# Visual Question Answering and the relation between language, perception and the world

**Staffan Larsson** **Bill Noble** **Robin Cooper**
Centre for Linguistic Theory and Studies in Probability (CLASP)
Department of Philosophy, Linguistics and Theory of Science
University of Gothenburg, Sweden
staffan.larsson@ling.gu.se bill.noble@gu.se cooper@ling.gu.se

## Abstract

We outline a general account of VQA using a perception-oriented formal semantic framework. We believe that it is instructive to describe VQA not only in terms of an engineering challenge, but also as a linguistically and philosophically relevant task that can help us better understand the relation between language, perception and the world. Specifically, we will argue that linguistic meaning helps us structure our takes on visual scenes, enabling us to classify situations so that we e.g. can answer questions and determine whether a sentence correctly describes a scene.

## 1 Introduction

When studying abstract matters like the relation between language, perception and the world, it is often helpful to look at specific problems. Visual Question Answering (VQA) is a machine learning task where the model is provided with an image and a natural language question and must provide an answer to the question based on the image. The task is focused on the natural language and visual understanding capabilities of the model. Many VQA datasets, such as the eponymous dataset from Antol et al. (2015), are susceptible to relatively superficial correlations between the inputs and the answer (Zhang et al., 2016; Agrawal et al., 2016; Goyal et al., 2019), which allow machine learning models to avoid the complexity involved in associating language and world more generally.

While Visual LLMs generalize better on VQA and related tasks when compared to older vision-and-language models, LLMs are essentially "black boxes" which can be trained and deployed but whose inner workings are opaque. Hence, LLMs do not necessarily help us derive insights into the relation between language, perception and the world.

In formal semantics in the Montague tradition and Possible World Semantics, the meaning of a word such as "dog" is taken to be a function from possible worlds (and times) to its extension, i.e. the set of all dogs in that world. Such logical representations, however, have no natural interpretation in terms of neural implementation (be it artificial or biological), and offer no biologically plausible explanation of how language is related to sensory perception and the physical world. In modern computational linguistics the meaning of a word such as "dog" might be related to a statistically-based perceptual classifier that is able to recognize dogs or give a probability estimation that a given individual is a dog. This corresponds to our ability to recognize a dog when we see one and is also much more hopeful in terms of neural modelling.

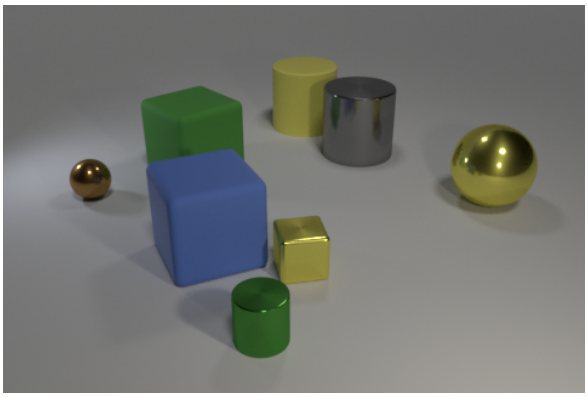

Figure 1: Consider this example from the CLEVR dataset (Johnson et al., 2016, image ID: 11568; question ID 115673). What is involved in answering the question about the scene? Q: *What color is the cylinder in front of the yellow cube?*

We adopt a view of linguistic meaning that relates to the notion of classifiers while at the same time preserving important techniques from formal semantics, such as the treatment of compositionality.

In this paper, we outline a general account of VQA, taking the CLEVR data set (see Figure 1) as an example and using a perception-oriented for-



*Proceedings of the 6th Workshop on Natural Language Meets Logic and Machine Learning (NALOMA)*, pages 60–70

mal semantic framework.[1] We believe that it is instructive to describe VQA not only in terms of an engineering challenge, but also as a linguistically and philosophically relevant task that can help us better understand the relation between language, perception and the world. Specifically, we will argue that linguistic meaning helps us structure our *take* on a visual scenes, enabling us to classify situations so that we can e.g. answer questions and determine whether a sentence correctly describes a scene.

## 2 TTR

TTR is a theory of types with records (Cooper and Ginzburg, 2015; Cooper, 2023). It is a broad semantic framework intended to account for how agents dynamically understand and interact with each other and the world. Thus, TTR accounts have been proposed for a multitude of linguistic and language related phenomena, including perception (Larsson, 2015, 2018), spatial and pragmatic reasoning (Dobnik and Cooper, 2017; Breitholtz, 2020), learning (Larsson and Cooper, 2009; Larsson, 2020b), quantification (Cooper, 2013; Lücking and Ginzburg, 2022; Cooper, 2023) and humour (Maraev et al., 2021).

We can only give a brief and partial introduction to TTR here; see also Cooper (2005, 2012); Cooper and Ginzburg (2015); Cooper (2023). To begin with, $s : T$ is a judgment that some $s$ is of type $T$. To make explicit who is making this judgment, the of-type relation may be subscripted with an agent $A$, as in $:_A T$. One *basic type* in TTR is Ind, the type of an individual; another basic type is $\mathbb{R}$, the type of real numbers. Given that $T_1$ and $T_2$ are types, $T_1 \rightarrow T_2$ is a *functional type* whose domain is objects of type $T_1$ and whose range is objects of type $T_2$.

Next, we introduce *records* and *record types*. If $a_1 : T_1, a_2 : T_2(a_1), \ldots, a_n : T_n(a_1, a_2, \ldots, a_{n-1})$, where $T(a_1, \ldots, a_n)$ represents a type $T$ which depends on the objects $a_1, \ldots, a_n$, the record to the left in Figure 2 is of the record type to the right.

In Figure 2, $\ell_1, \ldots \ell_n$ are *labels* which can be used elsewhere to refer to the values associated with them. A sample record and record type is shown in Figure 3.

Types constructed with predicates may be *dependent*. This is represented by the fact that arguments to the predicate may be represented by labels used on the left of the ':' elsewhere in the record type. In Figure 3, the type of $c_{man}$ is dependent on ref (as is $c_{run}$).

$$\begin{bmatrix} \ell_1 = a_1 \\ \ell_2 = a_2 \\ \ldots \\ \ell_n = a_n \\ \ldots \end{bmatrix} : \begin{bmatrix} \ell_1 : T_1 \\ \ell_2 : T_2(l_1) \\ \ldots \\ \ell_n : T_n(\ell_1, l_2, \ldots, l_{n-1}) \end{bmatrix}$$

Figure 2: Schema of record and record type

$$\begin{bmatrix} \text{ref} = \text{obj}_{123} \\ c_{man} = \text{prf}(\text{man}(\text{obj}_{123})) \\ c_{run} = \text{prf}(\text{run}(\text{obj}_{123})) \end{bmatrix} : \begin{bmatrix} \text{ref} : \text{Ind} \\ c_{man} : \text{man}(\text{ref}) \\ c_{run} : \text{run}(\text{ref}) \end{bmatrix}$$

Figure 3: Sample record and record type

If $r$ is a record and $\ell$ is a label in $r$, we can use a *path* $r.\ell$ to refer to the value of $\ell$ in $r$. Similarly, if $T$ is a record type and $\ell$ is a label in $T$, $T.\ell$ refers to the type of $\ell$ in $T$. Records (and record types) can be nested, so that the value of a label is itself a record (or record type). As can be seen in Figure 3, types can be constructed from predicates, e.g., "run" or "man". Such types are called *ptypes* and correspond roughly to propositions in first order logic.

A fundamental type-theoretical intuition is that something of a ptype $T$ is whatever it is that counts as a proof of $T$. One way of putting this is that "propositions are types of proofs". In Figure 3, we simply use prf($T$) as a placeholder for proofs of $T$; below, we will show how low-level perceptual input can be included in proofs.

Judgements (and other type actions, see Cooper (2023)) are regulated by action rules of the form

$$(1) \quad \frac{\varphi_0, \varphi_1, \ldots, \varphi_n}{\psi}$$

Each $\varphi_i$ is either an action or some other condition that must hold true and $\psi$ must be a type action such as a judgement. The wavy line as opposed to a straight line indicates that this is not a rule of inference but that the premises above the line holding *license* or *afford* the action below the line. Thus there is no guarantee that the agent will carry out the action even if all the premises hold.

[1] https://github.com/GU-CLASP/types-vectors-spiking-neurons/

## 3 Related work

VQA is a rich test bed for investigating the compositional nature of– and relationships between language, perception, and reasoning. Because of its structure, the CLEVR dataset has been used extensively to test systems that explicitly combine these components. The Memory, Attention, and Composition (MAC) network (Hudson and Manning, 2018), is a modular neural network with a recurrent unit that sequentially decomposes the question into operations that retrieve information from the image representation. While the MAC architecture is designed to accommodate compositional reasoning, it does not create explicit symbolic representations of the image or question. One such approach is that of Johnson et al. (2017), who trains a sequence-to-sequence *program generator* that transforms the natural language question into a sequence of functions. A neural module network, with modules for each function, is then used to sequentially execute the program and derive an answer. Improving on the already high accuracy of these previous two methods, Yi et al. (2018) achieves near-perfect results on CLEVR by introducing a module that generates symbolic sense representations. This allows for the use of a deterministic program executor that answers the parsed question. A key insight of these approaches is already present in the ground-truth question semantics of CLEVR itself: the meaning of a question can be represented as a procedure to determine its answer. This insight can also be found in the treatment of questions in Formal Semantics; by introducing a neuro-symbolic system with TTR at its core, we hope that insights from VQA can be generalized to other domains such as perceptually-grounded linguistic interaction.

In line with this proposal, there have been recent efforts to combine neural network representations with linguistic formal semantics. For example, Liu and Emerson (2022) use a Convolutional Neural Network (CNN) to extract entity representations from images. Treating a collection of these representations as a world model, they learn probabilistic lexical and compositional functions that take the entity representations in input and use image captions as the learning signal. The framework used in that work is inspired by linguistic theory, but only attempts to model very simple predicate-argument structures. Likewise, Bekki et al. (2021) use neural network-based entity representations embedded in a dense vector space, but use these representations as the basis for interpreting types in a dependent type system.

Looking more specifically to work on VQA involving TTR, Dobnik and Cooper (2017) assume data in the form of a point map representing a visual scene. The authors propose two functions which (if taken together) take a point map and produces a set of record types such that records of that type would specify an individual, a witness of a ptype predicating that the individual occupies a certain region, and a witness of a ptype predicating the property of the individual. However, only the types of such records are provided, not the records themselves.

Matsson et al. (2019) can be regarded as an implemented variant of the account in Dobnik and Cooper (2017). Matsson's implementation uses YOLO classifiers and is limited to polar questions. You Only Look Once (YOLO) (Redmon et al., 2016) is a neural network model for image recognition that detects and classifies objects in an image. So here, the starting point is an image rather than a point map as in Dobnik and Cooper (2017). In YOLO, each detected object is represented as bounding box coordinates, a class label and a confidence score.

In both Dobnik and Cooper (2017) and Matsson et al. (2019), representations stay on the type level and witnesses takes are not represented as such. This leaves open the question of what constitutes a witness for a ptype, whereas we want to suggest an answer this question, namely that witnesses of ptypes consists of tuples of high-level perceptual data (H-data) associated with the individuals involved in the ptype (typically, the arguments of a predicate).

The account in Utescher (2019) is more similar in spirit to the approach suggested in this paper, both in representing witness takes explicitly and in doing lazy classification conditioned by the need to answer questions. However, Utescher does not address the problem of structuring perceptual data to match the semantics of linguistic expressions as we do here.

We believe that the account presented below is an improvement both technically and philosophically by showing how, when relating linguistic expressions to visual scenes, we *use linguistic meaning to structure our take on the visual scene*, which includes finding the referents of linguistic expressions referring to objects in the scene. This also allows us to handle predicates of arity 2 or more in an elegant way, whereas the examples in Utescher

(2019) are limited to 1-place predicates. Furthermore, the account presented here is meant to generalise over specific kinds of visual data and classifiers.

## 4 A general TTR account of VQA

Larsson (2011, 2013, 2020a); Larsson et al. (2021); Larsson and Cooper (2021); Larsson (2021) present work towards a general formal perception-oriented semantics. Here, we revise and extend this account to fit with the VQA task. In Section 5 we will show how the general account can be adapted to the CLEVR dataset, where H-data is regarded as CLEVR scene graphs.

### 4.1 Low-level and high-level perceptual data

When talking about an agent's perception of a sitation, we can call the "raw" input "low level perceptual data" (L-data). This can be e.g. the retinal image in a biological system, a digital image from a camera connected to a computer (as in Matsson et al. (2019) and Utescher (2019)), or a point map (as in Dobnik and Cooper (2017)). We will use *LScene* for the type of L-data from a scene (a visually perceived situation).

When processing perceptual data, L-data is turned into "high level perceptual data" (H-data). H-data is the result of processing L-data, and this processing may include e.g. object detection, individuation, feature detection and compression. Individuation is needed to pick out the part of L-data that concerns a single individual. A biological example of H-data is visual cortical information (simplifying somewhat, the output of the visual cortex when fed the retinal image as input). Computational examples include a set of pairs of point maps and associated features (Dobnik and Cooper, 2017), a set of image and a list of tuples, each consisting of a bounding box, a label and a confidence value (as in YOLO), or an image and a set of bounding boxes corresponding to detected objects.

We can say that H-data is the result of applying a function $f_{\text{pp}}$ ('pp' for 'perceptual processing') to L-data. If an agent $A$ is visually perceiving a real-world situation $s$, the L-data on $A$'s retina is $s^A_L$ : *LScene* and the H-data in the output layer of $A$'s visual cortex is

(2) $s^A_H = f_{pp}(s^A_L)$ : *HScene*

where

(3) $f_{pp}$ : *LScene* $\rightarrow$ *HScene*

is the function of $A$'s visual cortex.

We will assume that H-data is individuated, which in TTR terms means that sections of H-data is associated with different objects of type *Ind*[2]. We define a type *HScene* as a list of H-data items corresponding to individuals in a perceived scene.

(4) *HScene=List*( $\begin{bmatrix} \text{id} & : & \textit{Ind} \\ \text{data} & : & \textit{HData} \end{bmatrix}$ )

We can now define a scene-specific judgement of being of a type *Ind* in a scene $s$:

(5) For $h$:*HScene*, $a :_h$ *Ind* iff for some $n$, $a = h.n.\text{id}$ (and for all $i$, $a = h.i.\text{id}, i = h$)

We use a hash table to access H-data associated with an individual (i.e., an object of type *Ind*)[3].

(6) If $h$ : $HScene$ and $a :_s$*Ind* then $\#_h a = h.n.data$ for $n$ such that $a = h.n.id$

The scope of the H-data associated with an individual $a$ is basically "any available information that can be relevant to understanding an utterance referring to $a$". If information about other things than $a$, for example information about other individuals, is needed to understand an utterance referring to $a$, then such information can be included in #$a$.

### 4.2 Perceptual data as evidence for ptypes

In general, we will be using H-data as evidence/proofs for ptypes, echoing the dictum "propositions are types of proofs". We will here limit our analysis to ptypes whose arguments are individuals (objects of type *Ind*). Ptypes constructed from an $n$-place predicates with arguments $a_1, \ldots, a_n$ have an $n$-tuple $\langle \#a_1, \ldots, \#a_n \rangle$ as evidence.

(7) $\langle$#a$\rangle$ : yellow(a)
$\langle$#a$\rangle$ : cube(a)
$\langle$#b, #a$\rangle$ : in-front(b,a)
$\langle$#b$\rangle$ : cylinder(b)

Judging whether some H-data evidence is of a ptype can be done using classifiers,[4] for example

[2]For most datasets, making individuation explicit in the way we do here is trivial.

[3]The hash table is thus relative to an HScene $h$. However, we will be surpressing this subscript for the remainder of this paper.

[4]The examples (8) and (9) may appear confusing to a type-theorist. The left-hand side is a judgement in type theory whereas the right-hand side belongs not to type theory but to the world of classifiers. The classifiers are used to place constraints on what judgements are available in the type theory.

(assuming a colour classifier $\kappa_{\text{col}}$ whose domain is *Hdata* and whose range are colour labels RED, BLUE etc, and assuming that $a$ and $b$ are variables over objects of type *Ind*):

(8) $\langle \#a \rangle$ : red($a$) if $\kappa_{\text{col}}(\#a)$=RED
$\langle \#a \rangle$ : blue($a$) if $\kappa_{\text{col}}(\#a)$=BLUE
...

...and similarly for 2-place ptypes assuming a left-or-right classifier $\kappa_{\text{lor}}$:

(9) $\langle \#a, \#b \rangle$ : right($a, b$) if $\kappa_{\text{lor}}(\#a, \#b)$=RIGHT;
$\langle \#a, \#b \rangle$ : left($a, b$) if $\kappa_{\text{lor}}(\#a, \#b)$=LEFT

### 4.3 Natural language semantics

In Larsson (2020a), meanings of linguistic expressions are defined as objects of type *Mng*[5]:

(10) $\mathit{Mng}=\begin{bmatrix} \text{bg} & : & \mathit{RecType} \\ \text{intrp} & : & \text{bg}\rightarrow\mathit{Type} \\ \text{clfr} & : & \text{bg}\rightarrow\mathit{Type} \end{bmatrix}$

Intuitively, for a declarative utterance with meaning $p$:*Mng*, $p$.bg is the type of the context in $p$ is appropriately uttered, $p$.intrp is the context-independent meaning of the utterance (a function from a situation to a type) and $p$.clfr is a function that evaluates whether the utterance of $p$ holds of the situation.

In general, the bg field can be understood as capturing the presupposition of a linguistic expression. When interpreting linguistic expressions referring to visual scenes, an agent needs to structure their take on the scene to reflect these presuppositions. For example, "the red cube" needs to be mapped to a specific object (which is red and a cube) in the agent's take on the scene. This means that the agent, starting from low-level perceptual data, needs to individuate and object which is classified as both red and as cube (and also to make sure no other object can be individuated which can be classified as red and a cube).

For an utterance $u$ of a declarative sentence with meaning $p$:*Mng* and a situation $s$:$p$.bg, we can provide an action rule specifying when an agent $A$ may judge ($A$'s take on the situation) $s$ to be of the type specified by the (situated interpretation of) $u$ in $s$.

(11) $\dfrac{p :_A \mathit{Mng} \qquad s :_A p.\text{bg} \qquad p.\text{clrf}(s) = p.\text{intrp}(s)}{s :_A p.\text{intrp}(s)}$

[5]We are here disregarding the 'params' field.

So an agent is licensed to judge $s$ to be of the type $T$ specified by the (situated interpretation of the) $u$ in $s$ provided that $s$ fulfills the presuppositions of $p$, and $A$ perceives the situation to be of type $T$. For example, the meaning of the sentence p="The cylinder in front of the yellow cube is green" could be:[6]

(15) p=$\begin{bmatrix} \text{bg} = \begin{bmatrix} \text{x} & : & \mathit{Ind} \\ \text{colour}_{\text{x}} & : & \text{yellow(x)} \\ \text{shape}_{\text{x}} & : & \text{cube(x)} \\ \text{y} & : & \mathit{Ind} \\ \text{place}_{\text{yx}} & : & \text{in-front(y,x)} \\ \text{shape}_{\text{y}} & : & \text{cylinder(y)} \end{bmatrix} \\ \text{intrp}=\lambda r :\text{bg}.\left[\text{colour}_{\text{y}}:\text{green}(r.\text{y})\right] \\ \text{clfr} =\lambda r :\text{bg}.\ \left[\text{colour}_{\text{y}}:T_{\text{col}}(r.\text{y})\right] \end{bmatrix}$

where $T_{\text{col}}$ is a dependent type that takes a tuple of relevant H-data, calls a classifier function $\kappa_{\text{col}}$, and returns a ptype, e.g. red($r$.y) or green($r$.y) depending on the output of the classifier:

(16) $T_{\text{col}} : \mathit{Ind} \rightarrow \mathit{Type}$

(17) $T_{\text{col}}(x) =$
$\begin{cases} \text{red}(x) \text{ if } \kappa_{\text{col}}(\langle \#x \rangle)=\text{RED} \\ \text{blue}(x) \text{ if } \kappa_{\text{col}}(\langle \#x \rangle)=\text{BLUE} \\ \ldots \end{cases}$

For a two-place relation[7]:

[6]We are here and for the remainder of this paper for simplicity of exposition ignoring the uniqueness presupposition of the definite article. Nevertheless, for completeness we indicate in this footnote how to remedy this.

(12) If $T$ : *RecType* and $\ell \in \text{labels}(T)$, then $T_{\text{uniq}(\ell)}$ is a type.

(13) $o : T_{\text{uniq}(\ell)}$ iff $o : T$ and there is an $a$ such that for all $r :_H T$, $r.\ell.\text{x} = a$

Here, we use '$:_H T$' for a type judgement relative to an H-scene $H$, to establish whether $T$ could be instantiated with respect to $H$, that is, whether a record of this type could be created from $H$ with respect to the basic types and ptypes in $T$.

Given the above, the meaning of the sentence is now

(14) $\begin{bmatrix} \text{bg} = \begin{bmatrix} \text{o}_1 : \begin{bmatrix} \text{x:}\mathit{Ind} \\ \text{colour:yellow(x)} \\ \text{shape:cube(x)} \end{bmatrix}_{\text{uniq(x)}} \\ \text{o}_2 : \begin{bmatrix} \text{x:}\mathit{Ind} \\ \text{place:in-front(x,}\Uparrow\text{o}_1\text{.x)} \\ \text{shape:cylinder(x)} \end{bmatrix}_{\text{uniq(x)}} \end{bmatrix} \\ \text{intrp}=\lambda r\,\text{bg}\,.\ \left[\text{c:green}(r.\text{o2.x})\right] \\ \text{clfr} =\lambda r\,\text{bg}\,.\ \left[\text{c:}T_{\text{col}}(r.\text{o2.x})\right] \end{bmatrix}$

Other definitions in this paper would need to be adapted accordingly.

[7]We are here assuming that $\kappa_{\text{lor}}$ always returns LEFT or RIGHT.

(18) $T_{\text{place}}(x, y)$ =
$\begin{cases} \text{left}(x, y) \text{ if } \kappa_{\text{lor}}(\langle \#x, \#y \rangle) = \text{LEFT} \\ \text{right}(x, y) \text{ if } \kappa_{\text{lor}}(\langle \#x, \#y \rangle) = \text{RIGHT} \end{cases}$

Assuming that $s$ provides information that can be used by a classifier to decide whether the cylinder in front of the yellow cube is green or not, an agent $A$ may judge $s :_A \text{green}(s.\text{y})$ provided $T_{\text{col}}(s.\text{y}) = \text{green}(s.\text{y})$.

As mentioned above, H-data tuples are regarded as objects of ptypes that witness (i.e. provide evidence or proof of) the ptype. Consequently, a record type containing ptypes (such as the bg field above) can be instantiated by a record where values of ptype fields are H-data tuples. We use such records to represent an agent's take on a scene, a so-called 'witness take' . A situated interpretation of a linguistic expression $e$ is attained by applying $[e]$.bg to a witness take $s$.

Assume that the declarative sentence above is stated about a focus situation $s_{123}$ to an agent $A$ whose take on this situation is

(19) $\text{s}^A_{123} = \begin{bmatrix} \text{x} & = & \text{a} \\ \text{colour}_\text{x} & = & \langle \#\text{a} \rangle \\ \text{shape}_\text{x} & = & \langle \#\text{a} \rangle \\ \text{y} & = & \text{b} \\ \text{place}_\text{yx} & = & \langle \#\text{b}, \#\text{a} \rangle \\ \text{shape}_\text{y} & = & \langle \#\text{b} \rangle \\ \text{colour}_\text{y} & = & \langle \#\text{b} \rangle \end{bmatrix}$

To evaluate whether p is true for $\text{s}^A_{123}$, $A$ computes p.intrp($\text{s}^A_{123}$) and p.clfr($\text{s}^A_{123}$) and checks for equality.

(20) p.intrp($\text{s}^A_{123}$)=
$\lambda r{:}\begin{bmatrix} \text{x} & :\textit{Ind} \\ \text{colour}_\text{x} & :\text{yellow(x)} \\ \text{shape}_\text{x} & :\text{cube(x)} \\ \text{y} & :\textit{Ind} \\ \text{place}_\text{yx} & :\text{in-front(y,x)} \\ \text{shape}_\text{y} & :\text{cylinder(y)} \end{bmatrix}.\left[\text{colour}_\text{y}{:}\text{green}(r.\text{y})\right](\text{s}^A_{123})=$
$\left[\text{colour}_\text{y}{:}\text{green(b)}\right]$

(21) p.clfr($\text{s}^A_{123}$) = $\left[\text{colour}_\text{y}{:}\text{red(b)}\right]$
assuming that $\kappa_{\text{col}}(\langle \#\text{b} \rangle)$=RED and hence $T_{\text{col}}$(b)=red(b).

In this case, the equality fails and hence p is regarded as not true of $s_{123}$.

### 4.4 Questions

Now, what about questions? In this case, we suggest that the situated interpretation is not a Ptype but instead a function from an answer (in this case, a predicate function) to a Ptype. A question q="What color is the cylinder in front of the yellow cube?" could then have the meaning:

(22) q=
$\begin{bmatrix} \text{bg} = \begin{bmatrix} \text{x} & :\text{Ind} \\ \text{colour}_\text{x} & :\text{yellow(x)} \\ \text{shape}_\text{x} & :\text{cube(x)} \\ \text{y} & :\text{Ind} \\ \text{place}_\text{yx} & :\text{in-front(y,x)} \\ \text{shape}_\text{y} & :\text{cylinder(y)} \end{bmatrix} \\ \text{intrp}=\lambda r :\text{bg}.\lambda P{:}(\textit{Ind}\rightarrow\textit{Type}).\left[\ \text{colour}_\text{y} : P(r.\text{y})\ \right] \\ \text{clfr} =\lambda r :\text{bg}.\left[\text{colour}_\text{y}{:}T_{\text{col}}(r.y)\right] \end{bmatrix}$

Whereas for declaratives, we compared the interpretation output with the classifer output, for questions these functions have different roles. Specifically, the interpretation function allows an agent to compute a Ptype from the question and an (elliptical) answer, whereas the classifier function can be used to compute the answer to the question based on the agent's take on the situation referred to. The former is useful in integrating an agent's verbal answer to the question[8]. The latter is useful for figuring out the answer (which could be used when providing a verbal answer).

Assume that q="What color is the cylinder in front of the yellow cube?" is asked about a focus situation $\text{s}_{123}$. This question presupposes that there is a (unique) yellow cube and a (unique) cylinder in front of it. An agent who is asked this question might use their take on the scene to find an answer. The idea here is that the answer to a question is attained by applying a function $q$.clfr derived from the question to the take on the scene that the question is about. This requires the scene to be of the type specified by $q$.bg.

(23) q.clfr =
$\lambda r{:}\begin{bmatrix} \text{x} & :\textit{Ind} \\ \text{c}_\text{col-x} & :\text{yellow(x)} \\ \text{c}_\text{shape-x} & :\text{cube(x)} \\ \text{y} & :\textit{Ind} \\ \text{c}_\text{fob-yx} & :\text{in-front(y,x)} \\ \text{c}_\text{shape-y} & :\text{cylinder(y)} \end{bmatrix}.\ \left[\text{colour}_\text{y}{:}T_{\text{col}}(r.\text{y})\ \right]$

### 4.5 Finding an answer to a question

We can imagine the question above being asked about a focus situation $s_{123}$ to an agent $A$ whose take on this situation is $s^A_{123}$ as above.

Finding the answer to a question about the focus situation involves intepreting the question in light of one's take on the focus situation. Formally, the question function is applied to the take on the focus situation to yield an answer to the question.

[8] Assuming that a response r="green" has the meaning [r].intrp $= \lambda x : \textit{Ind}.\text{green}(x)$ so that q.intrp($s$)([r].intrp) $= \left[\text{colour}_\text{y}{:}\text{green(a)}\right]$.

(24) q.clfr($s^A_{123}$) = $\left[\begin{array}{ccc} \text{colour}_y & : & \text{red(b)} \end{array}\right]$

again assuming that $\kappa_{\text{col}}(\langle \#b \rangle)$=RED.

Given the above, for a question $q$ we want to take an H-scene $H$ and construct a take on the focussed witness situation $s$:$q$.bg that $q$.clfr can be applied to to find the answer to $q$. For ease of exposition, we will below use $T_{\text{bg}}$ for $q$.bg. The approach we will take is this: From $H$ and $T_{\text{bg}}$, create $s_H$ such that $s_H : T_{\text{bg}}$. We want this process to work for any kind of H-data, as long as we have access to classifiers that can take that data as input and produce the required judgements.

### 4.6 Definition of witness takes

To build $s_H : T_{\text{bg}}$ from $H$, the idea is to label the individual objects in $H$ so that the ptype classifier results (ptypes) come out as specified in $T_{\text{bg}}$. To do this, we start from all possible labellings of (H-data concerning) individuals in the H-data and then filter them out by applying classifiers when typechecking against the ptypes in $T_{\text{bg}}$.

Assuming an scene $H$ : *HScene*, we create records from $H$, enumerating all possible labellings (records) $\mathcal{S}$ of $H$ that match $T_{\text{bg}}$. For a record type $T$ and an H-scene $H$, there is a function $\# : \mathit{Ind} \rightarrow \mathit{Hdata}$ and a set of records $\mathcal{S}$ (the set of witness takes) such that:

- For each $h \in H$, $\#a = h$.data iff $h$.ind=$a$
- For each $s \in \mathcal{S}$:
  - for all $\ell$ such that $T.\ell$ : *Ind*, there is an $a :_H \mathit{Ind}$ such that $s.\ell = a$
  - for all $\ell$ such that $T.\ell = \langle f, L \rangle$ where $L = \langle \ell_1, \ldots, \ell_n \rangle$, $s.\ell = \langle \#(s.\ell_1), \ldots \#(s.\ell_n) \rangle$
  - $s : T$

Given the above, we can now tentatively provide an action rule licensing an agent $A$ answering a question $q$ : *Question* based on perceptual data $H$ : *HScene*.[9]:

(25) $\underbrace{q :_A \mathit{Mng} \quad H :_A \mathit{HScene} \quad s_H :_A q.\text{bg}}_{:_A \textsc{Answer}(A, q.\text{clfr}(s_H))!}$

[9]The notation $:_A T!$ means that $A$ creates an object of type $T$, see Cooper (2023). We are here making several simplifications for reasons of space, including allowing non-exhaustive answers, not requiring that $q$ is conversationally relevant, and not defining the type QUESTION.

### 4.7 Building witness takes from H-data

Given a type $T$ with labels $labels(T)$ and a H-scene $H$, we want to find all of the records $s$ that are witness takes on $H$, i.e., for which $s : T_{\text{bg}}$ holds. Recall that each $h \in H$ has a field $h$.id such that $h'$.id $\neq h$.id for any $h' \neq h$.

We define a function BUILDTAKES that uses tree search to build up list of situation takes respecting $T$. As input, takes a situation type, $T$ , an instance of H-data, $H$, and a partial label map $\mathcal{L} : \{\ell \in labels(T) \mid T.\ell : \mathit{Ind}\} \rightharpoonup \{h.id \mid h \in H\}$. Running BUILDTAKES with an empty inital label map produces the set of all takes on $H$ (records) that instantiate the given type.

The algorithm itself (see Algorithm 1) is not particularly important. Indeed, naive tree search is probably not cognitively plausible since the complexity scales exponentially with the number of objects in the scene, and there are many ways to make the search more efficient. What is of note here is the relationship between the H-data and the candidate situation takes, which are being built-up and structured according to the interpretation of the question. To satisfy the pre-conditions of the question (i.e., to find a witness for $q$.bg and candidate argument to $q$.clfr), it is not enough to use generic perceptual input; instead we must build up (or structure) a *take* on the situation based on high-level perceptual data and the question itself. Only then can the situation take be used to answer the question.

## 5 Adapting the general account to the CLEVR dataset

Everything we have said above is independent of the type of H-data. In this section, we will show a concrete instance of this general framework, where H-data is a set of CLEVR scene graphs. This is intended as an illustration and proof-of-concept only, showing an example of how the general TTR account applies to a specific type of H-data and classification method.

### 5.1 The CLEVR dataset

CLEVR (Johnson et al., 2016) is a synthetic dataset of programatically rendered 3d scenes of geometric objects and template-generated questions about them. Designed as a diagnostic dataset for visual and linguistic compositional reasoning, it is drawn from an extremely constrained visual and linguistic domain domain. Objects are drawn from limited

**Algorithm 1** Recursive tree search for finding situation takes that satisfy a type from given H-data. The function EXTRACTTAKE produces a record that represents all objects in $H$, labeled according to $\mathcal{L}$. Objects from $H$ that don't appear in the image of $\mathcal{L}$ are labeled with auxiliary labels that don't appear in $T$.

**Require:** $\mathcal{L} : \{\ell \in labels(T) \mid T.\ell : Ind\} \rightharpoonup \{h.id \mid h \in H\}$

**function** BUILDTAKES($T, H, \mathcal{L}$)
  $U \leftarrow labels(T) \setminus Dom(\mathcal{L})$
  $s \leftarrow$ EXTRACTTAKE($\mathcal{L}, H$)
  **if** $U = \varnothing$ **then**
    **if** $s : T$ **then**
      **return** $\{s\}$
    **else**
      **return** $\varnothing$
  **else**
    **if** $s : T \restriction Dom(\mathcal{L})$ **then**
      $l \leftarrow Pop(U)$
      $\mathcal{L}'_h \leftarrow \mathcal{L} \cup \{\langle l, h.\mathrm{id}\rangle\}$
      **return** $\cup\{$BUILDTAKES$(T, H, \mathcal{L}'_h) \mid h \in H\}$
    **else**
      **return** $\varnothing$

number of geometric solids and have a finite range of colors, sizes, and textures. Because the images are generated from code, it is possible to use image metadata to design train/test splits that probe for compositional representation by testing how well a model generalizes to novel situations.

Since the dataset is programatically generated, it is possible to produce "ground truth" semantics for the questions and schematic descriptions of the image that are sufficient for answering the questions. In this work, we directly use these schematic question and scene descriptions as input to validate that the symbolic component of our system has the inferential and representational power required for the task. This ensures that the performance of an eventual hypbrid system is reflective of the distributed representations and their interaction with the symbolic components, and not theoretical limitations.

Ground truth scene representations are presented as a *scene graph*, whose nodes are objects, annotated with attributes like color and shape (one-place predicates), and whose edges represent binary spatial relations between pairs of objects. Questions are represented as a functional programs that operate on those graphs.

### 5.2 Using CLEVR for H-data

In general, classification of visual scenes may involve e.g. processing of visual information in deep neural networks. However, in CLEVR the visual scenes are already annotated with formal structures that we will exploit when defining the classifiers required. In effect, we will treat the annotations as H-data, despite not actually being the result of processing of L-data. The classifiers themselves will be rather trivial, and they are meant only as an illustration of how the general account can be adapted to a specific dataset.

Assume we have as a scene graph $g$, consisting of a list of object descriptions that in turn consist of feature-value pairs. The original CLEVR data relies on the order of the items. To avoid this, we massage the data so that it includes a field `'id'` to avoid reliance on order:

```
[{'id': 'h0',
  '3d_coords': [2.408, -2.868, 0.350],
  'color': 'green',
  'material': 'metal',
  'pixel_coords': [213, 257, 7.728],
  'rotation': 233.806,
  'shape': 'cylinder',
  'size': 'small'
  'behind': ['h1', 'h2', 'h3',...],
  'front': [ ],
  'left': ['h2', 'h3', ...],
  'right': ['h5', ...]},
 {'id:'h1',
```

```
  '3d_coords': [1.649, -1.450, 0.350],
  'color': 'yellow',
  'material': 'metal',
  'pixel_coords': [246, 201, 9.081],
  'rotation': 140.802,
  'shape': 'cube',
  'size': 'small'
  'behind': ['h2', 'h3', ...],
  'front': ['h0'],
  'left': ['h2', 'h3', ...],
  'right': ['h5', ...]},
]
```

We can transform $g$ into H-data $H_s$:

(26) $H_s$=[ [id =a, data= {'id': 'h0, '3d_coords':...}], [id =b, data={'id': 'h1', '3d_coords':...}] ]

From $H_s$ and a background type $T_{bg}$, we can construct a witness take as described above in Sections 4.6 and 4.7.

As mentioned above, we are using H-data as evidence/proofs of types, and judging if some H-data is of a ptype is done using classifiers. In the case where H-data are scene graphs, the classification becomes rather trivial as the required information is already available in the H-data. For example, a colour classifier might be specified as follows:

(27) $\kappa_{\text{col}} = \lambda h : \langle Hdata \rangle$.
$\begin{cases} \text{RED if } h(1)\texttt{['color'] == 'red'} \\ \text{BLUE if } h(1)\texttt{['color'] == 'blue'} \\ \ldots \end{cases}$

For a two-place predicate:

(28) $\kappa_{\text{col}} = \lambda h : \langle Hdata, Hdata \rangle$.
$\begin{cases} \text{LEFT if } h(2)\texttt{.['id'] in } h(1)\texttt{['left']} \\ \text{RIGHT if } h(2)\texttt{.['id'] in } h(1)\texttt{['right']} \end{cases}$

Note that we are here mixing TTR notation with `Python code`. This is specific to the type of H-data we are using and a particular way of accessing it. For other types of H-data, the classifier definition may look quite different. However, we have set things up so that the dependence on H-data type has been isolated to the classifier definitions.

## 6 Summary and future work

We outlined a formal account of VQA where an agent perceives a real world situation by receiving low-level perceptual data, processes that data into high-level individuated perceptual data, structures this data in light of the semantics of a linguistic expression (a question), and then classifies the structured data to arrive at an answer to the question. Importantly (although not directly in scope of this paper), word meanings in general, including perceptual meaning aspects modeled as classifiers, are dynamic and subject to learning and coordination between dialogue participants (Larsson, 2015; Larsson et al., 2021; Larsson, 2021).

In future work, we aim to apply and extend the account to also encompass semantic pointers (Eliasmith, 2015; Larsson et al., 2025).

## 7 Limitations

There are many ways in which the work presented here could be extended. We only provide examples for the CLEVR dataset, but to concretely show the general applicability of the proposed account, we want to provide examples a large and diverse range of datasets. Furthermore, whereas the current examples focus on individual object classification, we would like to offer a more detailed explanation of how classification of images in terms of complex spatial relations. Another limitation is that the account does not yet cover classification of image sequences in terms of events that are extended in time. Finally, classification is in general probabilistic, and we would like to employ probabilistic TTR (Larsson, 2020a)to extend the current account accordingly.


## Acknowledgments

This work was supported by grants from the Swedish Research Council VR project 2025-04592, Semantic Pointers in Natural Language Semantics (SPiNLS) and VR project 2014-39 for the establishment of the Centre for Linguistic Theory and Studies in Probability (CLASP) at the University of Gothenburg.


## References


Aishwarya Agrawal, Dhruv Batra, and Devi Parikh. 2016. Analyzing the Behavior of Visual Question Answering Models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1955–1960, Austin, Texas. Association for Computational Linguistics.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. 2015. VQA: Visual Question Answering. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2425–2433, Santiago, Chile. IEEE.

Daisuke Bekki, Ribeka Tanaka, and Yuta Takahashi. 2021. Integrating Deep Neural Network with Dependent Type Semantics. In *Proceedings of the Symposium Logic and Algorithms in Computational Linguistics 2021 (LACompLing2021) 2021 (LACompLing2021)*, page 37, Stockholm University.

Ellen Breitholtz. 2020. *Enthymemes and Topoi in Dialogue: The Use of Common Sense Reasoning in Conversation*, volume 41 of *Current Research in the Semantics/Pragmatics Interface*. Brill, Leiden/Boston.

Robin Cooper. 2005. Austinian truth, attitudes and type theory. *Research on Language and Computation*, 3:333–362.

Robin Cooper. 2012. Type theory and semantics in flux. In Ruth Kempson, Nicholas Asher, and Tim Fernando, editors, *Handbook of the Philosophy of Science*, volume 14: Philosophy of Linguistics. Elsevier BV. General editors: Dov M. Gabbay, Paul Thagard and John Woods.

Robin Cooper. 2013. Clarification and Generalized Quantifiers. *Dialogue and Discourse*, 4(1):1–25.

Robin Cooper. 2023. *From Perception to Communication: a Theory of Types for Action and Meaning*. Oxford University Press.

Robin Cooper and Jonathan Ginzburg. 2015. Type theory with records for natural language semantics. *The handbook of contemporary semantic theory*, pages 375–407.

Simon Dobnik and Robin Cooper. 2017. Interfacing language, spatial perception and cognition in type theory with records. *Journal of Language Modelling*, 5(2):273–301.

Chris Eliasmith. 2015. *How to Build a Brain: A Neural Architecture for Biological Cognition*. Oxford Series on Cognitive Models and Architectures. Oxford Univ. Press.

Yash Goyal, Tejas Khot, Aishwarya Agrawal, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2019. Making the V in VQA Matter: Elevating the Role of Image Understanding in Visual Question Answering. *Int. J. Comput. Vision*, 127(4):398–414.

Drew A. Hudson and Christopher D. Manning. 2018. Compositional Attention Networks for Machine Reasoning. In *ICLR*. arXiv. ArXiv:1803.03067 [cs.AI].

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross Girshick. 2016. CLEVR: A Diagnostic Dataset for Compositional Language and Elementary Visual Reasoning. *arXiv preprint*. ArXiv:1612.06890 [cs].

Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Judy Hoffman, Li Fei-Fei, C. Lawrence Zitnick, and Ross Girshick. 2017. Inferring and Executing Programs for Visual Reasoning. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3008–3017. ISSN: 2380-7504.

Staffan Larsson. 2011. The ttr perceptron: Dynamic perceptual meanings and semantic coordination. In *Proceedings of the 15th Workshop on the Semantics and Pragmatics of Dialogue (SemDial 2011)*, Los Angeles (USA). Institute for Creative Technologies.

Staffan Larsson. 2013. Formal semantics for perceptual classification. *Journal of Logic and Computation*.

Staffan Larsson. 2015. Formal semantics for perceptual classification. *Journal of Logic and Computation*, 25(2):335–369. Published online 2013-12-18.

Staffan Larsson. 2018. Perceptual semantics and dialogue processing. In *Proceedings of the Workshop on Dialogue and Perception*.

Staffan Larsson. 2020a. Discrete and probabilistic classifier-based semantics. In *Proceedings of the Probability and Meaning Conference (PaM 2020)*, pages 62–68, Gothenburg. Association for Computational Linguistics.

Staffan Larsson. 2020b. Extensions are indeterminate if intensions are classifiers. In *Proceedings of the 24th Workshop on the Semantics and Pragmatics of Dialogue–Full Papers, Virtually at Brandeis, Waltham, New Jersey. SEMDIAL.*

Staffan Larsson. 2021. The role of definitions in coordinating on perceptual meanings. In *Proceedings of the Workshop on the Semantics and Pragmatics of Dialogue (SemDial 2021)*.

Staffan Larsson, Jean-Philippe Bernardy, and Robin Cooper. 2021. Semantic learning in a probabilistic type theory with records. In *Proceedings of Workshop on Computing Semantics with Types, Frames and Related Structures 2021*.

Staffan Larsson and Robin Cooper. 2009. Towards a formal view of corrective feedback. In *Proceedings of the EACL 2009 Workshop on Cognitive Aspects of Computational Language Acquisition*.

Staffan Larsson and Robin Cooper. 2021. Bayesian classification and inference in a probabilistic type theory with records. In *Proceedings of NALOMA 2021*.

Staffan Larsson, Jonathan Ginzburg, Robin Cooper, and Andy Lücking. 2025. Finding answers to questions: Bridging between type-based and computational neuroscience approaches. In *Proceedings of the 16th international conference on computational semantics*, pages 118–126.

Yinhong Liu and Guy Emerson. 2022. Learning Functional Distributional Semantics with Visual Data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3976–3988, Dublin, Ireland. Association for Computational Linguistics.

Andy Lücking and Jonathan Ginzburg. 2022. Referential transparency as the proper treatment for quantification. *Semantics and Pragmatics*, 15(4).

Vladislav Maraev, Ellen Breitholtz, Christine Howes, Staffan Larsson, and Robin Cooper. 2021. Something Old, Something New, Something Borrowed, Something Taboo: Interaction and Creativity in Humour. *Frontiers in Psychology*, 12.

Arild Matsson, Simon Dobnik, and Staffan Larsson. 2019. Imagettr: Grounding type theory with records in image classification for visual question answering. In *Proceedings of the IWCS 2019 Workshop on Computing Semantics with Types, Frames and Related Structures*, pages 55–64.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788.

Ronja Utescher. 2019. Visual ttr-modelling visual question answering in type theory with records. In *Proceedings of the 13th International Conference on Computational Semantics-Student Papers*, pages 9–14.

Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Josh Tenenbaum. 2018. Neural-Symbolic VQA: Disentangling Reasoning from Vision and Language Understanding. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Peng Zhang, Yash Goyal, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2016. Yin and Yang: Balancing and Answering Binary Visual Questions. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5014–5022. ISSN: 1063-6919.