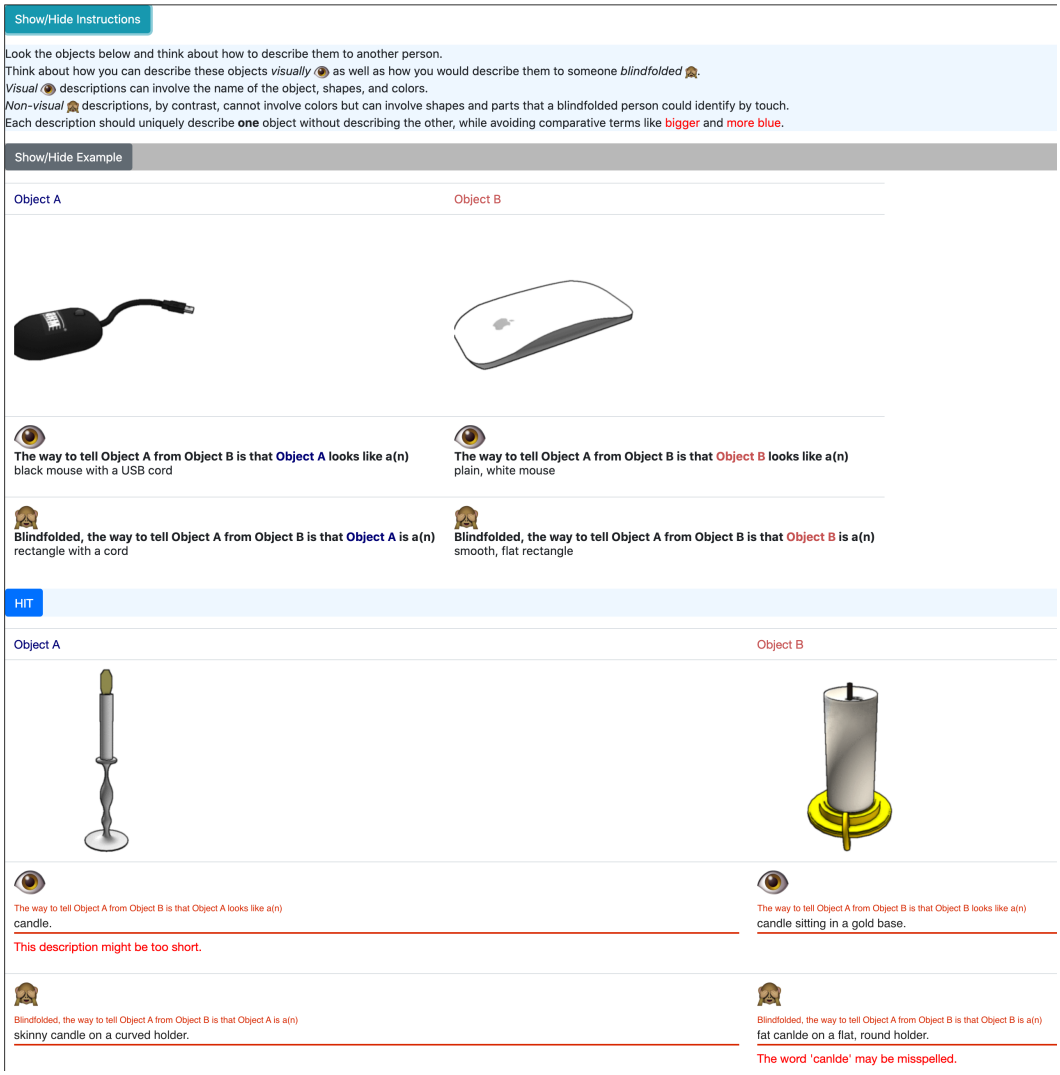


Figure 6: Workers were given instructions about the task as well as an example of how to complete it, then asked to fill in the blanks for visual and blindfolded descriptions of two objects. In this interface example, warnings are given for a misspelled word and a short description.

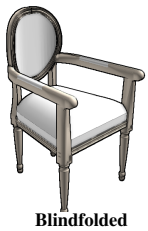
7 Supplementary Material

7.1 Amazon Mechanical Turk Experiments

In total, we collected 50,594 referring expression annotations from Amazon Mechanical Turk workers. We ran annotation tasks, qualification tasks, and validation tasks in parallel, gathering data iteratively. In total, the annotations cost around $10,800 USD.

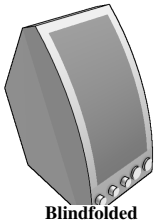
Annotation Interface Workers wrote referring expressions conditioned on a pair of objects per HIT (Human Intelligence Task). The interface for this annotation task is shown in Figure 6. Two expressions described one object, and two described the other. For each object, one expression was primed to be visual in nature, while the other was primed to be blindfolded.

During annotation, warnings were given for misspelled words, descriptions being too short, and for having substantial overlap with one another. Workers were not prevented from submitting HITs with warnings, but workers who accrued many warnings were automatically disqualified from the task.

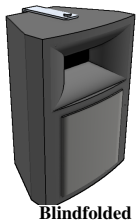


Visual	Blindfolded
classic armchair with white seat	oval back, and vertical legs
white armchair	seat cushion circular back rest and 4 carved wood legs
white chair	has straight legs

Visual	Blindfolded
mod black chair	scoop shaped like puzzle piece, on four angled legs
black chair	lowset chair with 4 angled out legs
black chair	has legs that flare out



Visual	Blindfolded
speaker with five knobs	circles on the face
light grey speaker with knobs on front	five small circles projected off front
light gray speaker	curved speaker with five knobs on the front



Visual	Blindfolded
rectangular speaker	rectangle that becomes more narrow
dark grey speaker metal stripe down back	square with square hole towards top and bolt on top
tall, dark grey speaker	tall trapezoid shaped speaker with hole near the top

Figure 7: Samples of object pairs and the object referring expressions in SNARE.

Quality Control Workers first qualified for our annotation task by completing a fixed qualification HIT. In the qualification HIT, users read the instructions and wrote visual and non-visual descriptions for the example objects (`ComputerMice`) shown in Figure 6. To submit the qualification HIT, users had to produce four referring expressions that created no automatic warnings.

Each referring expression was validated by a secondary HIT to ensure the SNARE task has a high human success rate. For every referring expression, two votes were gotten from qualified workers asked to choose which of the two objects was the referent. Object positions (left versus right) were randomly scrambled to identify and remove referring expressions that made use of the object position on the screen at annotation time. Validation workers were asked to select the object that the referring expression best described, and had the option to select that it described both equally or described neither. Votes were marked as endorsements for the referring expression only if the correct object was selected. If the two votes disagreed, a tie-breaker vote was collected. Referring expressions with a majority of endorsing votes were kept. In total, 5,018 annotations were flagged as unreliable, representing about 9% of the total annotations gathered. New annotation HITs were launched to replace such unreliable expressions.

Workers exhibiting poor performance, estimated through automated metrics, were disqualified from the task, and could not be re-qualified by taking the qualification task again. Workers were also automatically disqualified for the annotation task if the average number of problematic expressions (those given on-screen warnings, as exhibited in Figure 6) or number of rejected expressions (as determined by validation workers) became an outlier. Workers were automatically disqualified for the validation task if the average number of times they were the minority voter in a disagreement became an outlier. Outliers were identified as those whose averages on these metrics were more than one standard deviation above the mean, or 2 standard deviations in the case of automated warnings.

7.2 SNARE Analysis

Additional Examples Figures 7 and 8 show additional examples of object pairs shown to workers and the resulting referring expressions gathered for SNARE.

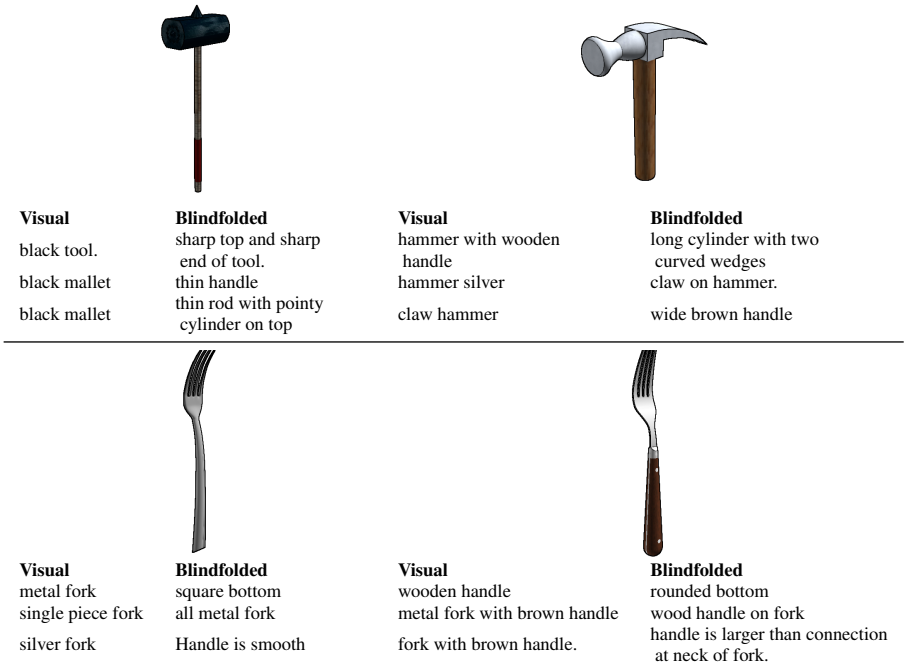


Figure 8: Samples of object pairs and the object referring expressions in SNARE.

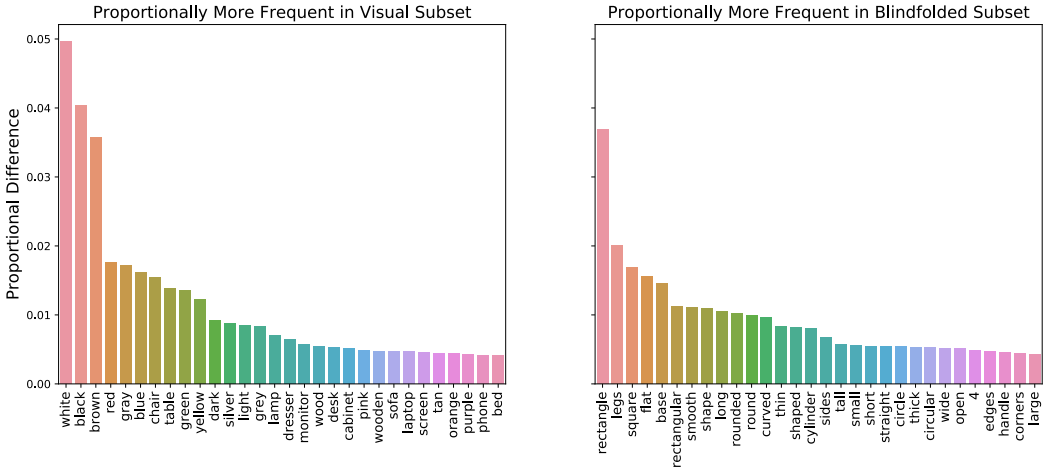


Figure 9: The top 30 token types (omitting stop words) with the highest proportional frequency difference between visual and blindfolded primed referring expressions. Color and category words like *white*, *red*, and *table* appear more frequently in visual descriptions (Left), in contrast to shape and part words like *rectangle* and *legs* in blindfolded descriptions (Right).

Referring Expression Token Diversity There are 216,146 unique token types across all referring expressions, with an average of 4.27 ± 2.07 tokens per expression. However, visual expressions have less token diversity (93,733 types) and are shorter (3.63 ± 1.72 tokens) than blindfolded expressions (122,413 types, 4.95 ± 2.19). Visual descriptions largely focus on color and object category, while blindfolded expressions include shape and part descriptions (Figure 9).

7.3 Data Folds

We split the data into train, validation, and test folds by ShapeNet category. We ensure that closely related categories such as `2Shelves` and `3Shelves` or `DiningTable` and `AccentTable` are

Pair Folds	# Categories	# Objects	# Referring Expressions	% Dataset Size
Train-Train	207	6153	39104	78.0
Val-Val	7	371	2304	4.6
Train-Val	-	25	76	0.15
Test-Test	48	1357	8751	17.4

Table 3: Fold summaries in the SNARE benchmark by pairs of object categories. The uniform folds, like "Train-Train", correspond to the folds of Table 1. The "Train-Val" fold can be used when training a final model for evaluation on the "Test-Test" fold, because those models can be trained on both training and validation data. "Train-Val" pairs are all cross-category, hence the '-' entry for its number of category pairs.

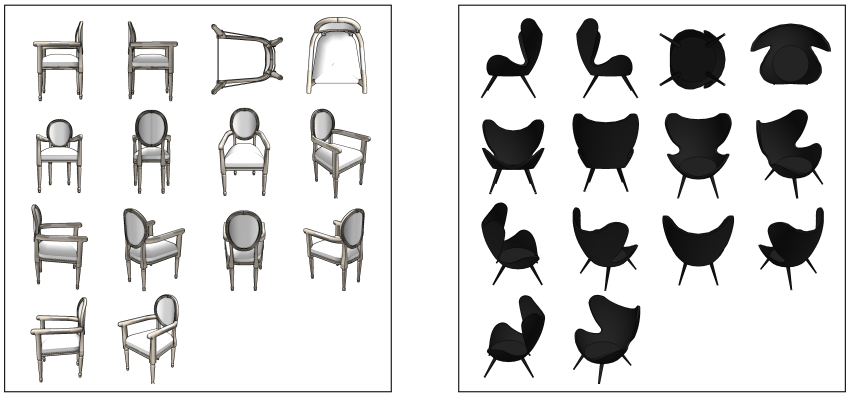


Figure 10: ViLBERT takes in an image and a language expression. To represent a 3D model, we tile 14 views of the object into a single image. An instance of the SNARE task is then represented by a language expression, such as *lowset chair with 4 angled out legs*, paired with two such tiled images. The model must predict the image that the expression refers to.

contained to a single fold. To do so, we mapped each category to a single descriptive word, such as "shelves," then iteratively assigned each category to the fold with which it had maximum cosine similarity in GloVe [50] space according to that word. For example, all shelf-related and bed-related categories were sorted into the train and test folds, respectively.

Objects are annotated in pairs, and these pairs are primarily drawn from the same ShapeNet category. However, due to single-instance categories and the iterative data collection process, there are data pairs of objects paired with others outside of their category. These *cross-category* object pairs comprise only about 4% of the 6,019 pairs used to gather object referring expressions. Cross-category object pairs were assigned to a fold only if both categories were assigned to that fold.

To widen the available training data for the test fold, models can be trained on both the training and validation fold before evaluating on the held-out test fold. When training to evaluate on the test fold, cross-category object pairs where one category is in train and one is in validation can also be used (only 13 object pairs fit this description). Cross-category object pairs where one category was in the test fold and the other was not cannot be used for training, validation, or testing (only 59 object pairs fit this description). Table 3 presents a full summary of folds considering these cross-category pairs.

7.4 ViLBERT Model

While our MATCH module uses a CLIP [1] backbone, we also experimented with a ViLBERT [6] backbone. ViLBERT training and inference are slower, but 3D objects could be encoded via a single tiled image because ViLBERT attends to individual bounding boxes in input images. We compare CLIP and ViLBERT performance on full-object (all view) encodings and find that they perform similarly, choosing CLIP as the backbone for MATCH for training speed.

Fold	Model	Task Pretraining	Accuracy (%) ↑ All	Vis⊂	Blind⊂
Val	ViLBERT	None; Architecture Only	49.0	48.8	49.1
	ViLBERT	Conceptual Captions	82.6	89.0	76.1
	ViLBERT	12-in-1	**83.1**	**89.5**	**76.6**
Test	ViLBERT	Conceptual Captions	75.1	79.4	70.5
	ViLBERT	12-in-1	**76.6**	**80.2**	**73.0**

Table 4: ViLBERT model accuracy on the SNARE benchmark under different pretraining regimes.

Model	Views	Pooling	Validation Visual⊂	Blind⊂	All
CLIP	All	`maxpool`	83.7 (0.0)	65.2 (0.0)	74.5 (0.0)
CLIP	All	`meanpool`	**85.1** (0.0)	**65.7** (0.0)	**75.5** (0.0)
MATCH	All	`maxpool`	89.2 (0.9)	75.2 (0.7)	82.2 (0.4)
MATCH	All	`meanpool`	**90.1** (0.7)	**75.6** (0.7)	**82.9** (0.4)

Table 5: We experiment with meanpool as an alternative to maxpool for considering multiple object views at once via CLIP zero shot scoring and fine-tuned MATCH approaches.

ViLBERT. We encoded each object as a set of 14 views from different angles as provided by ShapeNetSem (Figure 10). We provide ViLBERT the gold-standard bounding boxes for these 14 views during feature extraction, removing potential pipeline errors from the pretrained object detector [25]. Each position in the tiled image represents a fixed camera angle from which the object is rendered, so spatial information encoded by the bounding boxes is consistent across objects; for example, the model may learn to look at the fourth image in the top row to assess facts about the "top" of the object. In Table 4, we compare ViLBERT models that are not pretrained on any tasks versus those pretrained only on conceptual captions [51] or on 12-in-1 task pretraining [25]. We find that the base ViLBERT architecture without task pretraining is unable to learn anything meaningful with only our task data, and so only run this evaluation on the validation fold. Note that this model still includes a pretrained BERT [52] and ResNet [2] backbone, but without task-oriented training is too parameter-heavy to learn on our data. ViLBERT pretrained on conceptual captions achieves nearly as high performance as the model pretrained on 12 language and vision tasks, indicating that this large amount of aligned language and vision data is sufficient to initialize better parameters for our task.

7.5 Multi-view Pooling Operations

In Table 5, we compare `meanpool` and `maxpool` operations to unite the vector representations of the 8 views covering each object. Using a Welch's two-tailed t-test, we compared the average accuracy across 10 seeds of training of MATCH trained with these pooling operations, and found that `meanpool` yields statistically significantly higher performance than `maxpool`.[7] Our experiments currently use `maxpool` to combine two for both MATCH and LAGOR, and so this ablation indicates a further performance boost may be possible by re-training models with `meanpool`.

7.6 Additional Robot Demonstrations

Figure 11 gives 11 referring expressions against pairs of objects and the zeroshot CLIP versus trained LAGOR object picks. The robot recorded two "clean" images for each object, with the background removed, to make the input images similar to those seen at training time. In order to get clean masks, we performed a dilation operation on the masks, resulting in a thin border of background around each one. By adding segmentation, this pipeline should allow us to select individual objects from a

[7] $p = 0.0022 < 0.01$; Bonferroni multiple-comparison correction applied to threshold 0.05 because we run five such tests in total.

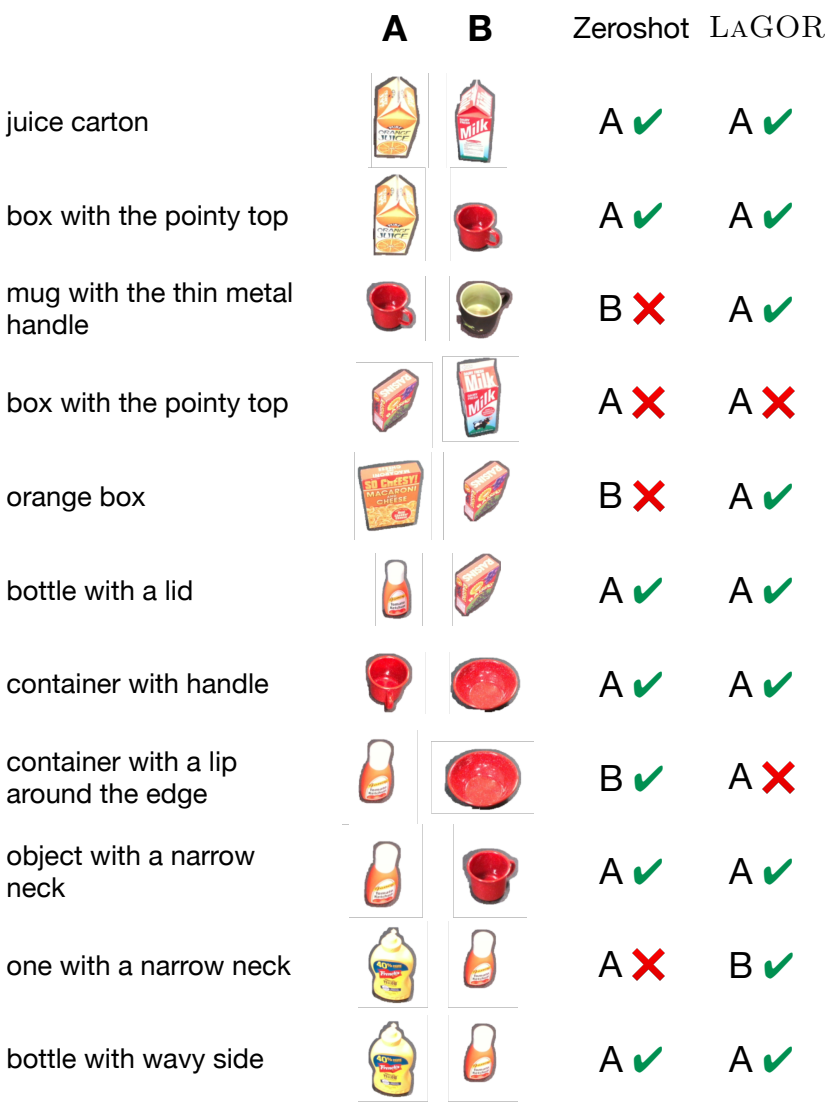


Figure 11: Examples of zeroshot CLIP versus LAGOR decisions on which object a referring expression references based on segmented detections. Zeroshot CLIP uses only one view of each candidate objects, while LAGOR uses two views from different angles.

cluttered scene. As such, it represents closely how LAGOR could be used on a real robotic system for household pick and place tasks.