), ("question": "is desk brown?", "answer": "yes"), ("question": "is desk pink?", "answer": "**yes**")}	{"If a couple walk hand in hand down a street, then a couple is walking together.", $(p \rightarrow q)$ "No couple is walking together.", $(\neg q)$ "**Either a couple walk hand in hand down a street, or a couple is walking together.**"$(p \vee q)$}

Table 1: Examples of consistent (S_C) and inconsistent (S_I) sets from the Set-LConVQA and Set-SNLI datasets. The bolded portions indicate the different statements that distinguish the consistent set from the inconsistent set. The gray-colored propositional symbols explain the logical relationship in sentences. In the Set-LConVQA example, inconsistency is explicitly evident in the QA pairs (e.g., conflicting answers about the desk's color), whereas the Set-SNLI example illustrates more subtle logical discrepancies among sentences as it emerges from nuanced differences in phrasing and logical entailment rather than explicit factual contradictions.

A **statement** (s) refers to an individual unit of information, such as a sentence / sentences in a news article or a QA-pair in a question-answering dataset.

Set (S) refers to a collection of statements. The size of a set S, denoted as $|S|$, represents the number of statements within S. In this paper, we consider only finite sets.

If there exists a subset $\tilde{S} \subseteq S$ with $|\tilde{S}| \geq 2$ such that the statements in $\tilde{S}$ are logically inconsistent, then S is called an **inconsistent set**. Otherwise, S is referred to as a **consistent set**.

3.2 Notations

A statement s can be either a standalone sentence or a pair (x, y) (e.g., a QA-pair). We now present the notations for both cases.

Standalone Sentence Formulation: Let $\mathcal{S}$ denote the space of standalone sentences. In this formulation, the SC-Energy is defined as

$$E_\theta : \mathcal{S}^* \rightarrow \mathbb{R},$$

which is a parameterized function that takes an arbitrary number of sentences as input and produces a real-valued energy score. Note that, the asterisk (*) indicates that the function accepts a sequence (or set) of sentences drawn from $\mathcal{S}$, that is, an arbitrary number of sentences can be provided as input.

$(\mathbf{x}, \mathbf{y})$ **Pair Formulation:** Let $\mathcal{X}$ and $\mathcal{Y}$ denote the input and output spaces, respectively. For statements represented as QA-pairs, the SC-Energy is defined as

$$E_\theta : (\mathcal{X} \times \mathcal{Y})^* \rightarrow \mathbb{R}.$$

In this setting, the network processes an arbitrary number of (x, y) pairs and outputs a real-valued energy score.

For simplicity, we denote a consistent set as S_C and an inconsistent set as S_I. When constructing new sets by merging different sets, we concatenate their indices to indicate their composition. For instance, the union of two distinct consistent sets is denoted as S_{CC}. We consider the union of different consistent sets to remain consistent, thus, S_{CC} is essentially a consistent set and could be regarded as S_C. However, for more fine-grained analysis, rather than simplifying it with S_C, we explicitly denote the data generation process and label such set as S_{CC} in our experiments.

3.3 Training SC-Energy

The function E_θ is trained to assign lower energy values to consistent sets and higher energy values to inconsistent sets. Given a consistent set S_C and an inconsistent set S_I, the loss function L_E is defined as:

$$L_E(S_C, S_I) = \left[E_\theta(S_C) - E_\theta(S_I) + \alpha\right]_+,$$

where $[\cdot]_+ = \max(\cdot, 0)$ and α is a hyperparameter.

In Section 3.3.1, we discuss the methods for constructing S_C and S_I; detailed procedures for each dataset are provided in Section 4. In Section 3.3.2, we introduce the training methodology for SC-Energy using S_C and S_I.

3.3.1 Construction of S_C and S_I

We first introduce construction of S_C and S_I based on (x, y)-pair setting. Given a consistent

set $S_C = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, we construct an inconsistent set S_I such that it preserves the overall semantic content and form of S_C while introducing logical inconsistencies. The simplest example is to modify one output y_i in a QA-pair by replacing it with y_i^*, thereby yielding $S_I = \{(x_1, y_1), \ldots, (x_i, y_i^*), \ldots, (x_n, y_n)\}$. In this construction, S_I remains similar to S_C in terms of semantics and structure, yet the altered element induces a logical inconsistency. In the context of a set of sentences, a similar strategy can be adopted by prompting an LLM to regenerate some of the sentences in S_C to produce S_I.

These constructions can be carried out using purely rule-based methods (e.g., Set-LConVQA dataset) or through a combination of rule-based techniques and LLM-generated modifications (e.g., Set-SNLI dataset), which will be introduced in section 4 in detail.

3.3.2 Fine-grained Training of SC-Energy

We derive eight contrastive signals to train E_θ, capturing varying levels of logical consistency. Each of these eight contrasts contributes a loss term that guides the model to assign lower energy to more consistent sets and higher energy to more inconsistent ones. The loss function for SC-Energy incorporates the L_E values from all eight contrasts:

1. **Basic Contrast:**
   - (S_C vs. S_I): A direct comparison between a consistent set and an inconsistent set.
2. **Union-Based Contrasts:** By forming unions, we generate an additional consistent set S_{CC} and two extra inconsistent sets S_{CI} and S_{II}. This yields five additional comparisons:
   - (S_C vs. S_{CI}), (S_C vs. S_{II}), (S_{CC} vs. S_I), (S_{CC} vs. S_{CI}), (S_{CC} vs. S_{II}).
3. **Inconsistency Degree Contrasts**
   - **(S_{CI} vs. S_I):** In S_{CI}, we regard the presence of additional neutral (or consistent) elements alongside inconsistent ones makes it less contradictory overall than the fully inconsistent S_I. Thus, we regard S_{CI} to be more consistent (i.e., of lower inconsistency degree) than S_I.
   - **(S_I vs. S_{II}):** Here, S_{II} is formed by uniting multiple inconsistent sets, thereby amplifying the overall contradiction. Consequently, we treat S_{II} to have higher degree of inconsistency than S_I.

3.4 Inference and Thresholding

Since E_θ is a real-valued function, a predefined threshold is used to determine set-consistency. A set is classified as consistent if its energy score is below the threshold and inconsistent otherwise.

Detailed information on the training process using the eight contrasts, the conversion of a set S into the input for SC-Energy, and the selection of the threshold can be found in appendix A.

4 Datasets

To evaluate the proposed task of set-consistency verification, we introduce two new datasets: Set-LConVQA and Set-SNLI. Set-LConVQA is derived from the existing LConVQA dataset (Ray et al., 2019), and Set-SNLI is constructed by modifying the SNLI dataset (Bowman et al., 2015). Both datasets are designed to group multiple instances into logically consistent or inconsistent sets, thereby enabling systematic evaluation of set-consistency verification.

We provide four types of data splits for both datasets: training, validation1, validation2, and test sets. The training sets contain 6,754 and 6,225 consistent and inconsistent sets for each, and the validation1, validation2, and test sets contain 200 consistent and 200 inconsistent sets for each Set-LConVQA and Set-SNLI, respectively. Below, we describe each dataset in detail.

4.1 Set-LConVQA

Set-LConVQA is designed to assess whether a given set of question-answer (QA) pairs is mutually consistent or inconsistent. The original LConVQA dataset is a Visual Question Answering (VQA) dataset automatically generated from the Visual Genome scene graph by extracting object, attribute, and relation information to create logically consistent / inconsistent QA pairs. Set-LConVQA removes the image dependency and focuses purely on the textual QA pairs.

An example from the dataset is shown in Table 1. A key characteristic of Set-LConVQA is that in each inconsistent set S_I, there exists a specific pair of QA pairs, denoted as s_i and s_j, such that the size-2 set $\{s_i, s_j\}$ is inconsistent. This property distinguishes Set-LConVQA from the Set-SNLI

dataset, which is introduced in Section 4.2. Moreover, through manual verification, we confirmed that in all test instances where the set size is at least four, there is exactly one element in S_I whose removal renders the set consistent. Therefore, in addition to set-consistency verification, we can utilize Set-LConVQA for an additional task (see Section 5.2) to show the usefulness of SC-Energy. Further details regarding the Set-LConVQA dataset are provided in Appendix B.

4.2 Set-SNLI

Set-SNLI dataset can be used to evaluate the ability to determine whether a set of natural language sentences is logically consistent. Unlike traditional Natural Language Inference (NLI) tasks that assess relationships between a single premise and a hypothesis, Set-SNLI requires reasoning over multiple sentences to capture more complex logical interactions.

For example, consider the three sentences: "Either the train arrives at 8 AM or it arrives at 9 AM.", "The train does not arrive at 8 AM.", "The train does not arrive at 9 AM.". No two sentences contradict each other when examined pairwise, but collectively they introduce a contradiction. Since the ordering of sentences is irrelevant in this task, traditional entailment relationships in NLI cannot be determined. Instead, sets are classified as either consistent or inconsistent, with inconsistency implying that not all sentences in the set can simultaneously be true.

Inspired by (Nakamura et al., 2023), we construct Set-SNLI by transforming SNLI sentence pairs into multi-sentence sets using predefined logical rules. Examples are presented in Table 1, and detailed construction steps are provided in Appendix C.

In contrast to Set-LConVQA, which guarantees that for each inconsistent set S_I there always exists a size-2 subset $\tilde{S} \subset S$ that is inconsistent, this property is not assured in the Set-SNLI dataset. In Set-SNLI, some inconsistent sets exhibit this property while others do not.

Set-SNLI is more challenging than Set-LConVQA because the inconsistencies in Set-SNLI tend to be more subtle and require holistic reasoning over multiple sentences. While Set-LConVQA usually presents explicit factual contradictions (e.g., conflicting answers about a desk's color), the logical discrepancies in Set-SNLI arise from nuanced differences in phrasing and entailment among sentences, making it harder to detect inconsistencies.

5 Experiments

This section presents the experimental results of set-consistency verification and downstream task, demonstrating that SC-Energy outperforms other baselines. For performance comparison, the baselines are evaluated along two axes: **Model Architecture** and **Verification Strategy**.

Model Architecture: This axis categorizes the baseline models based on their output representations and training objectives. In particular, the differences lie in the output format: LLM-based models output natural language, binary classifiers output a 2-dimensional vector, and energy-based models output a single real-valued score. These models can be divided into three types:

1. **LLM-based**: Utilizes a pre-trained, frozen large language model to assess set-consistency via prompt-based querying. We employed the GPT-4o & o3-mini model from OpenAI, with few-shot with/without Chain-of-Thought prompting (Wei et al., 2022).

2. **Binary Classifier (2-dim vector)**: A conventional classification model with an output dimension corresponding to the number of labels (consistent/inconsistent), trained using a cross-entropy loss function.

3. **Energy-based (Real-valued)**: Following energy-based model's principle, this model produces a continuous real-valued output, trained such that consistent sets yield lower scores and inconsistent sets yield higher scores. Training is performed using a margin-based loss as in section 3.3.

Verification strategy: This axis defines how consistency is evaluated within a set:

1. **Element-wise Verification**: Similar to (Yang et al., 2024; Falke et al., 2019), this strategy evaluates logical inconsistency by comparing all possible pairs of statements within a set. For a set S of size $|S| = N$, this requires $\frac{N(N-1)}{2}$ pairwise comparisons, where each pair is assessed for 1:1 consistency.

2. **Set-level Verification**: Directly inputs the entire set into the model, reducing computational

overhead and capturing inconsistencies that may be overlooked by pairwise comparisons.

To clarify, our SC-Energy model employs a energy-based (real-valued) model architecture along with a set-level verification strategy. The training and evaluation procedures vary slightly depending on the model architecture and verification strategy used. For detailed information on the prompts and experimental methods, please refer to Appendix D.1.1.

Verification Strategy	Model Architecture	Training Data		Test Data	
		Set-LConVQA	Set-SNLI	Set-LConVQA	Set-SNLI
Element-wise	LLM - GPT-4o (few-shot w/o CoT)	-	-	0.584	0.533
	LLM - GPT-4o (few-shot w/ CoT)	-	-	0.566	0.493
	LLM - o3-mini (few-shot w/o CoT)	-	-	0.780	0.588
	LLM - o3-mini (few-shot w/ CoT)	-	-	0.908	0.655
	Binary Classifier	✓		$0.577_{\pm 0.096}$	$0.475_{\pm 0.055}$
			✓	$0.434_{\pm 0.040}$	$0.490_{\pm 0.016}$
		✓	✓	$0.607_{\pm 0.104}$	$0.556_{\pm 0.022}$
	Energy-Based	✓		$0.697_{\pm 0.097}$	$0.523_{\pm 0.032}$
			✓	$0.413_{\pm 0.017}$	$0.563_{\pm 0.034}$
		✓	✓	$0.440_{\pm 0.003}$	$0.534_{\pm 0.027}$
Set-level	LLM - GPT-4o (few-shot w/o CoT)	-	-	0.919	0.713
	LLM - GPT-4o (few-shot w/ CoT)	-	-	0.926	0.710
	LLM - o3-mini (few-shot w/o CoT)	-	-	0.964	0.918
	LLM - o3-mini (few-shot w/ CoT)	-	-	0.974	0.919
	Binary Classifier	✓		$0.985_{\pm 0.005}$	$0.474_{\pm 0.048}$
			✓	$0.533_{\pm 0.050}$	$0.973_{\pm 0.010}$
		✓	✓	$0.913_{\pm 0.099}$	$0.967_{\pm 0.011}$
	Energy-Based (SC-Energy)	✓		$\mathbf{0.987}_{\pm 0.005}$	$0.432_{\pm 0.023}$
			✓	$0.508_{\pm 0.063}$	$\mathbf{0.981}_{\pm 0.008}$
		✓	✓	$0.969_{\pm 0.018}$	$0.941_{\pm 0.023}$

(a) Macro-F1 scores for set-consistency verification

Model	EM	Precision	Recall	F1
LLM - GPT-4o (few-shot w/ CoT)	0.734	0.846	0.985	0.875
LLM - o3-mini (few-shot w/ CoT)	0.707	0.776	**0.997**	0.815
Binary Classifier	$0.784_{\pm 0.142}$	$0.880_{\pm 0.102}$	$0.993_{\pm 0.001}$	$0.910_{\pm 0.079}$
Energy-Based (SC-Energy)	$\mathbf{0.923}_{\pm 0.059}$	$\mathbf{0.957}_{\pm 0.033}$	$0.981_{\pm 0.015}$	$\mathbf{0.961}_{\pm 0.035}$

(b) Locate Task Performance

Table 2: (a) Macro-F1 scores for set-consistency verification across different verification strategies and model architectures, and (b) performance evaluation for the locate task in identifying inconsistent QA pairs. The best performance for each dataset is highlighted in bold. For both verification (a) and locate (b), SC-Energy shows the best performance for all of datasets. SC-Energy is superior at locate task (b) thanks to learning fine-grained degrees of inconsistencies across diverse sets. For trainable models (Binary Classifier and Energy-Based models), average and standard deviation for five different seed pairs are provided.

5.1 Set-Consistency Verification

Set-consistency verification determines whether a given set S is logically consistent or inconsistent. Beyond evaluating individual sets such as S_C and S_I, the consistency assessment extends to unions of multiple sets. To evaluate set-consistency verification performance, we generate diverse types of sets to assess the model's ability to generalize to previously unseen set configurations. The evaluation data includes not only S_C and S_I but also sets constructed by merging two, three, or four different sets, thereby generating up to 14 possible dataset combinations with different labels (for example, S_C, S_I, S_{CC}, $\cdots$, S_{IIII}).

Table 2(a) presents the Macro-F1 score results for set-consistency verification tasks across various model architectures and verification strategies. Since both the binary classifier and energy-based architectures are data-driven, the specific training datasets used for each model are indicated by check marks (✓) in the Training Data column. For example, if both Set-LConVQA and Set-SNLI are checked, it implies that the model was trained on both datasets simultaneously.

Regarding the *verification strategies*, the set-level verification strategy consistently outperforms the element-wise strategy across all model architectures, highlighting the importance of providing the entire set as input for verifying logical inconsistencies. Note that, the element-wise verification strategy tends to predict a set as inconsistent with high probability. In fact, when the model uniformly classifies all sets as either consistent or inconsistent, the corresponding Macro-F1 scores are only 0.222 and 0.416, respectively. These results suggest that the element-wise strategy does not achieve sufficiently high performance in set-consistency verification.

Although the overall Macro-F1 may appear high for set-level verification strategies, performance varies substantially by set type. As detailed in Appendix D.2, certain categories such as sets composed entirely of consistent statements (*e.g.*, S_{CCCC}) or those with only one inconsistent element (*e.g.*, S_{CCCI}) remain challenging for all models. For example, GPT-4o records only 88.5% on S_{CCCC} and 29% on S_{CCCI} in Set-LConVQA, and 83.5% and 87.5% on the same categories in Set-SNLI, which implies that some subsets of these two datasets are still challenging. Nevertheless, we deliberately retained these "easy" and "hard" mixes to reflect real-world uncertainty in set composition;

reporting per-type performance thus gives a more faithful picture of task complexity.

As mentioned earlier, in a element-wise verification approach, if any element-wise verification identifies an inconsistency, the entire document is theoretically deemed inconsistent. This raises the additional question: to what extent should a certain level of inconsistency be tolerated (even if this is not the ideal approach) in order to achieve optimal performance in set-consistency verification? Further details on this matter can be found in appendix D.2.1.

For models in set-level verification strategy, the binary-classifier model architecture exhibits performance comparable to the energy-based model SC-Energy. However, because SC-Energy learns fine-grained consistency across multiple sets, it performs well on a broader range of tasks beyond set-consistency verification. For further details, please refer to Section 5.2.

5.2 Locating Inconsistent Statements

To showcase the diverse applications of the models trained in Section 5.1, we introduce an additional task of *locate* which evaluate the ability to detect inconsistent statements within a set. We utilize Set-LConVQA dataset for the locate task. As mentioned in Section 4.1, through manual verification, we confirmed that in all sets in test data where the set size is at least four, there is exactly one element in S_I whose removal renders the set consistent. For the locate task, we evaluate and merge only those S_C and S_I sets that have a size of 4 or larger.

Similarly to the approach described in Section 5.1, for evaluation we use not only S_C and S_I but also sets constructed by merging two, three, or four different sets. This results in up to 14 possible dataset combinations with different labels (for example, S_C, S_I, S_{CC}, $\cdots$, S_{IIII}).

The locate task leverages the model trained in Section 5.1 without any additional task-specific training. For details on how the locate task is performed using the LLM-based, binary classifier, and energy-based model architectures, please refer to Appendix D.4.1.

We measure performance using EM (Exact Match), Precision, Recall, and F1-score. For instance, in the case of S_{IIII}, an exact match is achieved only when all four inconsistent QA-pairs contained in the four inconsistent sets are correctly identified. Table 2(b) shows the performances of set-level verification strategies regarding the locate task. As shown in Table 2(b), SC-Energy significantly outperforms LLM-based methods. Furthermore, SC-Energy also outperforms the set-level comparison & binary classifier architecture model. This is a particularly impressive result given that Table 2(a) shows only a marginal performance difference in set-consistency verification. SC-Energy learns fine-grained contrastive representations across multiple datasets, enabling it to capture the varying degrees of inconsistency within each set. This capability underlies its strong performance in accurately locating inconsistencies.

6 Ablation Study

6.1 Effect of Contrast Granularity

In Section 3.3.2, we introduced eight contrastive signals for training SC-Energy. To evaluate their impact, we conducted an ablation study with three training regimes, as illustrated in Figure 1: **Left Panel:** The model is trained solely with the basic contrast $L_E(S_C, S_I)$. This limited contrast approach yields lower classifying performance. **Middle Panel:** The model is trained with six contrasts ($L_E(S_C, S_I)$, $L_E(S_C, S_{CI})$, $L_E(S_C, S_{II})$, $L_E(S_{CC}, S_I)$, $L_E(S_{CC}, S_{CI})$, and $L_E(S_{CC}, S_{II})$) which compare sets with consistent labels against those with inconsistent labels, but do not capture variations within the inconsistent sets. **Right Panel:** In addition to the six contrasts of middle panel, it also incorporates the two inconsistency degree contrasts $L_E(S_{CI}, S_I)$ and $L_E(S_I, S_{II})$. The eight contrasts training regime enables the model to capture the order of inconsistencies. We posit three intuitive criteria for ordering inconsistency in sets: (i) sets with more inconsistent pairs should have higher energy scores; (ii) if a set contains many consistent pairs alongside a few inconsistent ones, its overall energy should be more moderate (mid-range) rather than extreme, reflecting the "dampening" effect of the consistent content; and (iii) even among sets that are entirely consistent, larger set (*i.e.*, those combining more statements) should receive higher energy scores, since adding more content can introduce more neutral relationships between statements. Figure 1 shows that the energy distributions learned under our eight-contrast training regime accord well with these criteria.

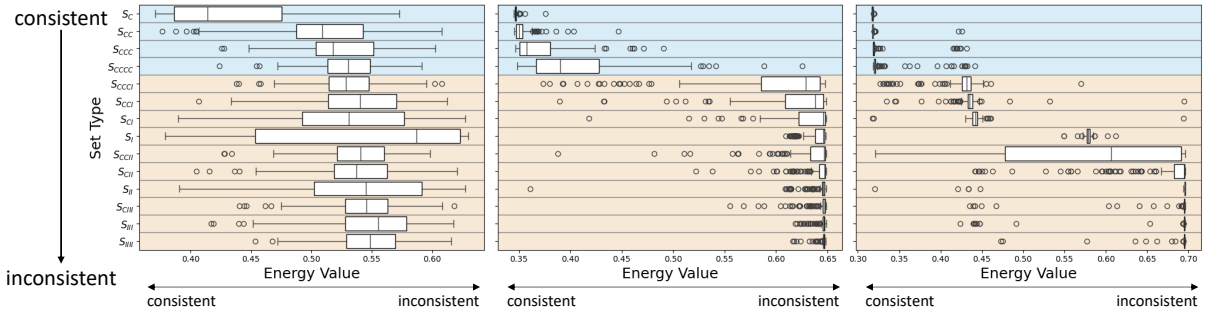

Figure 1: Box plots of energy values for various set types under different training regimes. **Left:** Training of SC-Energy just contrasting S_C vs. S_I. **Middle:** Training with six contrastive signals (contrasting all comparing sets with consistent vs. inconsistent labels), which does not capture the inconsistency degrees within inconsistent sets well. **Right:** Training with all eight contrastive signals, including the additional inconsistency degree contrasts, enables the model to distinguish sets more clearly. The set types are arranged from top to bottom in accordance with our inconsistency-ordering criteria, reflecting an increasing degree of inconsistency. Note that as additional fine-grained contrastive signals are incorporated, the resulting energy levels more faithfully reflect our intuitive ordering of inconsistency.

Source	Target (#Data)	Test Data Set-LConVQA	Test Data Set-SNLI
Set-LConVQA	Set-SNLI (100)	$0.938_{\pm 0.025}$	$0.666_{\pm 0.008}$
	Set-SNLI (200)	$0.942_{\pm 0.056}$	$0.727_{\pm 0.007}$
Set-SNLI	Set-LConVQA (100)	$0.872_{\pm 0.026}$	$0.979_{\pm 0.004}$
	Set-LConVQA (200)	$0.914_{\pm 0.043}$	$0.960_{\pm 0.002}$

Table 3: Macro-F1 scores, with five random seed runs, for transferring the SC-Energy toward different domain. The table demonstrates the model's ability to retain performance on its original dataset while effectively generalizing to new domains with small amount of fine-tuning data.

6.2 Fine-Tuning Efficiency

Table 3 shows the Macro-F1 scores of set-consistency verification task when fine-tuning the SC-Energy with a small amount of additional data from a different domain. Average and standard deviation for five different seed pairs are provided. The model retains strong performance on its original dataset while effectively adapting to new domains with minimal data. Refer to the appendix D.3 for experimental details.

6.3 Verifying Inconsistencies in LLM Outputs

Our SC-Energy can serve as a consistency evaluator that can be integrated with black-box LLMs to detect their inconsistent behaviors. While LLMs excel at various NLP tasks, they frequently produce outputs that are not logically coherent. For instance, an LLM might answer "yes" to "Does CO_2 increase the population of polar bears?' yet fail to provide a complementary response to "Does CO_2 decrease the population of polar bears?".

To expose such inconsistencies, (Asai and Hajishirzi, 2020) proposed a data augmentation technique for the WIQA dataset (Tandon et al., 2019) that substitutes keywords with antonyms (e.g., "more" with "less") to create question pairs with same/opposite expected answers. We evaluate the consistency of LLM on this augmented WIQA dataset to constrcut *WIQA set consistency*, a new set-consistency dataset based on LLM answer generation. In our construction, the LLM is queried independently for each related question to yield a set of question–prediction pairs whose logical coherence is then assessed in a rule-based manner. Our data collection with GPT-4o shows that it is 51.1% consistent.

In order to validate the SC-Energy's efficacy as a general consistency evaluator, we compare SC-Energy's accuracy on WIQA set consistency data along with the Self-Check mechanism where LLM evaluates the consistency of its own response set. As shown in Table 4, the consistency detection accuracy increases from 59.9% with the self-check to 68.7% when using SC-Energy. These results demonstrate that SC-Energy effectively identifies and flags inconsistent outputs, thereby functioning as a robust consistency evaluator when combined with existing LLMs. Detailed descriptions of the evaluation methodology, including prompt design and the self-check process, are provided in appendix D.5.

WIQA set consistency	Self-Check	SC-Energy (Ours)
Consistent (51.1%)	79.6	81.9
Inconsistent (48.9%)	39.2	54.8
Total Accuracy	59.9	**68.7**

Table 4: The table reports the accuracy of the set-consistency verification (%) on the WIQA set consistency dataset.

7 Conclusion

In this paper, we introduced the Set-Consistency Energy Network (SC-Energy), a novel approach for verifying logical inconsistencies across multiple statements.

Through extensive experiments on Set-LConVQA and Set-SNLI, we demonstrated that our energy-based framework significantly outperforms baseline methods, including LLM-based approaches and traditional classification models. The results show that our model excels in set-consistency verification and locating inconsistent statements within a set. Additionally, we explored its applicability as a consistency evaluator for verifying inconsistencies in LLM-generated outputs, showcasing its potential for real-world deployment.

8 Limitations and Potential Risks

Our approach processes entire sets in a single pass, which, while enabling holistic consistency assessment, poses challenges when dealing with longer sets that may exceed the maximum token limit of current language models. Additionally, our two datasets Set-LConVQA, Set-SNLI are generated using a combination of rule-based methods and LLM-based transformations. Incorporating more diverse and representative data generation criteria could enable further advancements in this task. Moreover, evaluating the generalization capabilities of our approach across different domains remains an important direction for future research.

Other potential risks include a sensitivity to noise and edge cases present in real-world data. Addressing these issues will be critical for scaling the proposed method to broader applications.

9 Acknowledgement

This work was supported in part by the National Research Foundation of Korea (NRF) grant (RS-2023-00280883, RS-2023-00222663); by the National Super computing Center with super computing resources including technical support (KSC-2023-CRE-0176, KSC-2024-CRE-0065); by the Korea Institute of Science and Technology Information (KISTI) in 2025 (No.(KISTI) K25L1M1C1), aimed at developing KONI (KISTI Open Neural Intelligence), a large language model specialized in science and technology; and by the Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (RS-2025-02263754); by the Korea Medical Device Development Fund grant funded by the Korea government (the Ministry of Science and ICT, the Ministry of Trade, Industry and Energy, the Ministry of Health & Welfare, the Ministry of Food and Drug Safety, 2025의약안003); partially supported by New Faculty Startup Fund from Seoul National University.

References

Akari Asai and Hannaneh Hajishirzi. 2020. Logic-guided data augmentation and regularization for consistent question answering. *arXiv preprint arXiv:2004.10157*.

David Belanger and Andrew McCallum. 2016. Structured prediction energy networks. In *International Conference on Machine Learning*, pages 983–992. PMLR.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

P Kingma Diederik. 2014. Adam: A method for stochastic optimization. *(No Title)*.

Alexander Fabbri, Chien-Sheng Wu, Wenhao Liu, and Caiming Xiong. 2022. QAFactEval: Improved QA-based factual consistency evaluation for summarization. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2587–2601, Seattle, United States. Association for Computational Linguistics.

Tobias Falke, Leonardo F. R. Ribeiro, Prasetya Ajie Utama, Ido Dagan, and Iryna Gurevych. 2019. Ranking generated summaries by correctness: An interesting but challenging application for natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2214–2220, Florence, Italy. Association for Computational Linguistics.

Philippe Laban, Tobias Schnabel, Paul N. Bennett, and Marti A. Hearst. 2022. SummaC: Re-visiting NLI-based models for inconsistency detection in summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177.

Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, Fujie Huang, et al. 2006. A tutorial on energy-based learning. *Predicting structured data*, 1(0).

Jay Yoon Lee, Dhruvesh Patel, Purujit Goyal, Wenlong Zhao, Zhiyang Xu, and Andrew McCallum. 2022. Structured energy network as a loss. *Advances in Neural Information Processing Systems*, 35:20862–20875.

Yinhan Liu. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 364.

Zheheng Luo, Qianqian Xie, and Sophia Ananiadou. 2023. Chatgpt as a factual inconsistency evaluator for text summarization. *arXiv preprint arXiv:2303.15621*.

Mutsumi Nakamura, Santosh Mashetty, Mihir Parmar, Neeraj Varshney, and Chitta Baral. 2023. LogicAttack: Adversarial attacks for evaluating logical consistency of natural language inference. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13322–13334, Singapore. Association for Computational Linguistics.

Arijit Ray, Karan Sikka, Ajay Divakaran, Stefan Lee, and Giedrius Burachas. 2019. Sunny and dark outside?! improving answer consistency in VQA through entailed question generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5860–5865, Hong Kong, China. Association for Computational Linguistics.

Niket Tandon, Bhavana Dalvi Mishra, Keisuke Sakaguchi, Antoine Bosselut, and Peter Clark. 2019. Wiqa: A dataset for" what if..." reasoning over procedural text. *arXiv preprint arXiv:1909.04739*.

Lifu Tu, Richard Yuanzhe Pang, and Kevin Gimpel. 2019. Improving joint training of inference networks and structured prediction energy networks. *arXiv preprint arXiv:1911.02891*.

Alex Wang, Kyunghyun Cho, and Mike Lewis. 2020. Asking and answering questions to evaluate the factual consistency of summaries. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5008–5020, Online. Association for Computational Linguistics.

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

Liyan Xu, Zhenlin Su, Mo Yu, Jin Xu, Jinho D Choi, Jie Zhou, and Fei Liu. 2024. Identifying factual inconsistency in summaries: Towards effective utilization of large language model. *arXiv preprint arXiv:2402.12821*.

Jiuding Yang, Hui Liu, Weidong Guo, Zhuwei Rao, Yu Xu, and Di Niu. 2024. Reassess summary factual inconsistency detection with large language model. In *Proceedings of the 1st Workshop on Towards Knowledgeable Language Models (KnowLLM 2024)*, pages 27–31, Bangkok, Thailand. Association for Computational Linguistics.

A Details of SC-Energy

This section covers the topic of detail information on the training procedure using the eight contrasts, the conversion of a set S into the input for SC-Energy, and the selection of the threshold.

A.1 Training Procedure Using the Eight Contrasts

In Section 3.3.2, we demonstrated that by appropriately merging S_C and S_I, we can extend the set and train SC-Energy using eight distinct contrast methods. During training, the entire training dataset is treated as if it contains all eight types of contrast pairs (e.g., (S_C, S_I), (S_C, S_{CI}), (S_C, S_{II}), etc.). Whenever a sample corresponding to a specific contrast type is drawn from the dataset, its loss $L_E(\cdot, \cdot)$ is computed based on that contrast.

A.2 Conversion of S into the Input for SC-Energy

For a set $S = \{s_1, s_2, \cdots, s_n\}$ composed of individual sentences, the input to SC-Energy is formed by concatenating all sentences s_i and prepending a CLS token at the beginning. For example, consider a set S from the Set-SNLI dataset defined as:

$$S = \{\text{“Either the train arrives at 8 AM or it arrives at 9 AM.”},\\ \text{“The train does not arrive at 8 AM.”},\\ \text{“The train does not arrive at 9 AM.”}\}$$

In the case of the RoBERTa-base model, the CLS token is represented as `<s>`, so the input for set S is:

```
<s> Either the train arrives at 8 AM or it arrives at 9 AM.
The train does not arrive at 8 AM. The train does not arrive at 9 AM.
```

In contrast, for a set $S = \{(x_i, y_i)\}_{i=1}^{n}$ composed of QA pairs, we concatenate each x_i and y_i using the delimiter *“The answer is”*. After concatenating all the QA pairs, we prepend a CLS token at the beginning. For example, consider a set S from the Set-LConvqa dataset defined as:

$$S = \{(\text{“what color is desk?”}, \text{“brown”}), (\text{“is desk brown?”}, \text{“yes”})\}$$

With RoBERTa-base, where the CLS token is `<s>`, the input for set S becomes:

```
<s> what color is desk? The answer is brown. is desk brown? The answer is yes.
```

When converting a set S into the input for SC-Energy, all elements of S are shuffled before being fed into the model.

A.3 Selection of Threshold

The dataset is split into training data, validation1 data, validation2 data, and test data. At the end of each training epoch, the threshold is updated using validation1 data. A set is classified as consistent if its energy value is below the threshold; otherwise, it is classified as inconsistent. The threshold is determined by maximizing macro classification accuracy across validation1 data consisting of up to two combined sets: S_C, and S_{CC} as consistent sets, while S_I, S_{CI}, and S_{II} are labeled as inconsistent. Finally, validation2 data is used for hyperparameter tuning.

B Set-LConVQA Dataset

The Set-LConVQA dataset is derived from the original LConVQA (Ray et al., 2019) dataset which is publicly available[2], where for each image, both consistent and inconsistent sets of QA pairs are generated. For example, a consistent set is represented as follows:

[2]https://arijitray1993.github.io/ConVQA/

```
"consistent": [
    [
        {"question": "what color is desk?", "answer": "brown"},
        {"question": "is desk brown?", "answer": "yes"},
        {"question": "is desk pink?", "answer": "no"}
    ]
]
```

In contrast, an inconsistent set is always composed of exactly two QA pairs, as in this example:

```
"inconsistent": [
    [
        {"question": "what color is desk?", "answer": "brown"},
        {"question": "is desk brown?", "answer": "no"}
    ],
    [
        {"question": "what color is desk?", "answer": "brown"},
        {"question": "is desk pink?", "answer": "yes"}
    ]
]
```

Our rule-based validation of the LConVQA dataset revealed two key observations:

1. For every consistent set derived from a given image, there exists at least one inconsistent set such that all questions that appear in that inconsistent set are also present in the corresponding consistent set.

2. In the inconsistent sets, one answer differs from the answer in the consistent set for the corresponding question.

In other words, an inconsistent set can be constructed from a consistent set by altering the answer of a single QA pair while keeping all the questions unchanged. Formally, given a consistent set

$$S_C = \{(q_1, a_1), (q_2, a_2), \ldots, (q_n, a_n)\},$$

an inconsistent set can be obtained as

$$S_I = \{(q_1, a_1), \ldots, (q_i, a_i^*), \ldots, (q_n, a_n)\},$$

where a_i^* differs from a_i. Notably, the generated inconsistent set S_I encompasses the inconsistent sets originally present in the LConVQA dataset. As defined in Section 3.1, a set S is deemed **inconsistent** if there exists a subset $\tilde{S} \subseteq S$ with $|\tilde{S}| \geq 2$ such that the statements in $\tilde{S}$ are logically contradictory. Hence, by our definition, S_I qualifies as an inconsistent set. On average, each set contains 3.43 QA pairs, with a maximum of six.

Because the element-wise verification strategy described in Section 5 is trained on sets of size 2, we can construct a dedicated element-wise training dataset from the generated Set-LConVQA dataset. Specifically, the inconsistent sets S_I in the original LConVQA dataset (which are of size 2) can be directly used for element-wise verification. Similarly, since every inconsistent set derived from a consistent set S_C involves altering only the answer while keeping all questions identical, the corresponding QA pairs can be extracted to form the element-wise consistent set for training with the element-wise strategy.

C Set-SNLI dataset

C.1 Dataset Creation

Inspired by previous work (Nakamura et al., 2023), we construct the Set-SNLI dataset by transforming existing pairwise SNLI datasets[3] according to predefined rules. The generation process consists of the following steps:

[3]SNLI dataset is publicly available, https://nlp.stanford.edu/projects/snli/. This dataset takes Creative Commons Attribution-ShareAlike 4.0 International License, allowing remix, transform, and build upon the material for any purpose.

1. Generating Negations: Using a large language model (LLM), we generate negated versions of the premises and hypotheses in the original dataset. The prompt used for negation generation is included in the appendix C.3.

2. Constructing Sentence Sets: One or two pairs of premises and hypotheses, along with their negations, are combined according to predefined rules.

The rules applied vary based on the label of the original premise-hypothesis relationship and the number of seed pairs used. Detailed rules are outlined in the appendix C.4. Some rules employ first-order logic inference to guarantee entailment between sentences, while others guarantee contradiction. Additional techniques involve leveraging sentence equivalence or combining previously generated sets to create larger sentence groups.

Sentences connected with conjunctions (AND) are always split so that each element in a set remains an atomic sentence. The definition of "atomic" may vary, but structures involving disjunctions (OR) or conditionals (if-then) are allowed. However, AND operations are disallowed to prevent arbitrary manipulation of set sizes by linking sentences with AND. Although the above dataset creation process can be applied to any single sentence based conventional NLI datasets, for our implementation, we derive our dataset from the test set of SNLI dataset.

C.2 Dataset Structure

Each data instance consists of the following: a set of two or more sentences, a label, i.e., either consistent or inconsistent, difficulty annotation, and set size. The difficulty annotation is categorized as either "medium" or "easy". The "easy" category is assigned only to contradictory sets that include a sentence and its direct negation, which we consider trivial to identify as contradictory. On average, each set contains 3.48 sentences, with a maximum of five.

C.3 Negation generation prompt

In order to create negated versions of premises and hypotheses, we instruct an LLM (in our case, gpt4o-mini) with the prompt provided in the Table 5

C.4 Dataset creation rules

In this section, we provide rules used to create a set-based NLI dataset from a conventional pair-wise SNLI dataset. For convenience, we divide the section in terms of the number and label of seed pair(s)(in other words, the original premise-hypothesis pair in SNLI dataset) we use to create each set. For tables 6, 7, 8, 9, the singleton rules are motivated by (Nakamura et al., 2023), and we generated additional rules from them (represented by the union mark $\cup$).

C.4.1 Rules for deriving sets from a single entailment seed pair

Please refer to Table 6.

C.4.2 Rules for deriving sets from a single contradiction seed pair

Please refer to Table 7.

C.4.3 Rules for deriving sets from a single neutral seed pair

Please refer to Table 8.

C.4.4 Rules for deriving sets from two entailment seed pairs

Please refer to Table 9.

C.4.5 Rules for Deriving Sets for Element-Wise Verification Strategy

Since the element-wise verification strategy (discussed in Section 5) is trained on sets of size 2, we construct a dedicated element-wise dataset using the generated Set-SNLI dataset.

Within a given set, a pair of propositions that exhibit a logically inconsistent relationship can be selected when the single entailment pair (as described in Table 6) satisfies one of the following conditions between p and h:

Instruction
Premise: An older man wearing a salon drape getting a haircut. Hypothesis: A man gets a haircut. Negation of Hypothesis: No man gets a haircut. Negation of Premise: No older man wearing a salon drape is getting a haircut. Premise: A man with glasses sitting at a restaurant staring at something that is not shown. Hypothesis: A man at a restaurant. Negation of Hypothesis: No man at a restaurant. Negation of Premise: No man with glasses sitting at a restaurant staring at something that is not shown. Premise: The school is having a special event in order to show the American culture on how other cultures are dealt with in parties. Hypothesis: A school is hosting an event. Negation of Hypothesis: No school is not hosting an event. Negation of Premise: The school is not having a special event to show the American culture or how other cultures are dealt with in parties. Premise: High fashion ladies wait outside a tram beside a crowd of people in the city. Hypothesis: Women are waiting by a tram. Negation of Hypothesis: No women are waiting by a tram. Negation of Premise: No high fashion ladies wait outside a tram beside a crowd of people in the city. Premise: People waiting to get on a train or just getting off. Hypothesis: There are people waiting on a train. Negation of Hypothesis: There are no people waiting on a train. Negation of Premise: No people are waiting to get on a train or just getting off. Premise: A couple play in the tide with their young son. Hypothesis: The family is outside. Negation of Hypothesis: The family is not outside. Negation of Premise: A couple does not play in the tide with their young son. { More examples omitted }
Take a closer look at these tricky cases; 'A' or 'An' needs to change to 'No' even if the sentence does not start with the subject, and also when two clauses exist in a single sentence. Premise: Under a blue sky with white clouds, a child reaches up to touch the propeller of a plane standing parked on a field of grass. Hypothesis: A child is reaching to touch the propeller of a plane. Negation of Hypothesis: Under a blue sky with white clouds, no child reaches up to touch the propeller of a plane standing parked on a field of grass. Negation of Premise: No child is reaching to touch the propeller of a plane Premise: A man in a green shirt is singing karaoke while a young woman with long brownish hair stands by and listens. Hypothesis: A man is singing a song. Negation of Hypothesis: No man is singing a song. Negation of Premise: No man in a green shirt is singing karaoke while no young woman with long browish hair stands by and listens.
When a sentence starts with subjects specified with 'the' or 'this', verb should be negated rather than the subject. Premise: A woman with a green headscarf blue shirt and a very big grin. Hypothesis: The woman is very happy. Negation of Hypothesis: The woman is not very happy. Negation of Premise: No woman with a green headscarf blue shirt and a very big grin. Premise: The man walks among the large trees. Hypothesis: The man walks among trees. Negation of Hypothesis: The man does not walk among trees. Negation of Premise: The man does not walk among the large trees.
Please note that given a compound sentence connecting two or more clauses with an 'and' or a comma, each of the clauses need to be negated, connected by an 'or'. Premise: Four people are near a body of water, two people walk on a sidewalk. Hypothesis: There are people outdoors. Negation of Hypothesis: There are no people outdoors Negation of Premise: No four people are near a body of water or no two people walk on a sidewalk.
Construct Negation of Hypothesis and Negation of Premise by following the examples above. Given the condtion that Premise entails Hypothesis, Negation of Premise needs to entail Negation of Hypothesis.

Table 5: Instruction used to create negation of premise and hypothesis of the original NLI dataset. The few shot examples in the first part of the instruction are taken from Nakamura et al., 2023.

1. p, $\neg p$
2. h, $\neg h$
3. p, $\neg h$
4. $p \vee h$, $\neg h$

For each single entailment seed pair (p, h), an inconsistent set S_I can be created by applying the above rules. Conversely, for constructing a consistent set S_C, any size-2 subset extracted from a consistent set qualifies as a consistent set, according to the definition provided in Section 3.1.

D Experiment

D.1 Set-Consistency Verification

In this section, we present further detailed experimental results related to set-consistency verification.

No.	Rule	Description	Set Size	Label	Difficulty
1	$\{p_1 \to h_1, \neg h_1 \to \neg p_1\}$	Transportation	2	Consistent	Medium
2	$\{p_1 \to h_1, \neg p_1 \vee h_1\}$	Material Implication	2	Consistent	Medium
3	$\{p_1 \to h_1, h_1\}$	Split Hypothesis of Rule 2 (1)	2	Consistent	Medium
4	$\{p_1 \to h_1, \neg p_1\}$	Split Hypothesis of Rule 2 (2)	2	Consistent	Medium
5	$\{p_1 \to h_1, p_1, h_1\}$	Modus Ponens	3	Consistent	Medium
6	$\{p_1 \to h_1, \neg h_1, \neg p_1\}$	Modus Tollens	3	Consistent	Medium
7	$\{p_1 \vee h_1, \neg h_1, p_1\}$	Disjunctive Syllogism (1)	3	Consistent	Medium
8	$\{p_1 \vee h_1, \neg p_1, h_1\}$	Disjunctive Syllogism (2)	3	Consistent	Medium
9	$\{p_1 \to h_1, \neg h_1 \to \neg p_1, \neg p_1 \vee h_1\}$	Rule 1 $\cup$ 2	3	Consistent	Medium
10	$\{p_1 \to h_1, \neg h_1 \to \neg p_1, h_1\}$	Rule 1 $\cup$ 3	3	Consistent	Medium
11	$\{p_1 \to h_1, \neg h_1 \to \neg p_1, \neg p_1\}$	Rule 1 $\cup$ 4	3	Consistent	Medium
12	$\{p_1 \to h_1, \neg p_1 \vee h_1, h_1\}$	Rule 2 $\cup$ 3	3	Consistent	Medium
13	$\{p_1 \to h_1, \neg p_1 \vee h_1, \neg p_1\}$	Rule 2 $\cup$ 4	3	Consistent	Medium
14	$\{p_1 \to h_1, \neg p_1, h_1\}$	Rule 3 $\cup$ 4	3	Consistent	Medium
15	$\{p_1 \to h_1, \neg h_1 \to \neg p_1, p_1, h_1\}$	Rule 1 $\cup$ 5	4	Consistent	Medium
16	$\{p_1 \to h_1, \neg h_1 \to \neg p_1, \neg p_1, \neg h_1\}$	Rule 1 $\cup$ 6	4	Consistent	Medium
17	$\{p_1 \to h_1, \neg p_1 \vee h_1, p_1, h_1\}$	Rule 2 $\cup$ 5	4	Consistent	Medium
18	$\{p_1 \to h_1, \neg p_1 \vee h_1, \neg p_1, \neg h_1\}$	Rule 2 $\cup$ 6	4	Consistent	Medium
19	$\{p_1 \to h_1, \neg p_1 \vee h_1, \neg h_1 \to \neg p_1, h_1\}$	Rule 1 $\cup$ 2 $\cup$ 3	4	Consistent	Medium
20	$\{p_1 \to h_1, \neg p_1 \vee h_1, \neg h_1 \to \neg p_1, \neg p_1\}$	Rule 1 $\cup$ 2 $\cup$ 4	4	Consistent	Medium
21	$\{p_1 \to h_1, \neg h_1 \to \neg p_1, \neg p_1, h_1\}$	Rule 1 $\cup$ 3 $\cup$ 4	4	Consistent	Medium
22	$\{p_1 \to h_1, \neg p_1 \vee h_1, \neg p_1, h_1\}$	Rule 2 $\cup$ 3 $\cup$ 4	4	Consistent	Medium
23	$\{p_1 \to h_1, \neg p_1 \vee h_1, \neg h_1 \to \neg p_1, p_1, h_1\}$	Rule 1 $\cup$ 2 $\cup$ 5	5	Consistent	Medium
24	$\{p_1 \to h_1, \neg p_1 \vee h_1, \neg h_1 \to \neg p_1, \neg p_1, \neg h_1\}$	Rule 1 $\cup$ 2 $\cup$ 6	5	Consistent	Medium
25	$\{p_1 \to h_1, \neg p_1 \vee h_1, \neg h_1 \to \neg p_1, \neg p_1, h_1\}$	Rule 1 $\cup$ 2 $\cup$ 3 $\cup$ 4	5	Consistent	Medium
26	$\{p_1, \neg h_1\}$	Negate Hypothesis (1)	2	Inconsistent	Medium
27	$\{p_1, \neg h_1, p_1 \to h_1\}$	Negate Hypothesis (2)	3	Inconsistent	Medium
28	$\{p_1 \vee h_1, \neg p_1, \neg h_1\}$	Negate Hypothesis of Rule 7	3	Inconsistent	Medium
29	$\{p_1 \vee h_1, p_1 \to h_1, \neg h_1\}$	Implicit Negate Hypothesis of Rule 7	3	Inconsistent	Medium
30	$\{p_1 \to h_1, \neg h_1, \neg p_1, p_1\}$	Rule 6 $\cup$ Rule 26	4	Inconsistent	Easy
31	$\{p_1 \to h_1, \neg h_1, \neg p_1, h_1\}$	Rule 6 $\cup$ Rule 3	4	Inconsistent	Easy
32	$\{p_1 \to h_1, p_1, h_1, \neg p_1\}$	Rule 5 $\cup$ Rule 4	4	Inconsistent	Easy
33	$\{p_1 \to h_1, p_1, h_1, \neg h_1\}$	Rule 5 $\cup$ Rule 26	4	Inconsistent	Easy
34	$\{p_1 \to h_1, \neg h_1 \to \neg p_1, p_1, \neg h_1\}$	Rule 1 $\cup$ Rule 26	4	Inconsistent	Medium
35	$\{p_1 \to h_1, \neg p_1 \vee h_1, p_1, \neg h_1\}$	Rule 2 $\cup$ Rule 26	4	Inconsistent	Medium
36	$\{p_1 \to h_1, \neg h_1, \neg p_1, p_1, h_1\}$	Rule 6 $\cup$ Rule 5	5	Inconsistent	Easy

Table 6: List of rules used to create sets from a single entailment seed pair. The seed pair is comprised of a premise and a hypothesis, i.e., $\{p_1, h_1\}$, that are in entailment relationship $p_1 \to h_2$.

No.	Rule	Description	Set Size	Label	Difficulty
1	$\{\neg p_1, h_1\}$	Negate Premise	2	Consistent	Medium
2	$\{p_1, \neg h_1\}$	Negate Hypothesis	2	Consistent	Medium
3	$\{p_1 \vee h_1, \neg h_1, p_1\}$	Disjunctive Syllogism (1)	3	Consistent	Medium
4	$\{p_1 \vee h_1, \neg p_1, h_1\}$	Disjunctive Syllogism (2)	3	Consistent	Medium
5	$\{p_1 \vee h_1, p_1, h_1\}$	Disjunction $\cup$ Seed Pair	3	Inconsistent	Medium
6	$\{p_1 \vee h_1, \neg p_1, \neg h_1\}$	Negate Hypothesis of Rule 3	3	Inconsistent	Medium

Table 7: List of rules used to create sets from a single contradiction seed pair. The seed pair consists of a premise and a hypothesis, i.e. $\{p_1, h_1\}$, that are in a contradiction relationship $p_1 \wedge h_2 \to \bot$.

No.	Rule	Description	Set Size	Label	Difficulty
1	$\{p_1 \vee h_1, \neg h_1, p_1\}$	Disjunctive Syllogism 1	3	Consistent	Medium
2	$\{p_1 \vee h_1, \neg p_1, h_1\}$	Disjunctive Syllogism 2	3	Consistent	Medium
3	$\{p_1 \vee h_1, \neg p_1, \neg h_1\}$	Negate Hypothesis of Rule 1	3	Inconsistent	Medium

Table 8: List of rules used to create sets from a single neutral seed pair. The seed pair consists of a premise and a hypothesis, i.e. $\{p_1, h_1\}$, that are in neutral relationship, i.e., $|p_1 \wedge h_1| > 0$ and $p_1 \not\to h_1$ and $h_1 \not\to p_1$.

No.	Rule	Description	Set Size	Label	Difficulty
1	$\{p_1 \to h_1, p_2 \to h_2, p_1 \vee p_2, h_1 \vee h_2\}$	Constructive Dilemma	4	Consistent	Medium
2	$\{p_1 \to h_1, p_2 \to h_2, \neg h_1 \vee \neg h_2, \neg p_1 \vee \neg p_2\}$	Destructive Dilemma	4	Consistent	Medium
3	$\{p_1 \to h_1, p_2 \to h_2, p_1 \vee \neg h_2, h_1 \vee \neg p_2\}$	Bidirectional Dilemma	4	Consistent	Medium
4	$\{p_1 \to h_1, p_2 \to h_2, p_1 \vee p_2, \neg h_1, \neg h_2\}$	Negate Hypothesis of Constructive Dilemma	5	Inconsistent	Medium
5	$\{p_1 \to h_1, p_2 \to h_2, \neg h_1 \vee \neg h_2, p_1, p_2\}$	Negate Hypothesis of Destructive Dilemma	5	Inconsistent	Medium
6	$\{p_1 \to h_1, p_2 \to h_2, p_1 \vee \neg h_2, \neg h_1, p_2\}$	Negate Hypothesis of Bidirectional Dilemma	5	Inconsistent	Medium

Table 9: List of rules used to create sets from two entailment seed pairs, $\{p_1, h_1\}$ and $\{p_2, h_2\}$. Each seed pair is in an entailment relationship, i.e. $p_1 \to h_1$ and $p_2 \to h_2$.

Original Premise & Hypothesis
p: A couple walk hand in hand down a street. h: A couple is walking together. Original label: entailment
Resulting Sets
$\Rightarrow$ Applied rule: $\{p \rightarrow h, \neg h, \neg p\}$ (Modus Tollens) $\Rightarrow$ Set: {"If a couple walk hand in hand down a street, then a couple is walking together.", "No couple is walking together.", "No couple walks hand in hand down a street."} $\Rightarrow$ Label: consistent
$\Rightarrow$ Applied rule: $\{p_1 \vee h_1, p_1 \rightarrow h_1, \neg h_1\}$ (Implicit Negate Hypothesis of Disjunctive Syllogism) $\Rightarrow$ Set: {"If a couple walk hand in hand down a street, then a couple is walking together.", "No couple is walking together.", "Either a couple walk hand in hand down a street, or a couple is walking together."} $\Rightarrow$ Label: inconsistent

Table 10: Examples of generating Set-NLI data from a conventional SNLI dataset. The original premises and hypotheses are sampled from the SNLI dataset. Note that Set-SNLI data is derived not only from premise-hypothesis pairs with an entailment relationship but also from those labeled as contradiction or neutral. We create both consistent and inconsistent sets from the seed pairs.

D.1.1 Experiment Setup Detail

In this section, we discuss the experimental setups employed for each model architecture and verification strategy.

LLM Model Architecture For the LLM-based model architecture, we employ few-shot with/without chain-of-thought (CoT) based prompting to request a consistency classification for a given set. For example, in the *set-level verification strategy*, the prompt for few-shot CoT prompting is formulated as follows:

Prompt:

> Tell me whether the following question-answer pairs are consistent or inconsistent.
>
> {Few-shot (CoT) examples (5-shot)}
>
> {Problem}
>
> Please think step by step: first, clearly articulate your thought process; then, provide your final consistency judgment by choosing either 'consistent' or 'inconsistent' after the 'Consistency:' mark.

For the *element-wise verification strategy*, the form of prompt and the number of few-shot CoT examples remain identical. However, there are two key differences:

1. When evaluating a set of size N, every possible pair of elements (i.e., $\frac{N(N-1)}{2}$ pairs) is compared for consistency. If even one pair is determined to be inconsistent, the entire set is classified as inconsistent.

2. The few-shot examples are constructed exclusively from sets consisting of two elements.

Binary Classification Model Architecture In the *set-level verification strategy*, the model's input is the entire set (for details on how to convert a set into the model's input, please refer to Appendix A.2; although this model is not SC-Energy, the method for converting the set into the input is the same). The

model outputs a 2-dimensional vector, where each component represents the score for either "consistent" or "inconsistent."

In the *element-wise verification strategy*, the process is similar; however, when evaluating a set of size N, every possible pair of elements (i.e., $\frac{N(N-1)}{2}$ pairs) is compared for consistency. This amounts to repeatedly assessing sets of size 2. If even one pair is determined to be inconsistent, the entire set is classified as inconsistent.

Typically, when making predictions from a 2-dimensional vector, it is common to select the class corresponding to the higher score. However, in our experiment setting, a threshold is also be learned for the binary classification model. The output scores for "consistent" and "inconsistent" are passed through a softmax function, and a threshold is learned on the "inconsistent" class score in the same manner as described in Appendix A.3. This learned threshold is then used during inference.

For the binary classification model architecture, training is conducted using a cross-entropy loss function. Under the set-level verification strategy, we treat S_C and S_{CC} as consistent label data and S_I, S_{CI}, and S_{II} as inconsistent label data. For the element-wise verification strategy, we used the training data described in Appendices B and C.

Energy-Based Model Architecture For the energy-based model architecture using the *set-level verification strategy*, the model is implemented as SC-Energy. Details regarding the training and inference procedures, as well as threshold selection, can be found in Appendix A. In the *element-wise verification strategy*, the procedure is analogous to that of the binary classification model architecture: for a set of size N, pairs of elements are extracted to form sets of size 2, and the consistency of each pair is evaluated. If even one pair is determined to be inconsistent, the entire set is classified as inconsistent.

We used Adam optimizer (Diederik, 2014), RTX A6000 GPU[4], and RoBERTa-base (Liu, 2019) model for all of trainings (for both binary classification and energy-based model architectures). We used grid search to find learning rate, one of $1, 1e-1, \cdots, 1e-7$. Learning rate of $1e-6$ is used for the loss function of binary classifier models, and learning rate of $1e-5$ is used for the loss function of energy-based models.

For the energy-based model architecture, training is conducted using a margin-based loss function. The training procedure of set-level verification strategy is described in 3.3.2. For the element-wise verification strategy, we used the training data described in Appendices B and C.

D.2 Difficulty Variation by Set Type

We observe that the dataset contains a meaningful internal structure: some set types are more difficult than others, despite high overall average scores. Tables 11 and 12 provide accuracy breakdowns by all 14 set types for three representative models on Set-LConVQA and Set-SNLI, respectively.

	S_C	S_I	S_{CC}	S_{CI}	S_{II}	S_{CCC}	S_{CCI}	S_{CII}	S_{III}	S_{CCCC}	S_{CCCI}	S_{CCII}	S_{CIII}	S_{IIII}
GPT-4o	0.895	0.995	0.875	0.930	1.000	0.865	0.915	0.995	1.000	0.885	0.835	0.975	1.000	1.000
o3-mini	0.995	0.950	0.995	0.965	1.000	0.985	0.925	0.990	1.000	0.970	0.930	1.000	1.000	1.000
SCV-Energy	1.000	1.000	1.000	0.994	0.998	0.994	0.979	1.000	1.000	0.966	0.931	0.993	1.000	1.000

Table 11: Accuracy by set type on Set-LConVQA across 14 set categories and three models.

	S_C	S_I	S_{CC}	S_{CI}	S_{II}	S_{CCC}	S_{CCI}	S_{CII}	S_{III}	S_{CCCC}	S_{CCCI}	S_{CCII}	S_{CIII}	S_{IIII}
GPT-4o	0.740	0.855	0.465	0.865	0.965	0.355	0.840	0.945	0.970	0.290	0.875	0.935	0.975	0.990
o3-mini	0.985	0.820	0.970	0.835	0.985	0.955	0.825	0.970	1.000	0.920	0.825	0.950	0.995	0.995
SCV-Energy	0.992	0.995	0.986	0.986	1.000	0.964	0.973	0.999	1.000	0.957	0.943	0.997	1.000	1.000

Table 12: Accuracy by set type on Set-SNLI across 14 set categories and three models.

These results indicate that sets composed entirely of consistent statements (*e.g.,* S_{CCCC}) or those with only one inconsistent element (*e.g.,* S_{CCCI}) present significant challenges.

[4]https://www.nvidia.com/content/dam/en-zz/Solutions/design-visualization/quadro-product-literature/proviz-print-nvidia-rtx-a6000-datasheet-us-nvidia-1454980-r9-web%20(1).pdf

D.2.1 Maximum Tolerance Rate for Element-wise Verification strategy

In a pairwise comparison approach, if any one-to-one pair comparison detects an inconsistency, the entire document is theoretically classified as inconsistent. This raises the question: to what extent should a certain level of inconsistency be tolerated (even if this is not the ideal approach) to achieve optimal performance in set-consistency verification?

To investigate this, we define the *Maximum Tolerance Rate* (MTR). For a set S with $|S| = N$, there are $\frac{N(N-1)}{2}$ possible pairwise comparisons. The MTR, denoted by p, represents the maximum allowable proportion of inconsistent pairs such that the set S is still classified as consistent. Formally, if

$$\frac{\text{number of predicted one-to-one pairs as inconsistent}}{\text{total number of pairs } (= \frac{N(N-1)}{2})} \leq p,$$

then S is deemed consistent; otherwise, it is classified as inconsistent. Note that when MTR $= 0$, it is the most appropriate decision rule (i.e., if even one one-to-one pair is predicted to be inconsistent, the entire set S is classified as inconsistent). As the value of MTR increases, the likelihood of classifying set S as consistent also increases.

Figure 2 shows the Macro-F1 score for the element-wise verification strategies across various MTR values and set sizes. **Left** figures represents the Macro-F1 score for Set-LConVQA dataset, and **Right** figures represents the Macro-F1 score for Set-SNLI dataset. Also, figures on the **Top** represents the Macro-F1 score for binary classifier models, and figures on the **Bottom** represents the Macro-F1 score for energy-based models. As the figure illustrates, there exists a point approximately around $20\% \sim 40\%$ where the Macro-F1 score peaks as the MTR is varied.

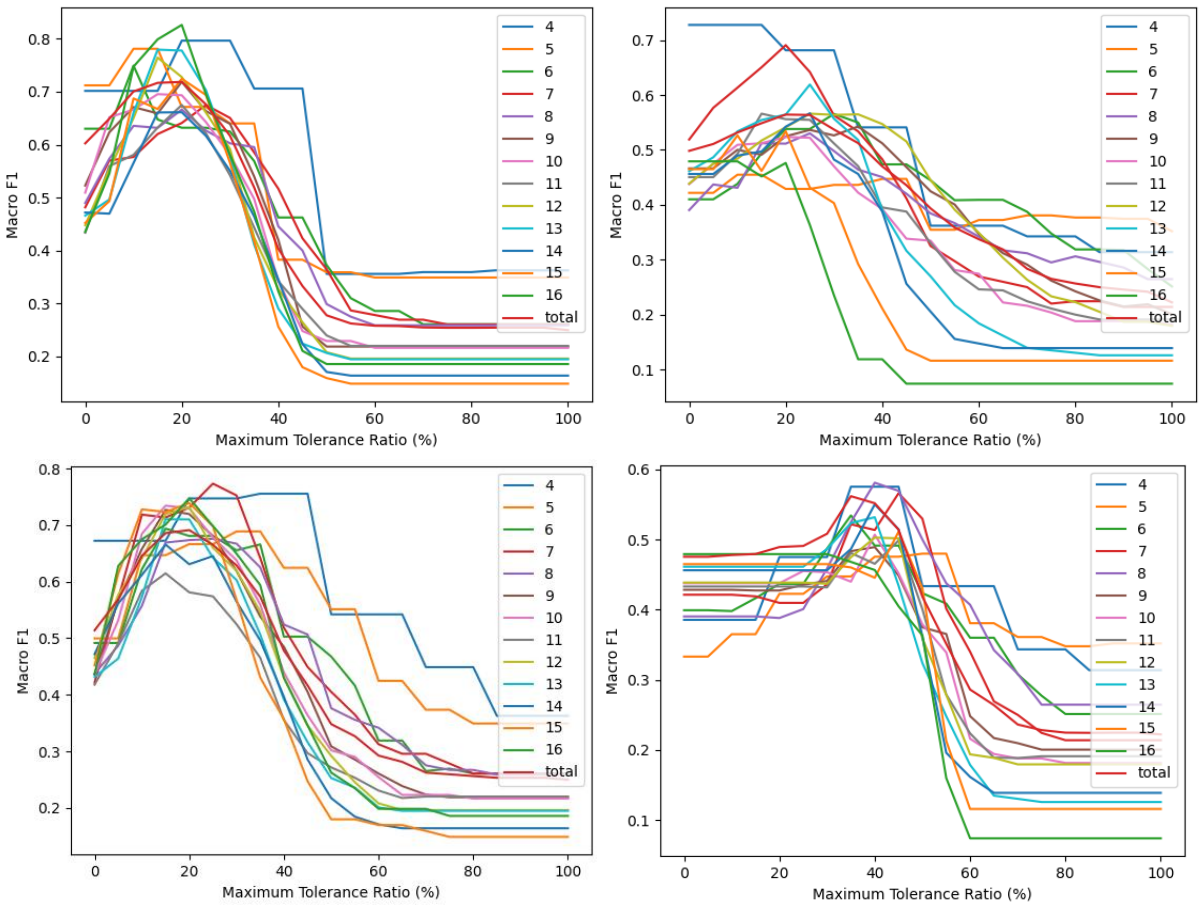

Figure 2: Figure shows the Macro-F1 score of set-consistency verification task across various MTR values and set sizes. **Left** figures represents the Macro-F1 score for Set-LConVQA dataset, and **Right** figures represents the Macro-F1 score for Set-SNLI dataset. Also, figures on the **Top** represents the Macro-F1 score for binary classifier models, and figures on the **Bottom** represents the Macro-F1 score for energy-based models.

D.3 Fine-Tuning Efficiency

This section details our fine-tuning experiments across different domain datasets. Specifically, when fine-tuning the SC-Energy model (originally trained on one dataset) using N samples from a different domain, we randomly select N samples from the original dataset and combine them with N samples from the target domain for each training epoch, resulting in a total of $2N$ training samples per epoch. An $L2$ regularizer with a weight of 0.00001 was applied during fine-tuning.

D.4 Locating Inconsistent Statements

In this section, we discuss the experimental setup details and the experimental results pertaining to the Locate task.

D.4.1 Experiment Setup Detail

In this section, we describe how the Locate task is performed for different model architectures.

LLM-based Model Architecture For the LLM-based model architecture, we use chain-of-thought (CoT) prompting. The prompt is as follows:

Prompt:

> Find the question-answer pairs among the following that are logically inconsistent with the rest. Specifically, identify the minimal collection of inconsistent pairs such that the remaining pairs are logically consistent with one another. If there are no inconsistent pairs, return nothing.
>
> {Few-shot CoT examples (5-shot)}
>
> {Problem}
>
> Think step by step, and respond with your answer containing the numbers of the inconsistent pairs after the '[Inconsistent pairs]' mark.

Binary Classifier / Energy-Based Model Architecture The binary classifier and energy-based model architectures can determine the overall inconsistency of a given set S, but they do not directly locate which specific QA pairs are responsible for the inconsistency. Therefore, we perform the set-consistency verification repeatedly to identify the inconsistent QA pairs within S. In the energy-based model, lower output scores indicate that a set is more consistent. Similarly, in the binary classifier, the score for the "inconsistent" label (obtained after a softmax) is used such that lower scores indicate higher consistency.

Below, we describe the procedure for performing the Locate task using a pseudocode algorithm.

Algorithm 1 Locate Task for Binary Classifier / Energy-Based Models

Input: Set $S = \{s_1, s_2, \ldots, s_N\}$ of QA pairs.
while True **do**
  Compute the set-consistency verification result for S.
  if S is classified as **consistent then**
    Return: Terminate — no inconsistent pair detected.
  else if S is classified as **inconsistent and** $|S| = 2$ **then**
    Return: Terminate — set too small to isolate an inconsistent pair.
  else
    for each $i = 1$ to N **do**
      Generate $S_i = S \setminus \{s_i\}$.
      Compute the energy value $E_\theta(S_i)$ (or the corresponding inconsistent score).
    end for
    Let s_j be the element whose removal yields the lowest energy value, i.e., $j = \arg\min_i E_\theta(S_i)$.
    Output: Identify s_j as the inconsistent QA pair.
    Update $S \leftarrow S_j$.
  end if
end while

In summary, for the binary classifier and energy-based model architectures, we repeatedly remove the QA pair whose exclusion minimizes the energy value (or inconsistent score) until the remaining set is classified as consistent.

D.5 Detecting Inconsistencies in LLM Outputs

In this section, we describe the data construction and prompt design procedures discussed in section 6.3.

D.5.1 Data Construction

Data Collection: Following the data augmentation approach of (Asai and Hajishirzi, 2020), we augment the WIQA dataset (Tandon et al., 2019). An example of an augmented data instance is shown below:

```
{
  "paragraph": [
    "Water from oceans, lakes, rivers, swamps, and plants turns into water vapor",
    "Water vapor forms droplets in clouds",
    "Water droplets in clouds become rain or snow and fall",
    "Some water goes into the ground",
    "Some water flows down streams into rivers and oceans",
    ""
  ],
  "choices": [
    {"label": "A", "text": "more"},
    {"label": "B", "text": "less"},
    {"label": "C", "text": "no effect"}
  ],
  "qa_pairs": [
    {
      "question": "suppose there is less water on the ground happens,  how will it
      affect less rain will fall.",
      "answer_label": "more",
      "answer_label_as_choice": "A"
    },
    {
      "question": "suppose there is more water on the ground happens, how will it
        affect less rain will fall.",
      "answer_label": "less",
      "answer_label_as_choice": "B"
    },
    {
      "question": "suppose more tadpoles develop in eggs happens, how will it affect
      less rain will fall.",
      "answer_label": "no_effect",
      "answer_label_as_choice": "C"
    },

    { ... additional QA pairs ... }
  ]
}
```

Data Cleaning: Our primary objective is to verify whether an LLM provides consistent answers when performing question-answering independently on related questions. To this end, we first group related questions by:

- Grouping questions that share the same paragraph.

- Selecting questions whose lists of words differ by at most two words.

For example, a group of related QA pairs might include:

```
{
  "question": "suppose there is more water on the ground happens, howwill it
affect a more intense water cycle.",
  "answer_label": "more",
  "answer_label_as_choice": "A"
},
{
  "question": "suppose there is less water on the ground happens, how will it
  affect a more intense water cycle.",
  "answer_label": "less",
  "answer_label_as_choice": "B"
},
{
  "question": "suppose there is more water on the ground happens, how will it
  affect a less intense water cycle.",
  "answer_label": "less",
  "answer_label_as_choice": "B"
},
{
  "question": "suppose there is less water on the ground happens, how will it
  affect a less intense water cycle.",
  "answer_label": "more",
  "answer_label_as_choice": "A"
}
```

If an LLM answers all these questions correctly, the corresponding QA pairs are consistent. However, consistency does not require perfect accuracy; in our augmented WIQA dataset, the answers follow one of three formats: `more`, `less`, or `no effect`. To facilitate automated consistency measurement, we discard QA pairs with the `no effect` answer and retain only those with `more` or `less`. In this setup, if an LLM answers all questions correctly or incorrectly, it is considered to have responded consistently. Only groups with at least two QA pairs are used in our experiments.

D.5.2 Prompt Design

Prompt for Question-Answering: The WIQA dataset provides a paragraph as context alongside each question. We use the following prompt to obtain the LLM's prediction for each question:

> **Prompt:** Given the following paragraphs, please select the correct answer for the given question. Return only the answer.
> **Paragraphs:** {A collection of sentences}
> **Choices:** more, less
> **Question:** {question}
>
> Specifically, please carefully read the question. For example, assume the question is "suppose less water in the environment happens, how will it affect a less intense water cycle." If you believe that a decrease in water leads to a less intense water cycle, you should answer `more` because less water results in a "less" intense water cycle.

Self-Check Mechanism: The self-check mechanism leverages the LLM's own architecture to perform set-level verification. In this process, the LLM is prompted by a zero-shot chain-of-thought (CoT) approach to verify the consistency of the question–prediction pairs it generated.