

Visual-Textual Entailment with Quantities Using Model Checking and Knowledge Injection

Nobuyuki Iokawa, Hitomi Yanaka

The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan
{iokawa, hyanaka}@is.s.u-tokyo.ac.jp

Abstract

In recent years, there has been great interest in multimodal inference. We concentrate on visual-textual entailment (VTE), a critical task in multimodal inference. VTE is the task of determining entailment relations between an image and a sentence. Several deep learning-based approaches have been proposed for VTE, but current approaches struggle with accurately handling quantities. On the other hand, one promising approach, one based on logical inference that can successfully deal with large quantities, has also been proposed. However, that approach uses automated theorem provers, increasing the computational cost for problems involving many entities. In addition, that approach cannot deal well with lexical differences between the semantic representations of images and sentences. In this paper, we present a logic-based VTE system that overcomes these drawbacks, using model checking for inference to increase efficiency and knowledge injection to perform more robust inference. We create a VTE dataset containing quantities and negation to assess how well VTE systems understand such phenomena. Using this dataset, we demonstrate that our system solves VTE tasks with quantities and negation more robustly than previous approaches.

Keywords: Visual-textual Entailment, Multimodal Inference, Numerical Understanding

1. Introduction

Multimodal inference is a challenging task that requires a unified understanding of different information types. In particular, inference between visual and textual data is a fundamental task that has been actively studied in recent years and has applications including image retrieval systems and robot interactions.

Visual-textual entailment (VTE; Xie et al., 2019; Suzuki et al., 2019) is one of the most critical tasks for multimodal inference. In a VTE task, an image and a natural language sentence are provided as a premise and a hypothesis, respectively. The task is to determine the entailment relation between the image and the sentence. The answer is *entailment* if the image entails the sentence, and *non-entailment* otherwise¹. Table 1 shows an example VTE task. In this example, the answer is *entailment* because the hypothesis sentence correctly describes the situation in the premise image. As in this example, determining entailment relations between an image and a sentence requires an accurate understanding of objects and their relationships in the image, as well as linguistic phenomena in the sentence, such as quantities and negation. Visual question answering (VQA; Antol et al., 2015) is another reasoning task between images and sentences. In a VQA task, the goal is to provide a

¹VTE can be regarded as a three-class classification task (*entailment*, *contradiction*, or *neutral*), but in this study, we consider a binary classification task by grouping *contradiction* and *neutral* into *non-entailment*.


Premise	
Hypothesis	Three plates are on the table.
Answer	<i>entailment</i>

Table 1: Example VTE task. The premise image was obtained from the Visual Genome (Krishna et al., 2017) dataset.

natural language answer when presented with an image and a corresponding natural language question about that image. Addressing VQA requires diverse skills, involving not only an understanding of the meaning conveyed in images and text but also the capability to formulate responses in alignment with the question format. In contrast, VTE is a straightforward classification task that predicts entailment relations. This task provides a more direct means of assessing whether the model accurately comprehends the meanings of images and sentences.

Various deep learning-based approaches have been developed to perform multimodal inference between images and sentences (Lu et al., 2019; Hu and Singh, 2021; Kim et al., 2021; Li et al., 2021; Singh et al., 2022). In these approaches, images and sentences are encoded as embed-

ding vectors, and neural network models trained with large datasets have achieved remarkable performance on multimodal inference tasks. However, recent analyses have shown that such deep learning-based approaches have difficulty in accurately handling quantities (Chattopadhyay et al., 2017; Zhang et al., 2018; Trott et al., 2018; Acharya et al., 2019; Parcalabescu et al., 2021).

Approaches based on logical inference have also been proposed for multimodal inference. Suzuki et al. (2019) proposed a logic-based system for solving VTE tasks. The system first generates logical meaning representations of a premise image and a hypothesis sentence, then uses an automated theorem prover to prove the entailment relation between them. This approach successfully solves VTE tasks with sentences featuring quantities and other semantically complex phenomena. However, logical representations of images are usually complex, and the computational cost of proving these representations is high, especially for problems involving many entities.

In this paper, we propose a logic-based VTE system with model checking. We use model checking to speed up inference, focusing on VTE tasks involving quantities and negation. Our system represents a premise image as a first-order logic (FOL) structure (called *an FOL model*) and a hypothesis sentence as a logical formula. The entailment relation is then determined by performing model checking between them. One crucial issue for logic-based methods is differences in vocabulary between a sentence and an FOL model. Another problem is that an FOL model generated from a premise image does not fully represent the information in the original image. To address these issues, we apply several types of knowledge injection and perform more robust inference.

We construct a dataset for VTE that includes quantities and negation to evaluate whether systems understand those phenomena. Using this dataset, we show that our proposed system solves VTE tasks with quantities and negations more robustly than the previous approaches (Suzuki et al., 2019; Kim et al., 2021; Singh et al., 2022).

Our system and dataset will be publicly available at <https://github.com/ynk1ab/LoVTEQ>.

2. Related Work

2.1. Deep Learning-based Systems

There has been remarkable progress in deep learning techniques in recent years, and approaches using deep learning are being actively studied in many areas of natural language processing. In particular, Transformer (Vaswani et al., 2017)-based large language models such as GPT (Rad-

ford et al., 2018), BERT (Devlin et al., 2019), and RoBERTa (Liu et al., 2019), have achieved remarkable performance on various natural language processing tasks.

One deep learning-based approach for VTE is Vision-and-Language Transformers (ViLT; Kim et al., 2021), which learn joint representations of the input image and text through the interaction of their features in the neural network. ViLT converts images and sentences into vector representations and predicts entailment relations using a neural network model trained on a large dataset. The Foundational Language And Vision Alignment (FLAVA; Singh et al., 2022) model is another vision-and-language model pretrained on both unimodal data like images and text and multimodal data like image-text pairs. FLAVA can solve image, text, and multimodal tasks through fine-tuning specific to the task at hand. Deep learning-based approaches such as ViLT and FLAVA are fast and robust to input noise. However, Parcalabescu et al. (2021) showed that deep learning-based vision-and-language models have difficulty counting the number of objects in an image.

2.2. Logic-based Systems

Suzuki et al. (2019) proposed a VTE system using logical inference. That system maps a premise image into a logical formula M and a hypothesis sentence into a logical formula T , then uses an automated theorem prover to determine whether $M \vdash T$ (M entails T). If so, the system determines *entailment*, and *non-entailment* otherwise. This system performs accurate inference for VTE tasks involving linguistic phenomena such as quantities, but reasoning about problems involving many objects takes time because it uses an automated theorem prover. In addition, M and T must share a consistent vocabulary, because when the same meaning is paraphrased differently, the system cannot recognize their identity.

The inference of entailment relations between tables and sentences has also been studied. Kurosawa and Yanaka (2022) proposed a logical inference system that focuses on understanding quantities between tables and sentences. This system maps a premise table into an FOL model and a hypothesis sentence into a logical formula, then determines the entailment relation by model checking between the model and the logical formula. Kurosawa and Yanaka (2022) applies model checking instead of theorem proving to accelerate the inference. Inspired by that approach, our proposed method uses model checking.

2.3. Scene Graph

The VTE system by Suzuki et al. (2019) uses scene graphs of images when mapping a premise image to a logical expression. A scene graph (Johnson et al., 2015) is a graph that represents information about an image. Each node of a scene graph represents either an object in the image, an object attribute, or a relationship between two objects. Scene graphs can capture detailed information about images and are applied to tasks such as image retrieval (Johnson et al., 2015), visual reasoning (Shi et al., 2019), and image generation (Johnson et al., 2018).

Previous studies (Yang et al., 2018; Zellers et al., 2018) proposed neural network models to generate scene graphs by detecting objects and their relationships in images. Large-scale scene graph datasets have also been constructed, including Visual Genome (VG; Krishna et al., 2017) and GQA (Hudson and Manning, 2019). The VG dataset contains manually annotated scene graphs for more than 100,000 images collected from Microsoft COCO (MS-COCO; Lin et al., 2014) and YFCC100M (Thomee et al., 2016). GQA is another scene graph dataset, constructed from VG by removing duplicated objects and unnatural nodes from VG scene graphs. The system by Suzuki et al. (2019) uses VG scene graphs as structured representations for premise images.

3. Method

3.1. Overview

Figure 1 shows an overview of our system for a VTE task. We first generate an FOL model representing a premise image generated from a scene graph of the image. We then map a hypothesis sentence to a logical formula via syntactic and semantic parsing. Finally, we apply model checking to determine whether the logical formula of the hypothesis sentence is valid under the model of the premise image. We predict *entailment* if the system outputs true, *non-entailment* otherwise. During model checking, we incorporate additional knowledge through knowledge injection.

We explain the details of each step in the following subsections.

3.2. Meaning Representation for Images

The following describes how we construct meaning representations for premise images. We first obtain a scene graph of a premise image, then construct an FOL model from that graph.

3.2.1. Scene Graph

We assume that a scene graph of the premise image is available in an existing dataset. There are studies about automatic scene graph generation (Yang et al., 2018; Zellers et al., 2018), and we will investigate using these approaches in future work.

Premise images in the dataset we used are from the VG (Krishna et al., 2017) dataset, and we obtained scene graphs of the images from the VG and GQA (Hudson and Manning, 2019) datasets. VG scene graphs contain many overlapping objects, making it difficult to accurately determine the number of objects in an image. By contrast, GQA scene graphs are created by normalizing VG graphs. Therefore, GQA graphs describe information about objects more correctly but lack information about attributes and relationships. To obtain scene graphs that best represent the image information, we create the following two graphs in addition to the graphs included in those datasets.

VG+GQA We use the following procedure to construct graphs by merging VG and GQA scene graphs for the same image. First, we add the VG graph objects to the GQA graph in order. If the added object is already in the graph, we merge the two objects, eliminating any VG graph overlaps. We determine that two objects refer to the same object if the intersection over union of their bounding boxes exceeds a threshold value, which we manually set to 0.70 based on prior experiments. We then add the attributes and relationships of the VG graph to obtain the final scene graph, which we call **VG+GQA**.

VG+GQA+OD Some image objects may not be annotated in the VG+GQA graph. In such cases, we use a pretrained object detection model (Faster R-CNN + Inception ResNet V2; Ren et al., 2015; Szegedy et al., 2017) to identify objects not in the VG+GQA graph. We then create a new graph by adding the detected objects to the VG+GQA graph, determining whether the detected objects differ from objects in the VG+GQA graph by the same method used when merging the graphs. We call the created graph **VG+GQA+OD** (where OD stands for "object detection").

3.2.2. FOL Model

We represent the premise image as an FOL model to perform model checking. The previous approach by Suzuki et al. (2019) translates a premise image to a formula for theorem proving, but we construct a model for model checking. An FOL model is defined in first-order predicate logic, denoted as a pair (D, V) , where D is a domain (a set of entities),

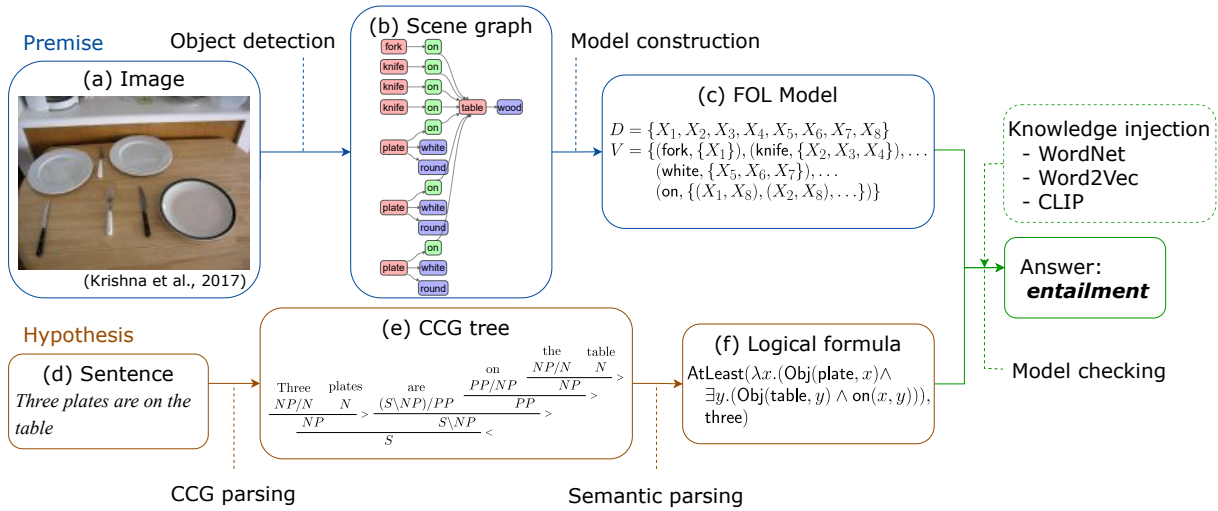


Figure 1: Overview of the proposed system. Our system first constructs an FOL model from the scene graph of the premise image and maps the hypothesis sentence to a logical formula. Then the system proves the entailment relation between them by model checking. The predicted label in this example is *entailment*.

and V is a valuation (a set of predicates, or mappings from entities to truth values). Each predicate in V is represented as a pair of a symbol and a set of (tuples of) entities that satisfy the predicate. Our approach leverages the scene graph of the image to construct an FOL model as follows:

- Extract objects from the scene graph of the image and define entities X_i corresponding to each object.
- Define object names and attributes as unary predicates and relationships between two objects as binary predicates.

Figure 2 shows the FOL model constructed from the premise image in Figure 1.

3.3. Meaning Representation for Sentences

In this subsection, we introduce meaning representations for hypothesis sentences. We first parse a hypothesis sentence based on Combinatory Categorical Grammar (CCG; Steedman, 2000) to obtain a CCG tree, then derive a logical formula as the meaning representation of the sentence according to the CCG tree.

3.3.1. CCG Parsing

We preprocess the hypothesis sentence in two steps. First, we identify compound nouns in the sentence by dependency parsing with spaCy² and combine each of them into a single token. Next, we use spaCy to apply part-of-speech (POS) tagging.

²<https://github.com/explosion/spaCy>

POS tags are used in the semantic parsing step. We then perform CCG parsing on the sentence and derive CCG parse trees. We use the off-the-shelf CCG parser depccg (Yoshikawa et al., 2017) to perform CCG parsing, deriving three trees with the highest probabilities for each sentence.

If the sentence has a there-construction, we filter the obtained CCG trees by their structure. The original semantic template of *there* cannot analyze sentences with numerals, so we set the meaning of *there is/are* to null and adopt the meaning of subsequent parts as the meaning of the whole sentence. For this reason, we select only trees with a *there is/are NP* structure.

We choose from among the selected trees the one with the highest probability. We then change the categories of numerals from N/N to NP/N so that their semantic representation is appropriately assigned in the next step, semantic parsing.

3.3.2. Semantic Parsing

Next, we use semantic parsing to derive logical semantic representations of the hypothesis sentence. Semantic parsing is performed according to the CCG parse trees using ccg2lambda (Martínez-Gómez et al., 2016) by lambda calculus. We use an *AtLeast* binary predicate as the semantic representation of numerals.³ When F is a predicate and n is a constant representing an integer,

³Numerals can express both the meaning of *at least* and the meaning of *exactly* (Bylinina and Nouwen, 2020). We use the *at least* meaning in this paper, but it is also possible to express the *exactly* meaning in the same way and perform inferences using that meaning by changing semantic templates.

$$\begin{aligned}
D &= \{X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8\} \\
V &= \{(\text{fork}, \{X_1\}), (\text{knife}, \{X_2, X_3, X_4\}), (\text{plate}, \{X_5, X_6, X_7\}), (\text{table}, \{X_8\}), \\
&\quad (\text{white}, \{X_5, X_6, X_7\}), (\text{round}, \{X_5, X_6, X_7\}), (\text{wood}, \{X_8\}), \\
&\quad (\text{on}, \{(X_1, X_8), (X_2, X_8), (X_3, X_8), (X_4, X_8), (X_5, X_8), (X_6, X_8), (X_7, X_8)\})\}
\end{aligned}$$

Figure 2: FOL model of the image in Figure 1.

$\text{AtLeast}(F, n)$ represents *there are at least n entities satisfying F* . For example, the logical expression

$$\text{AtLeast}(\lambda x. (\text{plate}(x) \wedge \text{white}(x)), \text{three})$$

means *there are at least three white plates*. In `cgg2lambda`, semantic templates define CCG lexical items. We use the semantic template shown in Table 2 to compose the meanings of numerals.

To improve the efficiency of model checking, we also define a binary predicate $\text{Obj}(c, x)$, where c is a constant representing an object name, and x is an entity. $\text{Obj}(c, x)$ means the name of the entity x is c . All nouns in the sentence are mapped to this Obj predicate.

3.4. Model Checking

3.4.1. Overview

We perform model checking between the model of the premise image and the logical formula of the hypothesis sentence. The system predicts *entailment* if the output is true, and predicts *non-entailment* otherwise. In model checking, we use the NLTK (Bird and Loper, 2004) evaluation function with additional processing for the AtLeast and Obj predicates.

3.4.2. Knowledge Injection

Robust inference requires capturing paraphrases (different words or phrases with the same meaning) between the model of the premise image and the logical formula of the hypothesis sentence. We employ two types of paraphrase recognition in the model checking process to achieve this: one using synonyms and hypernyms, the other using word embedding similarities. These methods are applicable when the model includes the attributes and relationships of the hypothesis, but cannot prove attributes and relationships that are not explicitly included in the model. Since models may not encompass all the information in the premise images, we retrieve and incorporate additional knowledge directly from visual information.

Synonyms and Hypernyms WordNet (Miller, 1995) is an extensive lexical database of semantic

relations between English words. We use WordNet synsets (groups of synonyms) to recognize whether words in the model and the formula are synonyms. We also address paraphrasing with hypernyms. Namely, if a word in the formula is a hypernym of a word in the model, an entailment relation holds. For example, if the model contains an object named *dog*, the hypothesis sentence *there is an animal* is true. We use hypernym relations in WordNet to handle such a paraphrase between *animal* and *dog*. We call this knowledge injection method **WordNet**.

Word Embedding We use word embeddings from the pretrained Word2Vec (Mikolov et al., 2013) model to cover paraphrases not included in WordNet. Namely, we calculate the cosine similarity between word or phrase embeddings in the model and one in the formula. The phrase embedding is computed by averaging the embeddings of the words in the phrase. We determine that two words are paraphrases if their similarity exceeds a threshold. We experimented with several threshold values and adopted those that achieved the best performance (0.40 for objects and 0.90 for attributes and relationships). We call this method **Word2Vec**. For example, *handle grip* and *grip* have similar embeddings (their similarity is 0.79), so they are determined to be paraphrases. Note that this method sometimes introduces inconsistencies because there are cases where words that are not paraphrases have similar embeddings, such as *dog* and *cat*, which tend to be used in the same context.

Knowledge Injection from Raw Images We use embedding vectors from the vision-and-language model CLIP (Radford et al., 2021) to retrieve visual information in images. The CLIP model is trained on many image-text pairs using contrastive learning to predict high scores for correct image-text pairs and low scores for others. We use CLIP’s image and text encoders to obtain embedding vectors of object images and phrases. We describe the procedure for knowledge injection on attributes and relationships using these embeddings.

Attribute To infer whether an object *obj* in the model has an attribute *attr* described in the

CCG category	NP/N
POS tag	CD
Semantics	$\lambda E, F_1, F_2, F_3. \text{AtLeast}(\lambda x. (F_1(x) \wedge F_2(x) \wedge F_3(x)), E)$

Table 2: Semantic template of numerals. This template means that when the CCG category is NP/N and the part-of-speech (POS) tag is CD (cardinal number), the semantics represented as a lambda expression is given. E in the lambda expression is assigned a numeral.

hypothesis, we compute embedding vectors of the object’s image and the two phrases, “*obj attr*” and “*attr obj*.” We then calculate the cosine similarities between the image embedding and the two phrase embeddings. If either similarity exceeds a threshold, we conclude that the object *obj* has the attribute *attr*.

Relationship To infer whether a relationship *rel* in the hypothesis holds between two objects *obj1* and *obj2* in the model, we compute embedding vectors of the image covering the two objects and the phrase “*obj1 rel obj2*.” We then compute the cosine similarity between the image and the phrase. If the similarity exceeds a threshold, we conclude that the relationship *rel* holds between the two objects *obj1* and *obj2*.

As with the word embedding thresholds for Word2Vec knowledge injection, we set the thresholds used in these methods to 0.20 from the results of prior experiments. We call this knowledge injection method **CLIP**.

We demonstrate an example of knowledge injection using CLIP embeddings. The image in Figure 1 entails the sentence *there is an empty plate*. The meaning representation of this sentence is

$$\exists x. (\text{Obj}(\text{plate}, x) \wedge \text{empty}(x)).$$

Among the entities in the model (Figure 2), those with the name plate are X_5 , X_6 , and X_7 . Here, we consider the case where X_5 is assigned to x . The next step is to check the truth of $\text{empty}(X_5)$, but the model has no empty attribute and no synonyms or hyponyms of empty. In that case, we calculate the CLIP embedding similarities between the image of object X_5 and the two phrases, *empty plate* and *plate empty*. Since these similarities (0.31 for *empty plate* and 0.28 for *plate empty*) exceed the threshold (0.20), the entity X_5 is determined to have the empty attribute, so the sentence is evaluated as true.

4. Experiments

4.1. Dataset

In existing VTE datasets, each hypothesis sentence is independent, and predicting the correct

entailment relation does not indicate whether the VTE system accurately understands quantities or negation. Therefore, we created a dataset to measure an accurate understanding of quantities and negation. The dataset was created from VG (Krishna et al., 2017), with each problem consisting of a premise image and two hypothesis sentences where the answer is *entailment* and *non-entailment*, respectively. The correct label must be predicted for both entailment and non-entailment sentences to solve each problem.

The dataset contains 100 problems with quantities and 100 problems with negation. In both cases, the entailment sentence is based on a phrase in VG describing the premise image, while the non-entailment sentence is created by changing the numeral of the entailment sentence or by switching the presence of negation. For negation, we deal with simple syntactic negations such as *not* and *no*. Table 3 shows examples of the created problems.

4.2. Experimental Setting

We compared the inference accuracy and execution time of our system with the deep learning-based models (ViLT, Kim et al., 2021 and FLAVA, Singh et al., 2022) and the logical inference VTE system (Suzuki et al., 2019) using our created dataset. We used the two different ViLT models fine-tuned on NLVR2 (Suhr et al., 2019) and SNLI-VE (Xie et al., 2019), respectively, and the FLAVA model fine-tuned on SNLI-VE. SNLI-VE (Xie et al., 2019) is a large standard VTE dataset. Since the VTE task in SNLI-VE is a three-class classification (*entailment*, *contradiction*, or *neutral*), *contradiction* and *neutral* are treated as *non-entailment* for the models fine-tuned on SNLI-VE. NLVR2 (Suhr et al., 2019) is another visual reasoning dataset wherein the task is to determine the correctness of a given caption for a pair of images. Captions within NLVR2 cover various linguistic phenomena, including quantity and negation. When making predictions using the models fine-tuned on NLVR2, we utilized pairs of identical premise images. In the experiment with the system of Suzuki et al. (2019), we used the VG and GQA scene graphs with a proof timeout of 60 seconds. For our system, we experimented with the VG, GQA, VG+GQA, and VG+GQA+OD graphs and applied all three knowledge injection methods.

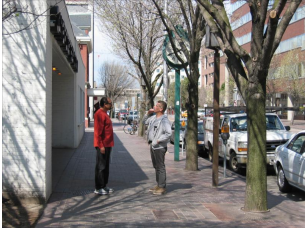
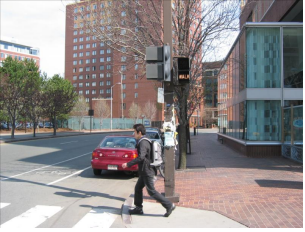
	(a)	(b)
Premise		
Hypothesis	E Two men are standing on the road.	The man is wearing a silver backpack.
	N Three men are standing on the road.	The man is not wearing a silver backpack.

Table 3: Example problems in the created VTE dataset. E and N stand for entailment and non-entailment sentences, respectively. The premise images are obtained from Visual Genome (Krishna et al., 2017). Problem (a) contains quantities, and problem (b) contains negation.

Model	Scene graph	Accuracy		Execution time (s)
		Quantity	Negation	
Random	-	0.25	0.25	-
ViLT (NLVR2)	-	0.07	0.38	0.763
ViLT (SNLI-VE)	-	0.16	0.38	1.405
FLAVA (SNLI-VE)	-	0.14	0.36	1.050
Suzuki et al. (2019)	VG	0.08	0.49	3.429
	GQA	0.04	0.48	1.840
Ours	VG	0.17	0.49	0.629
	GQA	0.20	0.46	0.406
	VG+GQA	0.20	0.47	0.456
	VG+GQA+OD	0.14	0.47	0.652

Table 4: Accuracy and average execution times (seconds) for inference. Random is the random baseline, and Ours is the proposed system. Datasets used for fine-tuning are shown in parentheses after model names. Scene graphs are not used in the deep learning-based models.

We also compared our results to a random baseline. In our dataset, models are required to accurately predict labels for both entailment and non-entailment sentences simultaneously to solve each problem. Given that the accuracy of random predictions for each sentence label is 0.5, the random baseline for this task is 0.25 ($= 0.5^2$).

In addition, we conducted ablation studies of knowledge injection to measure the effect of each knowledge injection method. We evaluated our system with various knowledge injection settings using the GQA scene graphs.

4.3. Results

4.3.1. Overall Results

The results with four different systems are shown in Table 4. The proposed method achieved the highest accuracy for quantity problems when using the GQA or VG+GQA scene graphs and the

highest accuracy for negation problems when using the VG scene graphs. Compared to the existing approaches, our method with the GQA or VG+GQA scene graphs achieved the highest accuracy for quantity problems. For negation problems, our method showed better accuracy than deep learning-based models and was comparable to the method of Suzuki et al. (2019). Furthermore, the execution times were shorter than the method of Suzuki et al. (2019) and comparable to the deep learning-based models.

Since the entailment and non-entailment sentences in quantity problems are identical except for the numerals, deep learning-based methods are prone to errors by embedding both sentences similarly and predicting the same label for both. On the other hand, logic-based methods are prone to errors due to incorrectly capturing the number of objects in the image.

Knowledge injection			Accuracy		Execution time (s)
WordNet	Word2Vec	CLIP	Quantity	Negation	
x	x	x	0.05	0.50	0.002
✓	x	x	0.07	0.52	0.034
x	✓	x	0.05	0.58	0.006
✓	✓	x	0.08	0.58	0.023
x	x	✓	0.14	0.44	0.212
✓	x	✓	0.18	0.41	0.285
x	✓	✓	0.12	0.48	0.415
✓	✓	✓	0.20	0.46	0.406

Table 5: Accuracy and average execution times (seconds) for model checking in each knowledge injection setting. GQA scene graphs were used. Details of each knowledge injection method are described in Section 3.4.2.


Premise	
Hypothesis	E There are no metallic chairs. N There are metallic chairs.

Table 6: Negation problem on which the CLIP knowledge injection led to incorrect predictions. E and N stand for entailment and non-entailment sentences, respectively. The premise image is obtained from Visual Genome (Krishna et al., 2017).

4.3.2. Knowledge Injection

Table 5 shows the results of various knowledge injection settings using the GQA scene graphs. For quantity problems, the highest accuracy was obtained by applying all types of knowledge injection, indicating the effectiveness of these methods. However, for negation problems, the accuracy was higher without CLIP knowledge injection than with it. CLIP knowledge injection also made the inference slower because it takes time to encode images and text to obtain feature embeddings. We present a detailed error analysis in Section 5.2.

5. Discussion

5.1. Execution Time

The original Suzuki’s approach uses automated theorem proving, which is generally slower than model checking. Because some images in the dataset contain many objects, and the models created from such images are complex, Suzuki’s approach using automated theorem proving took a long time to reason in such cases. In contrast,

the proposed method uses model checking, which is relatively fast even for problems involving such a large number of entities. For instance, when performing inference on an image with 29 objects using a VG scene graph, Suzuki’s method required over 60 seconds for inference, whereas the proposed method completed model checking in 1.18 seconds.

5.2. Error Analysis

We tried to improve inference accuracy by combining scene graphs from the two datasets and using object detection, but the accuracy did not improve substantially. This failure was due to object duplication. In order to accurately solve VTE tasks, especially those involving quantities, scene graphs need to contain exact information on objects in the image. However, the VG scene graphs contain duplicated objects, and although the duplication was removed in the process of creating the VG+GQA graphs, that removal was insufficient. In addition, we supplemented object information by object detection, but the duplication of objects increased in some cases, resulting in more inaccurate graphs.

Also, the accuracy on the negation problems decreased when CLIP knowledge injection was added, because this knowledge injection introduces erroneous attributes and relationships. For example, Table 6 shows a problem where CLIP knowledge injection led to wrong predictions. When using only paraphrase recognition for knowledge injection, both entailment and non-entailment sentences were predicted correctly. However, using CLIP knowledge injection, *chairs* were determined to have attribute *metallic*. The system thus predicted the label of the entailment sentence as *non-entailment* and the label of the non-entailment sentence as *entailment*.

We used the similarity of Word2Vec word embeddings in paraphrase recognition, but sometimes this approach failed. For example, *computer*

and *computer mouse* are not the same objects. However, the cosine similarity between their word embeddings is 0.83, which exceeds the similarity threshold for objects (0.40). Therefore, they were recognized as paraphrasing, and the object *computer mouse* was incorrectly counted as a computer.

In this experiment, approximately 80% of the incorrect predictions were due to inaccurate scene graphs or incorrect knowledge injection as described above, and parsing errors were relatively rare in this VTE task.

6. Conclusion

We presented a logic-based VTE system using model checking and knowledge injection. We also created a VTE dataset to evaluate the performance on problems containing quantities and negation. The results showed that our system solves VTE tasks involving such linguistic phenomena more robustly than the existing approaches. Also, by using model checking, our system successfully increases the efficiency of inference compared to the existing logical inference approach.

In future work, we hope to correct the errors mentioned in Section 5.2. We are also considering applying the proposed method to arbitrary images that are not annotated with scene graphs by utilizing scene graph generation (Yang et al., 2018; Zellers et al., 2018) and a method to predict FOL models directly from images (Hürlimann and Bos, 2016). In recent years, there has been development in models addressing visual reasoning tasks through programs generated by large language models (Gupta and Kembhavi, 2023; Surís et al., 2023). We plan to evaluate these models as part of our research.

7. Acknowledgements

We thank the three anonymous reviewers for their helpful comments and suggestions, which improved this paper. This work was supported by JST, PRESTO grant number JPMJPR21C8, Japan.

Limitations

The current parsing method cannot derive an appropriate semantic representation of phrases such as *in front of* because phrases consisting of multiple words are split into individual words. Combining phrases into a single token before parsing is necessary for more robust inference. Also, our system only supports cardinal numerals and cannot correctly parse other numerical expressions such as ordinal numerals (e.g., *first*) and numerical comparatives (e.g., *more than two*).

Another limitation of our method is the meaning representation of numerals. The current representation of numerals is distributive, which means each of multiple objects individually does one thing. There is another meaning of numerals called collective readings, in which multiple objects collectively do one thing (e.g., *Two men meet*). We need to modify the derivation of logical formulas and the model checking procedure to support such readings.

Moreover, in the experiments conducted in this study, we only used a small dataset that we created for the evaluation. In order to accurately compare performance with other inference models, it is necessary to use large datasets, such as NLVR2 and SNLI-VE, which are used to train these models. However, current large datasets do not contain scene graphs of premise images, and there are no existing VTE datasets focusing on quantities and negation. We thus manually created a small dataset focusing on these phenomena from Visual Genome.

Another issue is the setting of thresholds. The proposed method uses thresholds for (i) checking whether two objects are identical when merging scene graphs, (ii) recognizing paraphrases using Word2Vec embeddings, and (iii) determining whether a phrase correctly describes an object in an image using CLIP embeddings. These thresholds directly affect inference accuracy, but automatically calculating optimal threshold values is challenging. In this study, therefore, we manually set thresholds that achieved the best performance in preliminary experiments.

8. Bibliographical References

- Manoj Acharya, Kushal Kafle, and Christopher Kanan. 2019. Tallyqa: Answering complex counting questions. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):8076–8084.
- Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.
- Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain. Association for Computational Linguistics.
- Lisa Bylinina and Rick Nouwen. 2020. Numeral

- semantics. *Language and Linguistics Compass*, 14(8):e12390.
- Prithvijit Chattopadhyay, Ramakrishna Vedantam, Ramprasaath R. Selvaraju, Dhruv Batra, and Devi Parikh. 2017. Counting everyday objects in everyday scenes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [BERT: Pre-training of deep bidirectional transformers for language understanding](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Tanmay Gupta and Aniruddha Kembhavi. 2023. Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14953–14962.
- Ronghang Hu and Amanpreet Singh. 2021. Unit: Multimodal multitask learning with a unified transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1439–1449.
- Drew A Hudson and Christopher D Manning. 2019. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6700–6709.
- Manuela Hürlimann and Johan Bos. 2016. [Combining lexical and spatial knowledge to predict spatial relations between objects in images](#). In *Proceedings of the 5th Workshop on Vision and Language*, pages 10–18, Berlin, Germany. Association for Computational Linguistics.
- Justin Johnson, Agrim Gupta, and Li Fei-Fei. 2018. Image generation from scene graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. 2015. [Image retrieval using scene graphs](#). In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3668–3678.
- Wonjae Kim, Bokyung Son, and Ildoo Kim. 2021. [Vilt: Vision-and-language transformer without convolution or region supervision](#). In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 5583–5594. PMLR.
- Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73.
- Tomoya Kurosawa and Hitomi Yanaka. 2022. [Logical inference for counting on semi-structured tables](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 84–96, Dublin, Ireland. Association for Computational Linguistics.
- Wei Li, Can Gao, Guocheng Niu, Xinyan Xiao, Hao Liu, Jiachen Liu, Hua Wu, and Haifeng Wang. 2021. [UNIMO: Towards unified-modal understanding and generation via cross-modal contrastive learning](#). In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2592–2607, Online. Association for Computational Linguistics.
- Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *Computer Vision – ECCV 2014*, pages 740–755, Cham. Springer International Publishing.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*. Version 1.
- Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. [Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks](#). In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.
- Pascual Martínez-Gómez, Koji Mineshima, Yusuke Miyao, and Daisuke Bekki. 2016. [ccg2lambda: A compositional semantics system](#). In *Proceedings of ACL-2016 System Demonstrations*, pages 85–90, Berlin, Germany. Association for Computational Linguistics.

- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.
- Letitia Parcalabescu, Albert Gatt, Anette Frank, and Iacer Calixto. 2021. [Seeing past words: Testing the cross-modal capabilities of pretrained V&L models on counting tasks](#). In *Proceedings of the 1st Workshop on Multimodal Semantic Representations (MMSR)*, pages 32–44, Groningen, Netherlands (Online). Association for Computational Linguistics.
- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. [Learning transferable visual models from natural language supervision](#). In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR.
- Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.
- Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. [Faster r-cnn: Towards real-time object detection with region proposal networks](#). In *Advances in NeurIPS*, volume 28. Curran Associates, Inc.
- Jiaxin Shi, Hanwang Zhang, and Juanzi Li. 2019. Explainable and explicit visual reasoning over scene graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Amanpreet Singh, Ronghang Hu, Vedanuj Goswami, Guillaume Couairon, Wojciech Galuba, Marcus Rohrbach, and Douwe Kiela. 2022. FLAVA: A foundational language and vision alignment model. In *CVPR*.
- Mark Steedman. 2000. *The Syntactic Process*. The MIT Press.
- Alane Suhr, Stephanie Zhou, Ally Zhang, Iris Zhang, Huajun Bai, and Yoav Artzi. 2019. [A corpus for reasoning about natural language grounded in photographs](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6418–6428, Florence, Italy. Association for Computational Linguistics.
- Dídac Surís, Sachit Menon, and Carl Vondrick. 2023. [Vipergpt: Visual inference via python execution for reasoning](#). *arXiv preprint arXiv:2303.08128*.
- Riko Suzuki, Hitomi Yanaka, Masashi Yoshikawa, Koji Mineshima, and Daisuke Bekki. 2019. [Multi-modal logical inference system for visual-textual entailment](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 386–392, Florence, Italy. Association for Computational Linguistics.
- Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi. 2017. [Inception-v4, inception-resnet and the impact of residual connections on learning](#). *Proc. of the AAAI Conference on Artificial Intelligence*, 31(1).
- Bart Thomee, David A. Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. 2016. [Yfcc100m: The new data in multimedia research](#). *Commun. ACM*, 59(2):64–73.
- Alexander Trott, Caiming Xiong, and Richard Socher. 2018. Interpretable counting for visual question answering. In *International Conference on Learning Representations*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#). In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.
- Ning Xie, Farley Lai, Derek Doran, and Asim Kadav. 2019. Visual Entailment: A novel task for fine-grained image understanding. *arXiv preprint arXiv:1901.06706*. Version 1.
- Jianwei Yang, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. 2018. Graph r-cnn for scene graph generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*.
- Masashi Yoshikawa, Hiroshi Noji, and Yuji Matsumoto. 2017. [A* CCG parsing with a supertag and dependency factored model](#). In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 277–287, Vancouver, Canada. Association for Computational Linguistics.
- Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. 2018. Neural motifs: Scene graph parsing with global context. In *Conference on Computer Vision and Pattern Recognition*.

Yan Zhang, Jonathon Hare, and Adam Prügell-Bennett. 2018. Learning to count objects in natural images for visual question answering. In *International Conference on Learning Representations*.

9. Language Resource References

Miller, George A. 1995. *WordNet: A Lexical Database for English*. Association for Computing Machinery, ISLRN 379-473-059-273-1.