A APPENDIX

A.1 TASKS DETAILS

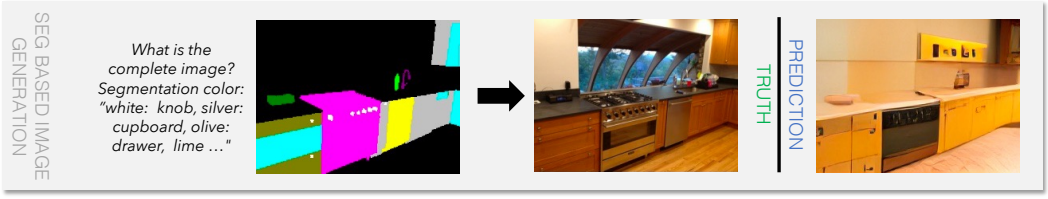
UNIFIED-IO is jointly trained on a large and diverse set of vision, language and vision & language tasks. In this section, we describe these tasks in detail and show the prompts we use during training and inference (text on the left of example cards). We also provide qualitative examples of both the ground truth and the predictions made by UNIFIED-IO.

A.1.1 IMAGE SYNTHESIS TASKS

Image Synthesis from Text. This task requires generating an image that matches a sentence. Training data comes from 4 captioning datasets: COCO Caption (Chen et al., 2015), Conceptual Captions 3M and 12M (Changpinyo et al., 2021), and RedCaps (Desai et al., 2021) as well datasets used for image classification using the object class as the input caption. Specialized image generation models like DALL·E 2 (Ramesh et al., 2022) use an order of magnitude more data, but we limit our sources to these sets for training efficiency.

Image Inpainting. This task requires filling in a region of an image with a target object. Training data for this task is built from object bounding box annotations from Open Images (Kuznetsova et al., 2020), Visual Genome (Krishna et al., 2017) and COCO (Lin et al., 2014). For each object, the input image becomes the source image with the object's bounding box blanked out. The input prompt provides the bounding box's location and the target category. The target output is the original image.

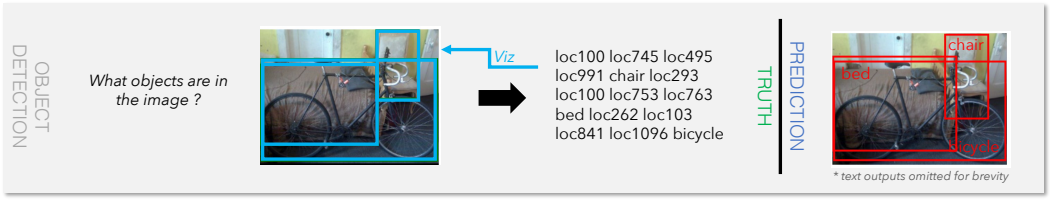
Image Synthesis from Segmentation. This task involves generating an image that matches an input semantic segmentation, i.e., a set of class labels for some or all of the pixels in the image. UNIFIED-IO is trained for this task using segmentation annotations from COCO (Lin et al., 2014), Open Images (Kuznetsova et al., 2020), and LVIS (Gupta et al., 2019) as input. Following the method from Section 3.1 the segmentation input is converted into a RGB image paired with a prompt listing the color-to-class mapping, and the target output is the source image.



A.1.2 SPARSE LABELLING TASKS

Object Detection. UNIFIED-IO is trained on object detection annotations from Visual Genome, Open Images, and COCO. For this task the input is a static prompt and an image, and the output

text includes the bounding boxes and class names of all objects in the image. We randomize the order of the output objects during training, but for simplicity leave integrating more complex data-augmentation techniques (Chen et al., 2022b) to future work.



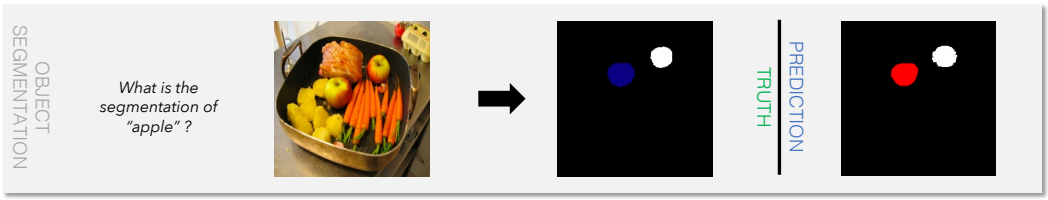
Object Localization. Object localization requires returning bounding boxes around all objects of a given category. Training data is derived from our object detection training data by constructing a training example from each category of objects present in an image. The input is then the image, a prompt specifying the target class, and the output is a list of all boxes that contain an instance of that class. The class for each box (which is always the class specified in the prompt) is included in the output for the sake of keeping the output format consistent with the object detection output. Object localization can use input categories which are not present in the image. To handle this, we construct negative samples by randomly selecting categories not present in the image to use as input, in which case the output is an empty sequence.

Referring Expression Comprehension. The task requires the model to localize an image region described by a natural language expression. The annotation is similar to Object Localization, except that the target is specified with natural language expression instead of class name. Datasets for this task include RefCOCO (Kazemzadeh et al., 2014), RefCOCO+ (Kazemzadeh et al., 2014) and RefCOCOg (Mao et al., 2016).

Keypoint Estimation. Keypoint estimation requires returning the location of 17 keypoints on a human body (e.g., eyes, nose, feet, etc.) for each person in an image. While it is possible to perform this task in one pass by listing the keypoints of all people in the image in a single output sequence, this can result in an extremely long output sequence, so UNIFIED-IO uses a multi-step approach instead. To do this UNIFIED-IO is trained to complete the subtask of detecting the keypoints for single a person in a given region. For this subtask, the input prompt specifies the target region and and the output is a list of 17 points (a pair of locations tokens for the x and y coordinates) along with a visibility labels (1 for not visible, 2 for partly visible, 3 for fully visible). Non-visible points are preceded by two copies of a new special tokens that indicate there are no valid coordinates. The keypoint metric does not award points for correctly identifying non-visible points, so during inference we mask that special token so the model makes a best-effort guess for the coordinates of every single point. Training data for this subtask comes from COCO human pose data (Lin et al., 2014) with the ground-truth person regions as input. During inference we locate person regions using the object localization prompt, then apply UNIFIED-IO again to find keypoints for each detected region.

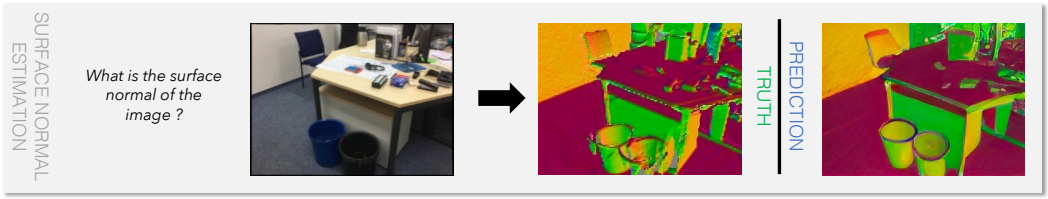
A.1.3 Dense Labelling Tasks

Object Segmentation. Object segmentation requires finding the binary segmentation mask of each instance of a particular category in an image. The input is an image and a prompt that includes the target class, while the output is an RGB image with black background and instances of that class filled in with unique colors following the method in Section 3.1. The output image is resized to match the input image if needed using a nearest-neighbor resizing method, and binary masks are built from each unique color. In practice the output image from UNIFIED-IO can have slightly non-uniform colors or extraneous background pixels, likely due to limitation in what the D-VAE can decode/encode, so the output pixels are clustered by color and and connected components of less than 8 pixels are removed to build cleaned instance masks. Segmentation annotations come from Open Images LVIS, and COCO.



Depth Estimation. Depth estimation requires assigning each pixel in an image a depth value. This task uses a static prompt as input, and the output is a grayscale image representing the normalized depth at each pixel. The generated output image is reiszed to the same size as the input image and then pixel values are rescaled to the maximum depth in the training to get an output depth map. Training data comes from the NYU Depth Dataset V2 (Nathan Silberman & Fergus, 2012).

Surface Normal Estimation. UNIFIED-IO is trained on FrameNet (Huang et al., 2019a) and BlendedMVS (Yao et al., 2020) surface normal estimation datasets. For this task the input is a static prompt and an image and the output is an RGB representation of the $x/y/z$ orientation of the surface at each pixel. The generated output image is resized to match the input image and converted back to $x/y/z$ orientations to produce the final output.



A.1.4 IMAGE CLASSIFICATION TASKS

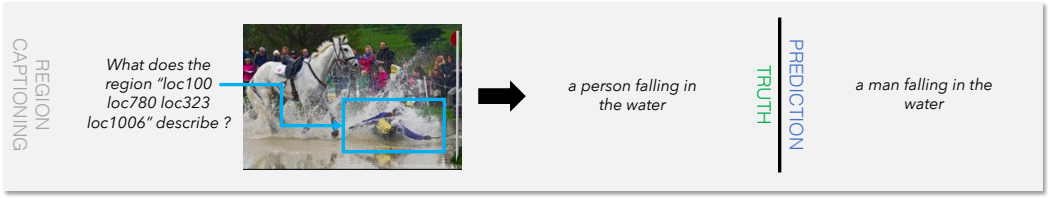
Image Classification. UNIFIED-IO is trained on 6 image classification datasets: ImageNet 2012 (Deng et al., 2009), ImageNet21k (Ridnik et al., 2021), Places365 (Zhou et al., 2017), Sun397 (Xiao et al., 2010), iNaturalist (Van Horn et al., 2018) and Caltech birds 2011 (Welinder et al., 2010). For this task the input is an image and a static prompt, and the output is a class name. During inference we compute the log-probability of each class label in the dataset being evaluated and return the highest scoring one. This ensures UNIFIED-IO does not return a category from a different categorization dataset that is a synonym or hypernym of the correct label.

Object Categorization. This task identifies which label, from a given set, best corresponds to an image region defined by an input image and bounding box. The input is the image, a prompt specifying the image region and the output is the target class name. We convert object detection annotations from Visual Genome, Open Images, and COCO for this task. Inference is constrained to return a valid label for the target label set just as with image classification.

A.1.5 IMAGE CAPTIONING TASKS

Image Captioning. Image captioning data comes from the same manually annotated and unsupervised sources used for Image Generation. In this case the inputs and output are reversed, the input is an image and the static prompt, and the output is a caption that matches the image.

Region Captioning. Region captioning tasks a model with generating a caption that describes a specific region in the image. Our format for this task is identical to Image Captioning except the region is included in the input prompt. Visual Genome (Krishna et al., 2017) is used for the training data.

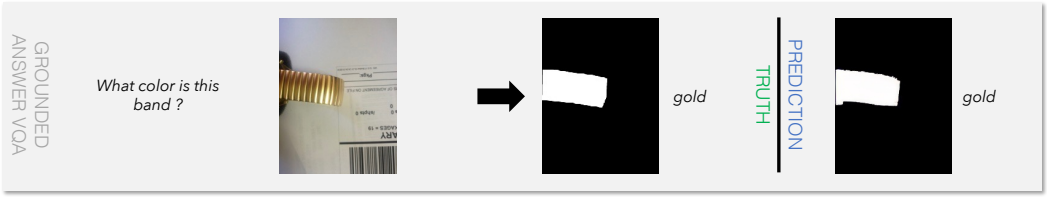


A.1.6 VISION & LANGUAGE TASKS

Visual Question Answering. UNIFIED-IO is trained on a collection of VQA datasets including VQA 2.0 (Goyal et al., 2017), Visual Genome, VizWizVQA (Gurari et al., 2018), OKVQA (Marino et al., 2019) and A-OKVQA (Schwenk et al., 2022). For VQA, the question is used as the prompt, and the output is the answer text. For VQA, it is common to constrain the model to predict an answer from a fixed last of common VQA answers (Wang et al., 2022b;d) during inference, but we avoid doing this since we find it does not benefit UNIFIED-IO in practice.

We additionally convert data from several other datasets in a VQA format, including imSitu (Yatskar et al., 2016), where we treat predicting the verb and then the related slots as separate VQA questions, VisualCOMMET (Park et al., 2020) where we convert the before/after/intent into questions by converting the input regions into location tokens, SNLI-VE (Xie et al., 2019) where we integrate the entailed text into an input question, and VCR (Zellers et al., 2019a) where we again integrate the input regions into the prompt by encoding them with location tokens and integrate the rationales into the target text for the answer justification task.

Answer-Grounded Visual Question Answering. This task requires both answering a question and returning a binary mask specifying the region of the image used to answer the question. The format for this task follows the one for VQA except that a binary mask is also used as an additional output. Training data comes from VizWiz-VQA (Chen et al., 2022a), a dataset designed to train models that could benefit people with visual impairments.

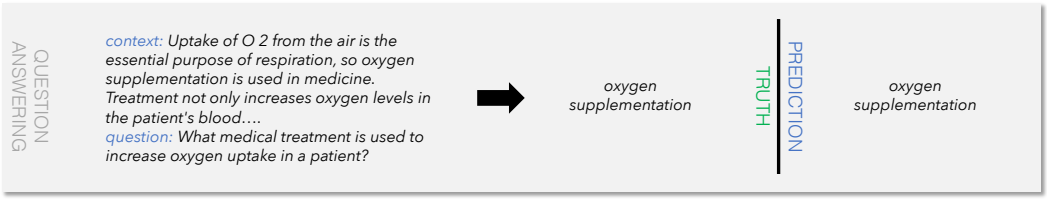


Relationship Detection. This task requires predicating a relationship between a pair of objects which are grounded by bounding boxes. The prompt contains both the object regions, and the output is the predicted predicate. There are 2 datasets in this tasks: Visual Genome (Krishna et al., 2017) and Open Images (Kuznetsova et al., 2020).

A.1.7 NATURAL LANGUAGE PROCESSING TASKS

Question Answering. Following prior work in natural language processing (Raffel et al., 2020), QA tasks are formatted by placing both the question and any text context (e.g., an paragraph containing the answer) into the prompt and training the model to generate the text answer. UNIFIED-IO is trained on several QA datasets including SQuAD 2.0 (Rajpurkar et al., 2016), other training datasets from the MRQA (Fisch et al., 2019) shared tasks (Trischler et al., 2017; Joshi et al., 2017; Dunn et al., 2017; Yang et al., 2018; Kwiatkowski et al., 2019), QA datasets from SuperGLUE (Wang

et al., 2019; Clark et al., 2019; Khashabi et al., 2018; Roemmele et al., 2011), Cosmos QA (Huang et al., 2019b), OpenBookQA (Mihaylov et al., 2018), and HellaSwag (Zellers et al., 2019b). If the text context is longer then our maximum sequence length we use a sliding-window approach following Devlin et al. (2019) which exposes the model to different windows of text from the context and returns the highest-confidence answer.

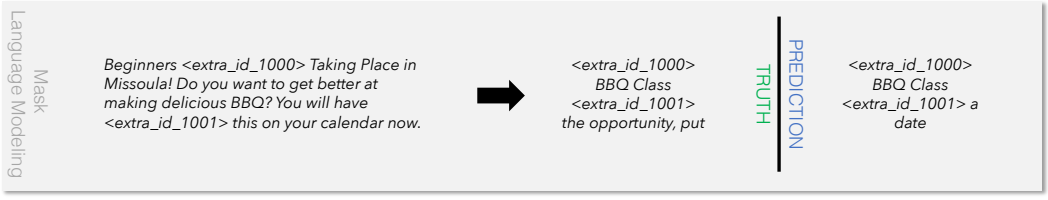


Text Classification. Also following past work (Raffel et al., 2020), text classification tasks are formatted by placing the input sentences and a query in the prompt and training the model to generate the target class. Datasets include tasks from GLUE and SuperGLUE (Wang et al., 2018; 2019; Warstadt et al., 2018; Socher et al., 2013; Dolan & Brockett, 2005; Iyer et al., 2017; Cer et al., 2017; Williams et al., 2018; Dagan et al., 2005; Bar-Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009; Levesque et al., 2012; Williams et al., 2018; De Marneff et al., 2019; Pilehvar & os'e Camacho-Collados, 2018), as well as SNLI (Bowman et al., 2015), SciTail (Khot et al., 2018), IMDB Reviews (Maas et al., 2011), and PAWS (Zhang et al., 2019).

Text Summerization. Text summarization is done again by providing the input paragraph and a prompt as input and generating a summary as output. We use the Gigaword dataset (Graff et al., 2003; Rush et al., 2015) for training data.

A.1.8 Language Modeling Tasks

Mask Language Modeling. Following T5 (Raffel et al., 2020), the mask language modelling objective randomly samples and then drops out 15% of tokens in the input sequence. All consecutive spans of dropped-out tokens are replaced by a single sentinel token. The target is to recover the dropped tokens given the sentinel token. We use C4 (Raffel et al., 2020) and Wikipedia (Foundation) datasets.



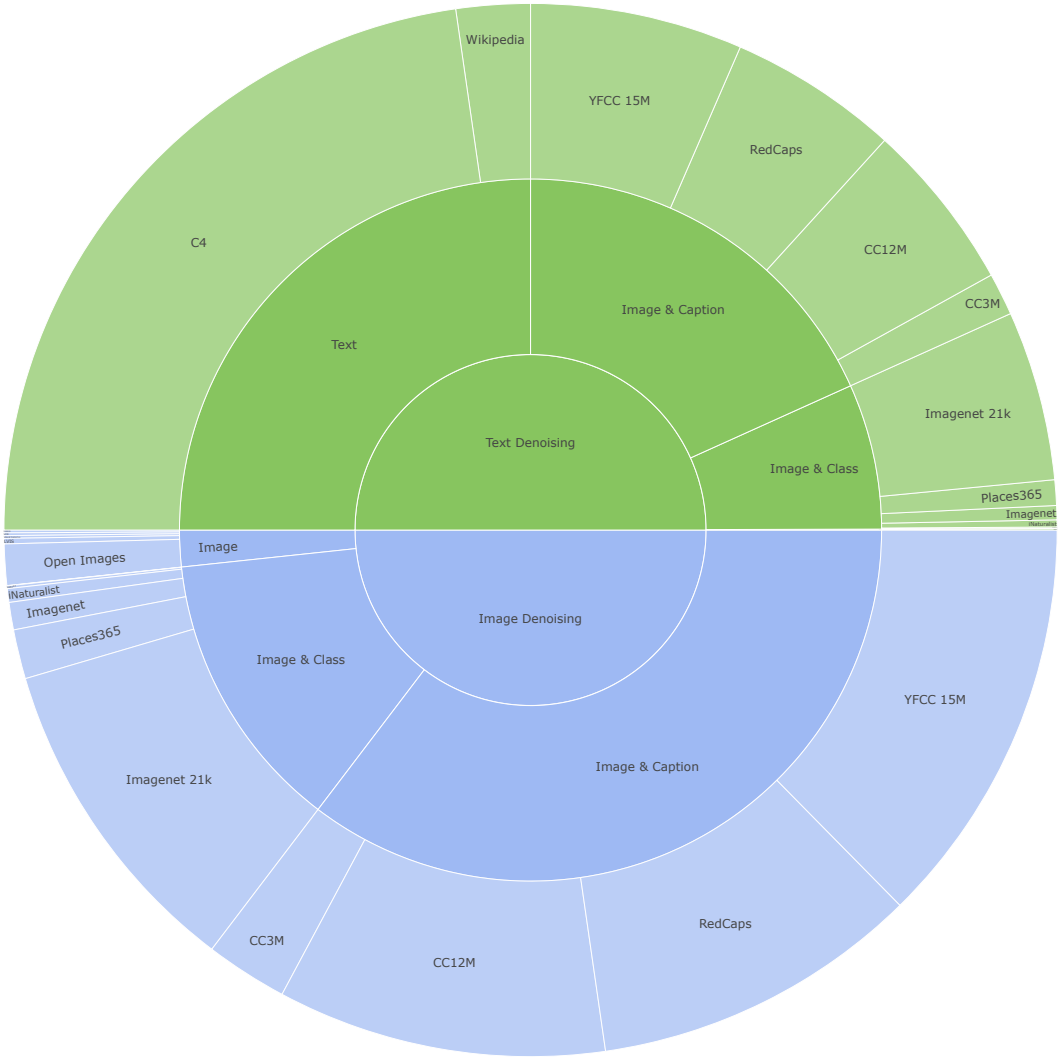


Figure 3: Pre-training objectives (inner circle), annotation types (middle circle) and datasets (outer circle) used in pre-training of UNIFIED-IO. Sizes correspond to the sampling rate in the training distribution. Best viewed in color.

A.2 PRE-TRAINING DATA DISTRIBUTION

Figure 3 shows a visualization of pre-training data distribution used by UNIFIED-IO. As discussed in Section 3.3, we equally sample data with the text denoising and image denoising objective (inner circle of Figure 3). For text denoising, half of the samples are from pure text data, *i.e.* C4 and Wikipedia. The other half is constructed from image and class, such as Imagenet21k (Ridnik et al., 2021) or image and caption, such as YFCC15M (Radford et al., 2021). For image denoising, we use the text information when class and caption are present in the data source and sample the dataset proportional to the dataset size. For both text and image denoising, we random drop both modalities with 10% of the time if both text and image as inputs.

Figure 4: Task groups (inner circle), tasks (middle circle) and datasets (outer circle) used in multi-task training of UNIFIED-IO. Sizes correspond to the sampling rate in the training distribution. Best viewed in color.

A.3 MULTI-TASKING DATA DISTRIBUTION

Figure 4 shows a visualization of the multi-task training distribution used by UNIFIED-IO from Table 1. As discussed in Section 3.3, we equally sample each group ($1/8$) except image synthesis ($3/16$) and dense labeling ($1/16$) since dense labeling has a much smaller sample size compared to image synthesis. We sample tasks and datasets (middle and outer circle) with a temperature-scaled mixing strategy to make sure the model is sufficiently exposed to underrepresented tasks. We raise each task's mixing rate to the power of $1/T$ and then renormalize the rates so that they sum to 1. Following Raffel et al. (2020), we use $T = 2$ in our experiments.

Due to the large variance in dataset size, some of the tasks are rarely sampled. For example, the depth estimation task only has the NYU Depth dataset source (Nathan Silberman & Fergus, 2012) and thus the sampling rate is only 0.43%. However, the model still works well for depth estimation tasks, even outperforming concurrent work (Kolesnikov et al., 2022) (0.385 *vs*. 0.467 RMSE). We suspect the large model capacity and masked image denoising pre-training improves the performance. Similarly, Grounding VQA (Chen et al., 2022a) has 0.15% sample rate, but the model can still achieve state-of-the-art performance on this task partly because it is trained on many related datasets for VQA and segmentation.

A.4 IMPLEMENTATION DETAILS

The total vocabulary size is 49536, with 32152 language tokens, 1000 location tokens, and 16384 vision tokens. During training, we random sub-sample 128 image patches for pre-training state and 256 image patches (out of 576) for multi-task stage. We do not use dropout. Adafactor (Shazeer & Stern, 2018) optimizer is used to save memory. We use a learning rate of 10^{-2} for the first 10,000 steps and then decay at a rate of $1/\sqrt{k}$. We train with $\beta_1 = 0.9$ and $\beta_2 = 1.0 - k^{-0.8}$, where k is the step number. We use global norm gradient clipping with 1.0 and find this is crucial to stabilized `XL` training. We train the `Small`, `Base` and `Large` with batch size of 2048 and `XL` with batch size of 1024 due to memory consideration. 4-way in-layer parallelism and 128-way data parallelism used to scale the 3B model training. For all models, we train $1000k$ steps – $500k$ for pre-training and multi-task training respectively.

A.5 EVALUATION ON SAME CONCEPT AND NEW CONCEPT

		restricted	params (M)	Categorization		Localization		VQA		Refexp		Segmentation		Keypoint		Normal	
				same	new	same	new	same	new	same	new	same	new	same	new	same	new
0	NLL-AngMF	✓	72	-	-	-	-	-	-	-	-	-	-	-	-	**50.7**	-
1	Mask R-CNN	✓	58	-	-	51.9	40.8	-	-	-	-	44.9	0.3	**70.9**	-	-	-
2	GPV-1	✓	236	58.7	0.8	48.3	37.8	58.4	74.0	29.7	23.1	-	-	-	-	-	-
3	CLIP		302	49.1	46.7	-	-	-	-	-	-	-	-	-	-	-	-
4	$\text{OFA}_{\text{LARGE}}$		473	28.9	15.8	-	-	74.9	88.6	63.4	58.5	-	-	-	-	-	-
5	GPV-2		370	**85.0**	13.5	54.6	54.2	69.8	81.7	57.8	48.3	-	-	-	-	-	-
6	$\text{UNIFIED-IO}_{\text{SMALL}}$		71	52.9	31.9	47.5	61.5	59.0	72.5	54.2	45.7	37.4	48.5	46.6	-	33.6	-
7	$\text{UNIFIED-IO}_{\text{BASE}}$		241	60.3	47.5	57.9	68.4	68.0	81.8	72.5	62.2	45.8	57.2	60.2	-	37.7	-
8	$\text{UNIFIED-IO}_{\text{LARGE}}$		776	63.0	52.7	63.3	70.9	72.1	84.3	79.2	66.3	50.4	62.2	67.7	-	40.3	-
9	$\text{UNIFIED-IO}_{\text{XL}}$		2925	66.1	**60.1**	**65.6**	**74.4**	**78.6**	**90.2**	**83.5**	**72.4**	**53.0**	**64.2**	68.2	-	45.1	-

Table 6: Generalization to new concepts on the GRIT ablation set.

GRIT provides a breakdown of metrics into two groups: *same* for samples that only contain concepts seen in the primary training data (a set of common datasets like COCO, ImageNet and Visual Genome), and *new* for samples containing at least one concept unseen in primary training data. Table 6 shows results for UNIFIED-IO and other leaderboard entries for the ablation set, divided into same and new concepts.

$\text{UNIFIED-IO}_{\text{XL}}$ shows little degradation in performance between *same* and *new*, compared to competing entries. On some tasks UNIFIED-IO is even able to outperform on the *new* split compared to the *same*. This indicates that the volume of training data used to train UNIFIED-IO has a broad coverage of concepts, and provides almost as effective a level of supervision as provided by large standard vision datasets like COCO. Furthermore, since UNIFIED-IO is a uniquely unified architecture with no task-specific parameters, it is very likely able to effectively transfer knowledge across different tasks.

In comparison to Mask-RCNN (row 1), GRIT metrics show UNIFIED-IO (row 14) is better by a large margin on *new* concepts, i.e., non-COCO examples (74.4 vs 40.8 for localization and 64.2 vs 0.3 on segmentation), but is still superior on the COCO-like examples (65.6 vs 51.9 for localization and 53.0 vs 44.9 on segmentation). UNIFIED-IO is also able to beat GPV-2 (row 5) on *new* concepts by large margins across all 4 tasks supported by GPV-2 even though GPV-2 is exposed to these concepts via webly supervised data and is designed to transfer concept knowledge across skills.

A.6 PROMPT GENERALIZATION CASE STUDY

To better understand how different prompts affect UNIFIED-IO, we do a case study on referring expressions. In particular, we re-evaluate UNIFIED-IO on the GRIT referring expression ablation set while replacing the prompt used during training (first row in the table) with a paraphrase (following rows). Results are shown in Table 7.

Overall, we find that the model has some capacity to generalize to paraphrases of the prompt (*e.g.*, row 3 works reasonably well despite using completely different words), but there are paraphrases that result in very significant performance decrease (*e.g.* rows 5, 6, and 8). We also find removing

	Prompt	Refexp Score
0	Which region does the text " REFEXP " describe ?	78.9
1	Which region does the text "REFEXP" describe?	76.7
2	Which region matches the text " REFEXP " ?	77.4
3	Locate the " REFEXP " .	65.6
4	Which region can be described as " REFEXP " ?	64.8
5	Locate the region described by " REFEXP " .	43.2
6	Where is the " REFEXP " ?	41.5
7	Where is the "REFEXP"?	0.1

Table 7: Case study on GRIT referring expressions using different prompts. The first prompt is the one used during training, the others are paraphrases. REFEXP is replaced by the referring expression text of individual examples during evaluation.

the spaces around the punctuation sometimes results in minor regressions (row 0 vs row 1) and sometimes in sharply reduced performance (row 6 vs row 7), showing UNIFIED-IO can be sensitive to formatting details. We hypothesize that this caused by the SentencePiece tokenizer changing the tokenization of the referring expressing if the quotes are not separated from it by spaces. Building multi-task models that can generalize to different prompts, and ideally to prompts for completely new tasks, is an exciting avenue for future work.

A.7 QUALITATIVE EXAMPLES

Here we present qualitative examples of predictions from UNIFIED-IO for all training tasks. For brevity, if prompts are identical for each example we only show the prompt once, and if the prompt follows the same template for each example we show the template with parts that would be substituted with different words or location tokens underlined, and then show just the substitution with individual examples.

A.8 OTHER RELATED WORK

Other Modalities. Multi-modal models for video (Li et al., 2022c;a; Wang et al., 2022a; Alayrac et al., 2022; Zellers et al., 2021; Yu et al., 2022a), audio (Zellers et al., 2022; Jaegle et al., 2022), and other modalities including game-playing and robot controlling (Reed et al., 2022; Jaegle et al., 2022; Liang et al., 2022) have also been studied. Integrating these modalities is an important line of research, however existing models often do not even support sparse structured output, and do not support dense structured outputs, so they do not meet our objective of supporting classic vision tasks.

Vision & Language Pre-Training. Vision and language pre-training has become standard practice for multi-modal models, including both unified models and non-unified models that require task-specific heads to trained from scratch during fine-tuning. Many initial pre-training strategies were inspired by BERT (Devlin et al., 2019) and included masked-language-modeling, image-text-matching, or mask-region-modeling objectives, often supplemented with objectives using the predictions of a strong object detector model (e.g, VILBERT (Lu et al., 2019), LXMERT (Tan & Bansal, 2019), VisualBERT (Li et al., 2019)). More recently contrastive-image-text losses (Radford et al., 2021; Li et al., 2022b; 2021) or auto-regressive generation losses (Wang et al., 2022d;a; Yu et al., 2022a), have become common. Several works have also directly used object detection or segmentation datasets for pre-training Yuan et al. (2021); Wang et al. (2022b); Sun et al. (2022). The generalized masked-data-modeling objective used in UNIFIED-IO is similar to ones used is several recent works (Wang et al., 2022c; Peng et al., 2022; Singh et al., 2022).

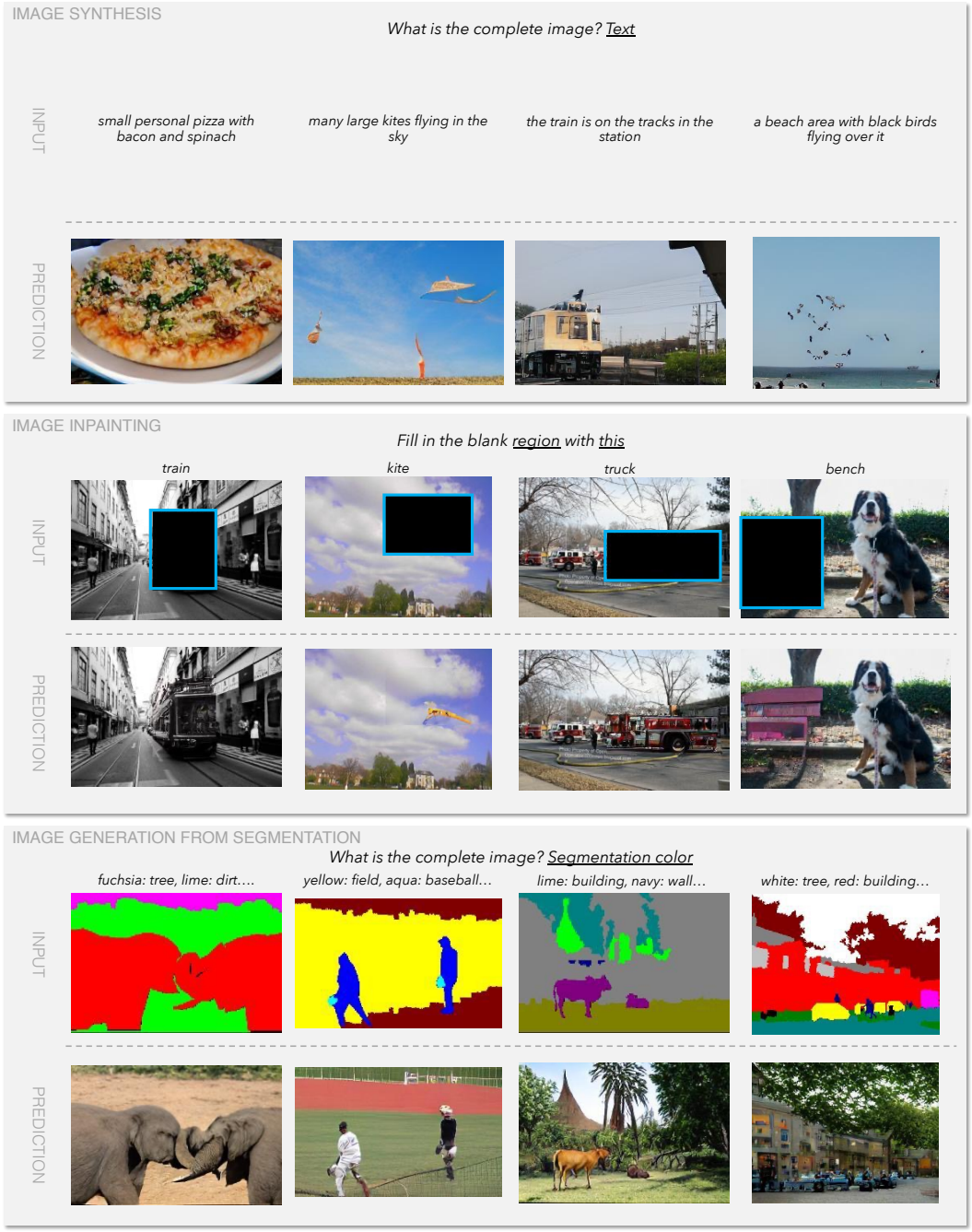


Figure 5: Image synthesis qualitative examples.

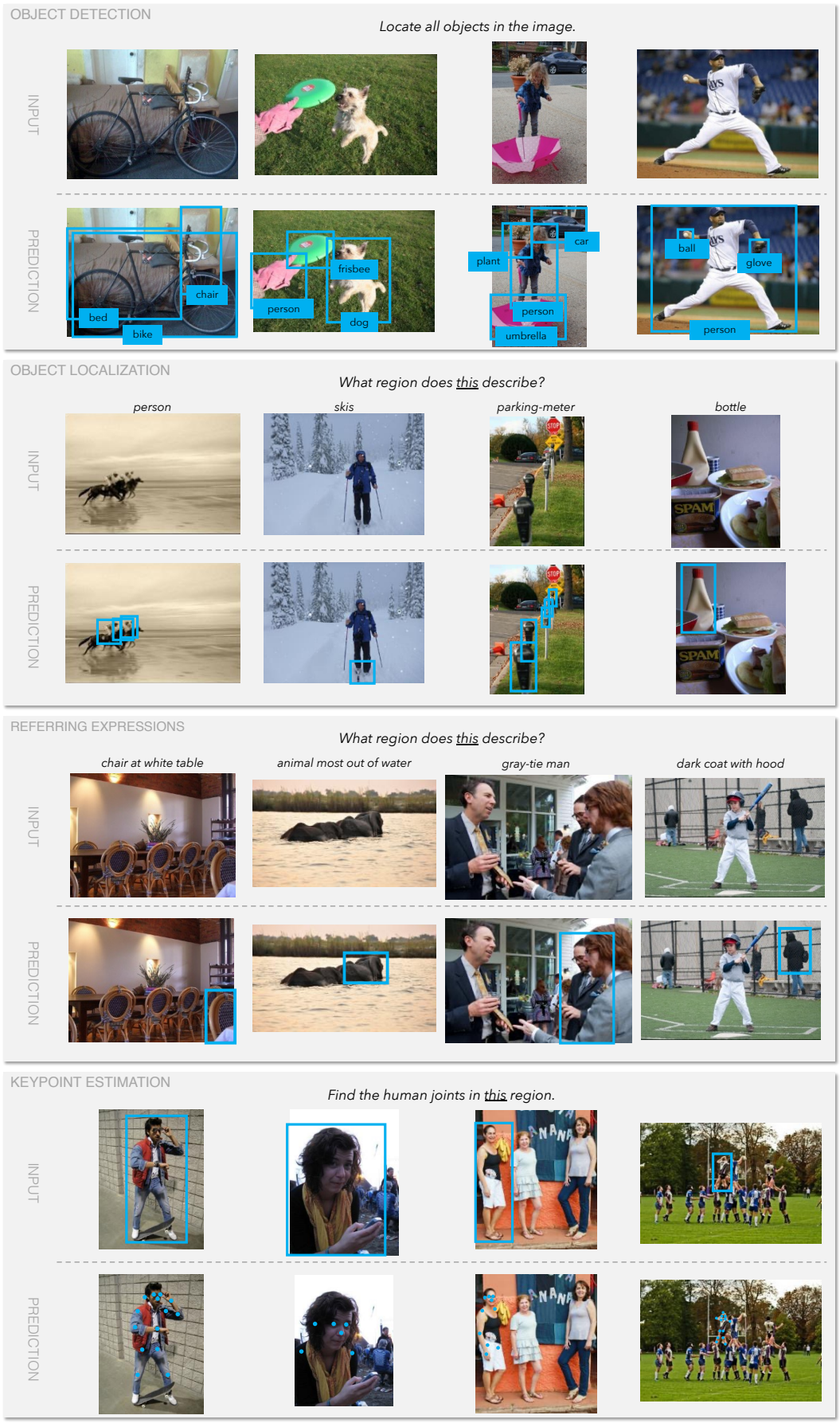


Figure 6: Sparse labelling qualitative examples.

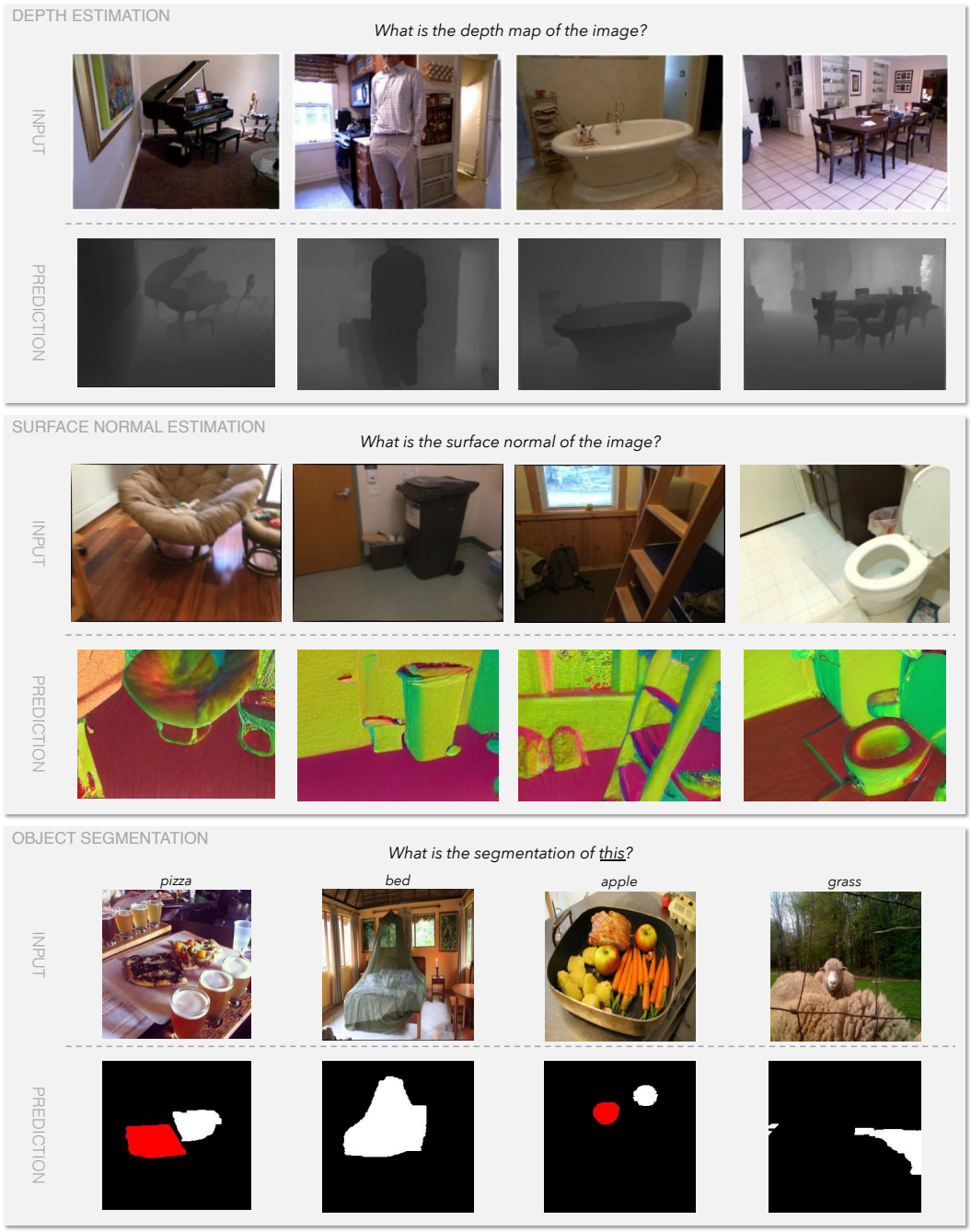


Figure 7: Dense labelling qualitative examples.

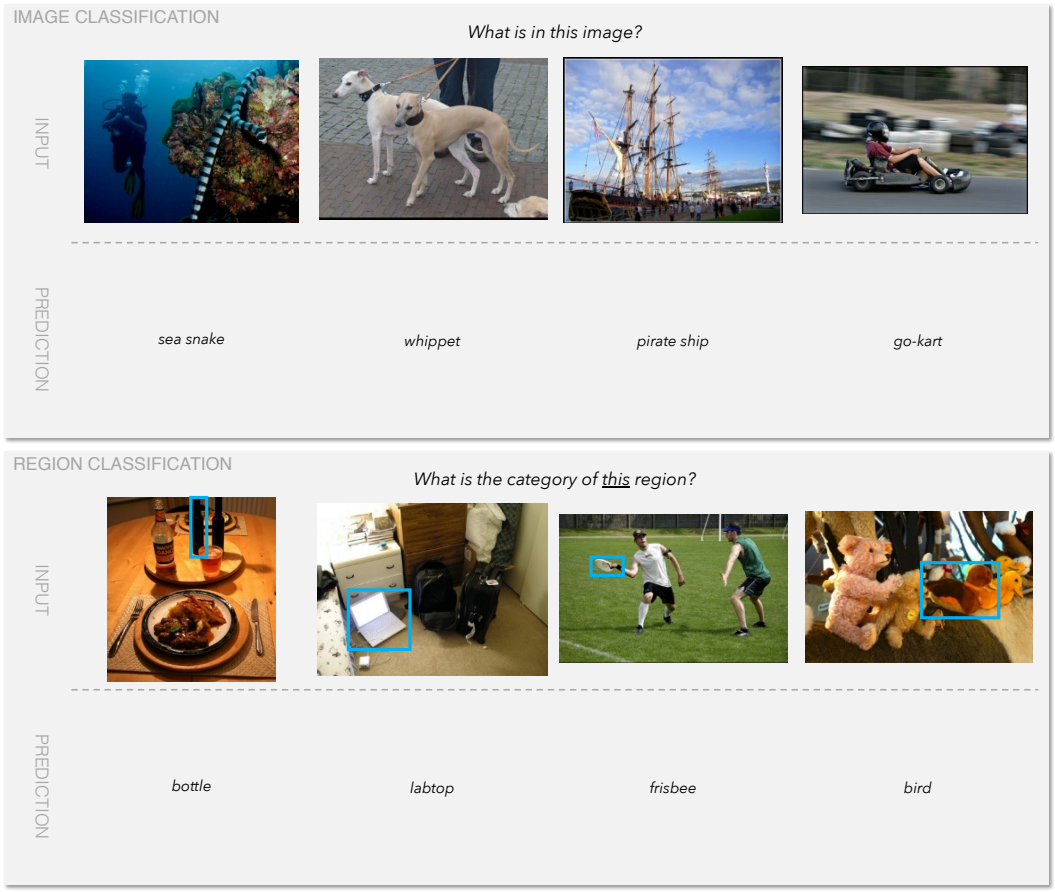


Figure 8: Image classification qualitative examples.

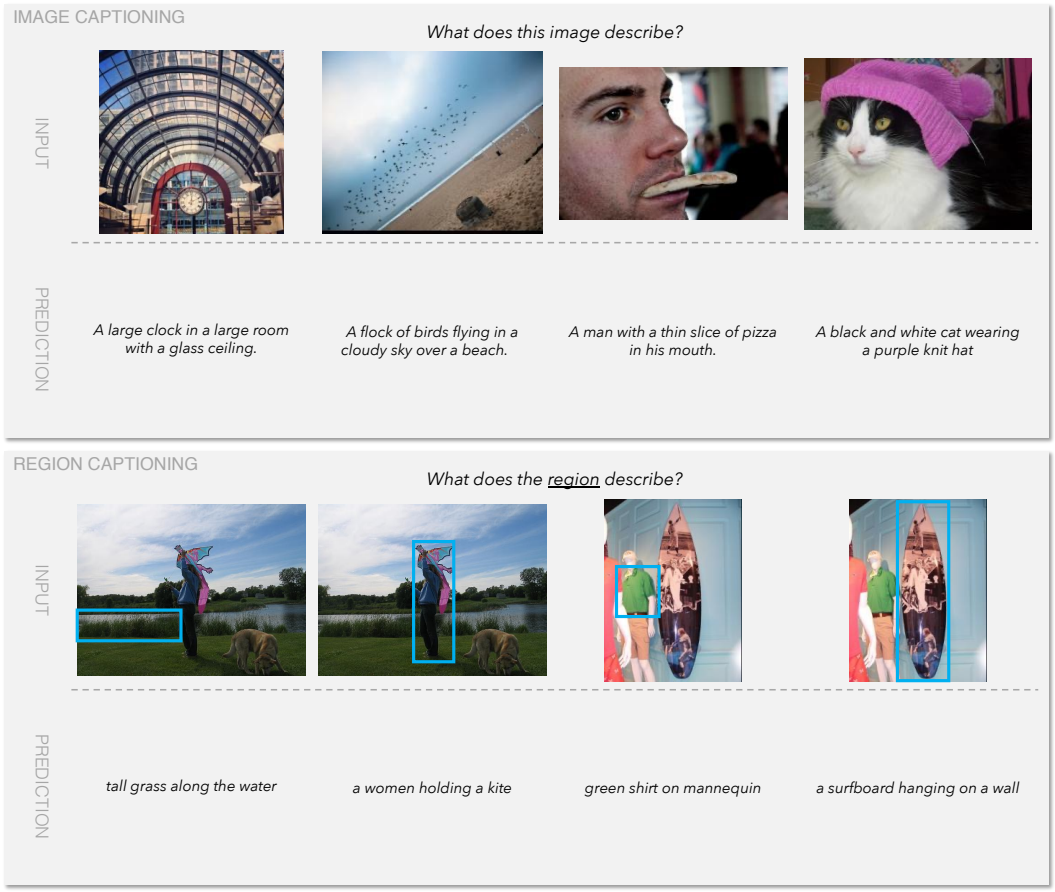


Figure 9: Image captioning qualitative examples.

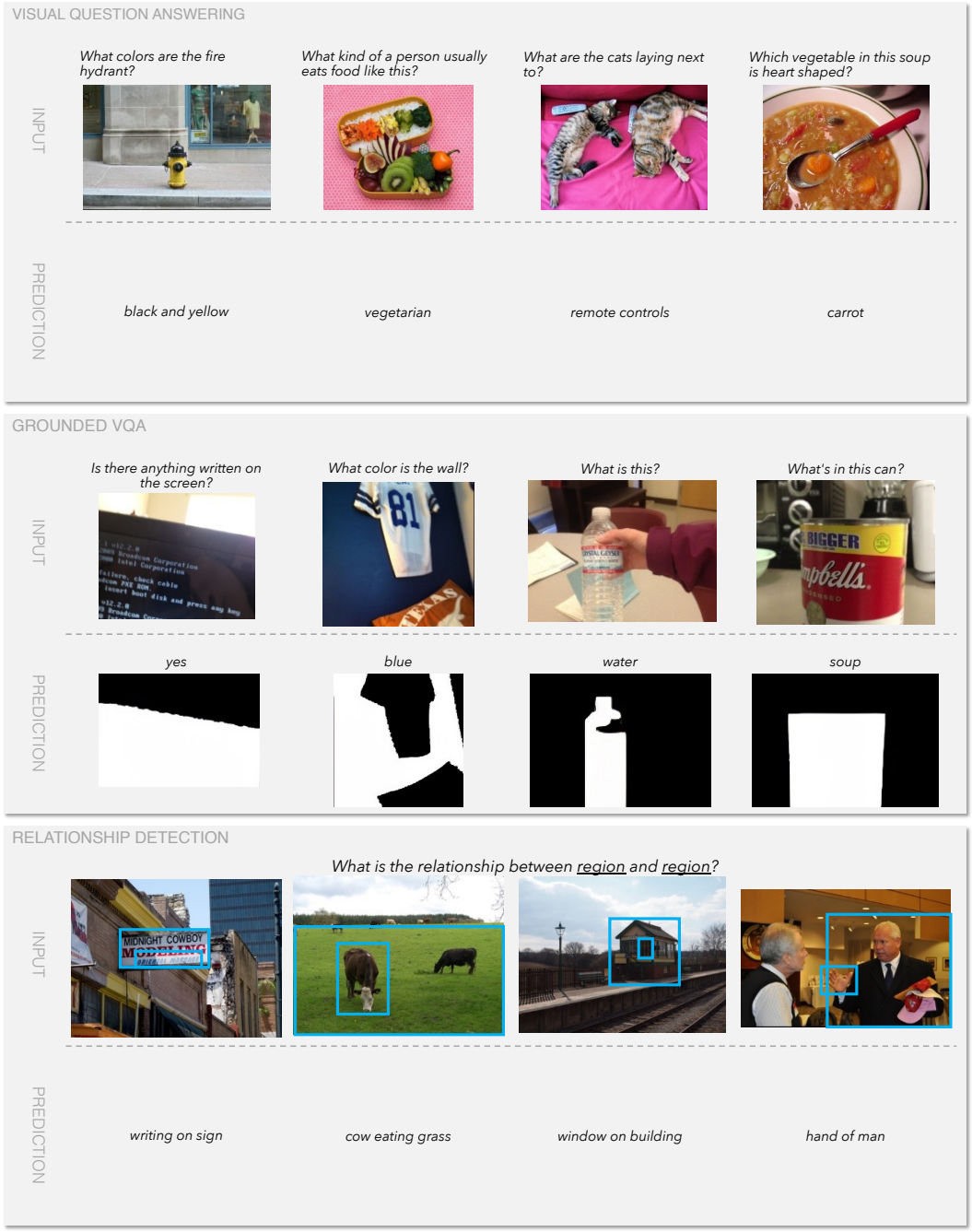


Figure 10: Vision and language qualitative examples.

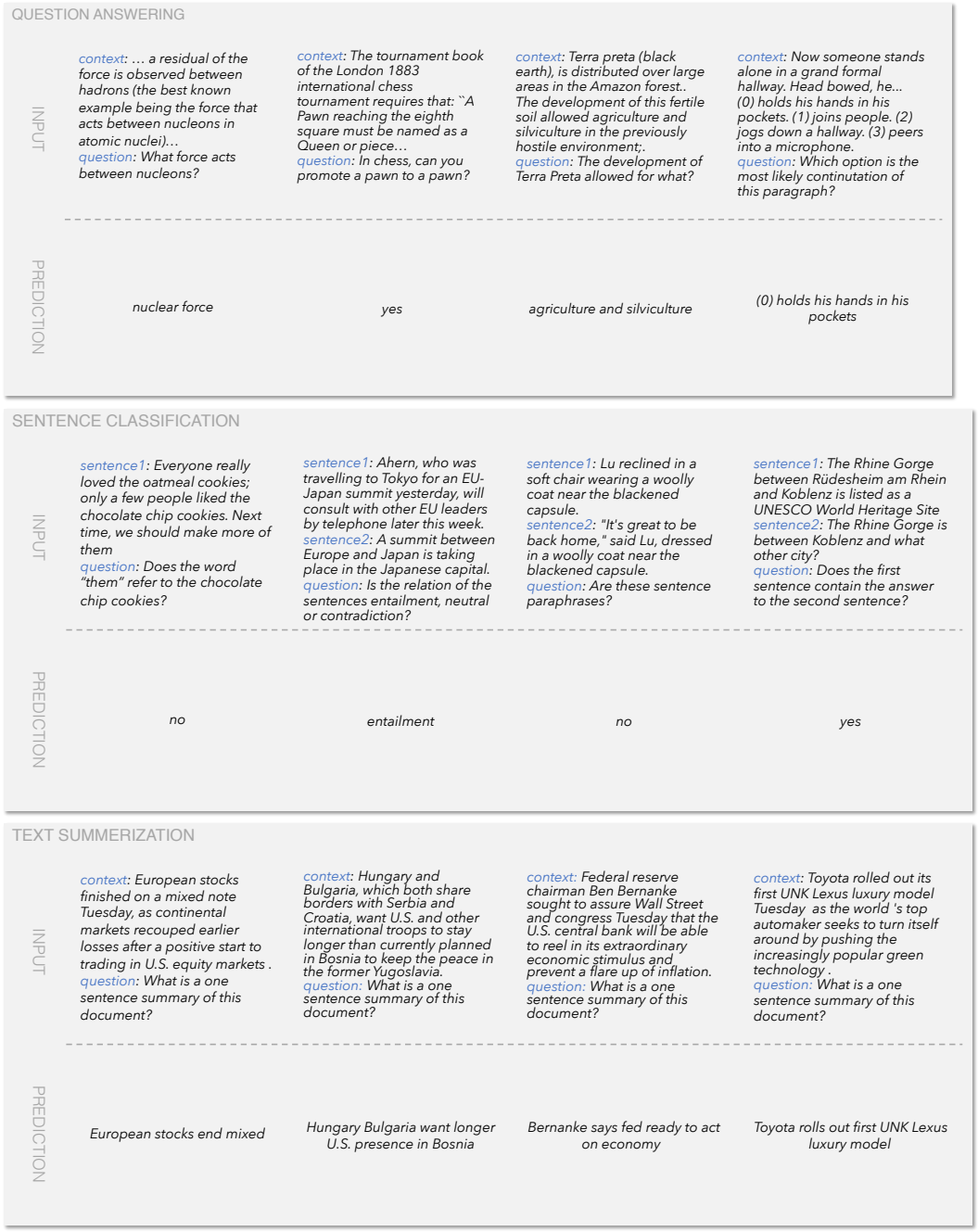


Figure 11: Natural language processing qualitative examples.