Shapeglot: Learning Language for Shape Differentiation Supplemental Materials

Panos Achlioptas Judy Fan Robert Hawkins
Noah Goodman Leonidas Guibas
Stanford University

Webpage/Code: https://ai.stanford.edu/~optas/shapeglot

1. ShapeGlot details

To build the triplets comprising the communication contexts of ShapeGlot, we exploited the *latent* (bottleneck-derived) vector space of a Point-Cloud based AutoEncoder (PC-AE) [1], trained with chair-only objects of ShapeNet [3]. Concretely, we used a PC-AE with small bottleneck (64D) to promote meaningful euclidean distances and after embedding all $\sim$ 7000 ShapeNet chairs in the resulting space, we computed their underlying 2-(euclidean)-nearest-neighbor graph. On this graph, we selected the $1K$ chairs with the highest in-degree to 'seed' the triplet generation. For each of the 1K (seed) chairs, we considered it together with its two nearest neighbors from the *entire* shape collection to form a *Hard* triplet. Also, we considered it together with the two chairs that were closest to it but which were also more distant from it than the median of all pairwise distances, to form an *Easy* triplet. The above procedure gives rise to 2000 communication contexts when target vs. distractor information is ignored. However, to counterbalance the dataset while annotating these contexts in AMT, we ensured that *each* chair of a context was considered as a distractor and as a target, and that each resulting combination was annotated by at least 4 humans. Last, we note that when building the Hard triplets, we applied a manually tuned distance-threshold, to reject triplets that contained objects that were 'too' close: we found that about $\sim 3\%$ of chairs had a geometric duplicate that could vary only wrt. its texture.

2. Image and point-cloud pre-training

For the listeners and speakers we trained a PC-AE under the Chamfer loss [1] with a 128D bottleneck and point clouds with 2048 points extracted from 3D CAD models, uniformly area-wise. We also fine-tuned a VGG-16 pre-trained on ImageNet on a 8-way classification, with 36,632 rendered images of textureless 3D CAD models, taken from a single view-point. Concretely, we used images of the 8 largest object classes of Shape-Net (car, airplane, vessel, sofa, chair, table, lamp, riffle) and a uniformly random i.i.d. split of [90%, 5%, 5%] for train/test/val purposes. We fine-tuned the network for 30 epochs. During the first 15 epochs we optimized *only* the weights of the last (fc8) layer and during the last 15 epochs the weights of all layers. The attained test classification accuracy was 96.9%. Last, to embed an image for the downstream listening/speaking tasks, we used the 4096D output activations of the penultimate (fc7) fully-connected layer.

3. Pre-processing utterances

We preprocessed the collected human utterances by i) lowercasing, ii) tokenizing by splitting off punctuation, iii) tokenizing by splitting superlative or comparative adjectives ending in -er, -est to their stem word, e.g. 'thinner:' $\rightarrow$ ['thin', 'er'] and, iv) replacing tokens that appear once or not at all in a training split with a special symbol marking an unknown token (<UNK>). Furthermore, we ignored the utterances comprised by more than 33 tokens (99th percentile) and those for which the human listener in the underlying trial did not guess correctly the target. Last, we concatenated listener and speaker utterances from the same trial (in their order of formulation) by adding in the end of each but the last utterance a special symbol marking a dialogue: (<DIA>), e.g. ['the', 'thin', 'chair', <DIA>, 'yes'].

4. Listeners details

For the listeners we used a uni-directional LSTM cell with 100 hidden units, the output of which was passed into a 3-layer MLP with [100, 50, 3] neurons that predicted the triplet's classification logits. To the output of each hidden layer of the MLP, batch normalization [4] and a ReLU [7] non-linearity was applied. The listeners' word-embedding was initialized with a $100D$ GloVe embedding pre-trained on the 6B Wikipedia 2014 corpus, and which was further fine-tuned during training. The PC-AE (128D)

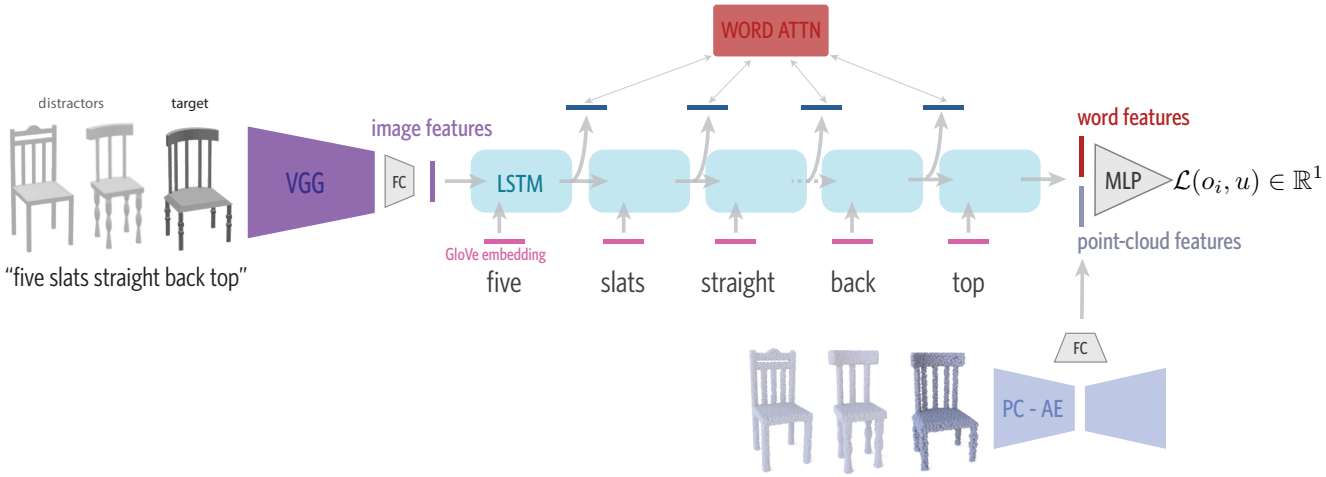


Figure 1: *Baseline* listener architecture combining 2D images, 3D point-clouds and linguistic utterances.

Hyper Parameters \ Architecture	*Baseline*	*Early-Context*	*Combined-Interpretation*
Learning rate	0.0005	0.001	0.001
Label-smoothing	0.9	0.9	0.9
L_2 regularization	0.3	0.05	0.09
LSTM-input-dropout	0.5	0.7	0.45

Table 1: Optimal hyper-parameters for ablated neural listener architectures, using both geometric modalities and word-attention and various degrees of context. Dropout numbers reflect the *keep* probability.

and VGG (4096D) latent vectors, that encoded each object, were passed as *input* to the LSTM when only one geometric modality was used. When the two modalities used together, the PC-AE codes were concatenated with the *output* of the LSTM, and the concatenated result was processed by the final MLP. In either case, we first re-embedded these geometric codes (100D) with 2 separate/single FC-ReLU layers (referred as 'projection' layers in the Main Paper Section 3). An overview of the proposed listener reflecting the overall design choices is given in Fig.1. We used dropout with 0.5 keep probability *before* the 'projection' layers with a drop-out mask that was the same for the objects of a given triplet. Separate dropout with 0.5 keep probability was applied in all input vectors of the LSTM (i.e. on the language tokens or the grounding geometric codes). Last, the ground-truth indicator vectors of each triplet were label-smoothed [10] by assigning 0.933 probability mass to the target and 0.0333 to the distractors (i.e. smoothing of 0.9).

Discussion Label smoothing yielded a mild performance boost of $\sim 2\%$ across all ablated listener architectures, in accordance with previous work [10]. We note that we did not manage to improve the best attained accuracies by applying layer normalization [2] in the LSTM, or adversarial regularization [8] on the word-embedding. Dropout [9] was by far the most effective form of regularization for our listeners ($\sim$[8-9]%), following by L_2 weight-regularization of the projection layers ($\sim$[2-3]%). Finally, using a separate MLP to process the PC-AE codes, was slightly better than feeding them directly in the LSTM (after the tokens of each utterance were processed). However, grounding the LSTM with the PC-AE codes, and using the VGG codes in the end of the pipeline (either via pre-MLP concatenation or by feeding the latter in the LSTM) deteriorate *significantly* all attained results.

Context Ablations We ablated three architectures that used simultaneously images and point-clouds, word attention and different degrees of context (See Main Paper Section 3). The optimal Hyper-Parameters (HP) for each architecture are shown in Table 1. We did a grid search over the space of HP associated with each architecture *separately*. To circumvent the exponential growth of this space, we search it into two phases. First, we optimized the learning rate (in the regime of [0.0001, 0.0005, 0.001, 0.002, 0.004, 0.005]) in conjunction with the drop-out (keep probability) applied at the LSTM's *input*, in the range [0.4-0.7] with increments of 0.05. Given the acquired optimal val-

ues, we searched for the optimal L_2 weight-regularization (in the range of [0.005, 0.01, 0.05, 0.1, 0.3, 0.9]) applied at the two projection layers, and label-smoothing ([0.8, 0.9, 1.0]). For these experiments we used a single random seed to control for the data splits with the *object-generalization task*. We note that for the *Early-Context* listener, using a single 1D convolutional layer to extract the grounding vector of each object, appeared to produce better results than using a single FC layer (or deeper alternatives). This single convolutional layer we used, converted the input signal $[f(v_j, v_k)||g(v_j, v_k)||v_i] \in \mathbb{R}^{100\times 3}$ to a $\mathbb{R}^{100\times 1}$ LSTM-grounding vector for *e*ach object v_i, with an $8 \times 3 \times 1$ kernel and stride 1.

Training We trained the *Baseline* and the *Combined-Interpretation* for 500 epochs and the *Early-Context* for 350. This was sufficient, as more training increased overfitting without improving the attained test/val accuracies. We halved the learning every 50 epochs, if the validation error was not improved in any of them. Namely, every 5 epochs we evaluated the model on the validation split in order to select the epoch/weights with the best accuracy. Because the *Combined-Interpretation* is sensitive in the input order of the object codes, we randomly permute them during training. We use the ADAM [6] ($\beta_1 = 0.9$) optimizer for all experiments.

5. Speaker details

Image-based speaker To find good model parameters for an image-based speaker, we considered a hyper-parameter search on a *literal* variant. Similarly, to what we did in the ablations of listener variants we conducted a two-stage grid search given a single random seed and the *object generalization* task. At the first stage, we searched models varying: a) the hidden neurons of the LSTM (100 or 200), b) the initial learning rate ([0.0005, 0.001, 0.003]), c) the drop-out keep probability applied on the word-embeddings ([0.8, 0.9, 1.0]) and d) the dropout keep probability applied at the LSTM's output ([0.8, 0.9, 1.0]). The two best performing models were further optimized by considering L_2-weight regularization applied at the FC-projection layer (with values in [0, 0.005, 0.01]) and the dropout keep-probability applied before the FC-projection layer ([0.5, 0.7, 0.9 1.0]). The resulting optimal parameters are reported in Table 2.

Point-cloud-based speaker For the point-based speaker, we did a similar but more constrained hyper-parameter search as we did for the image-based speaker, by also considering its *literal* variant. Here, we fixed the drop-out applied the word-embeddings and to the LSTM's output (0.8 and 0.9 keep-probability respectively) and ablated the remaining hyper-parameters as we did for the image-based speaker. We found the same configuration of parameters (Table 2) to be optimal for point-based models as well. Exception to this was the the dropout applied to the PC-AE codes before the FC-projection (no dropout at all was best in this case). Also, the point-based speakers needed more training to converge than the image-based ones (maximally 400 epochs vs. 300).

Model selection To do model selection for a training speaker, we used a pre-trained listener (with the same train/test/val splits) which evaluated the synthetic utterances produced by the speaker during training. To this purpose the speaker generated 1 utterance for each unique triplet in the validation set via greedy (arg-max) sampling every 10 epochs of training and the listener reported the accuracy of predicting the target given the synthetic utterance. In the end of training (300 epochs for image-based speakers vs. 400 for point-based ones), the epoch/model with the highest accuracy was selected.

Other details We initially used GloVe to provide our speakers pre-trained word embeddings, as in the listener, but found that it was sufficient to train the word embedding from uniformly random initialized weights (we used the range [-0.1, 0.1]). We also initialized the bias terms of the linear word-encoding layer with the log probability of the frequency of each word in the training data [5], which provided faster convergence. We train with SGD and ADAM ($\beta_1 = 0.9$) and apply norm-wise gradient clipping with a cut-off threshold of 5.0. The training utterances have a maximal length of 33 tokens (99th percentile of the dataset). For any speaker we sampled utterances of the maximum training length. For the *pragmatic* speaker we sample and score 50 utterances per triplet at test time (following Eq.1 of Main Paper).

Point-cloud & image-based speaker In *preliminary* experiments, we attempted to incorporate both geometric modalities: point-clouds and images in a speaker network, similarly to what we did for the best-performing listener. While, this resulted in a (*literal*) speaker model that could achieve higher neural-listener evaluation-accuracy than when either modality was used in isolation, we did not observe any improvement against the image-based speaker in AMT *human*-listener experiments.

We attempted three ways of 'mixing' the two modalities in a speaker. Namely, for each object of a communication context: a) providing the LSTM with the *concatenation* of its projected VGG code and its projected PC-AE code, b) same as a) but instead of concatenation, using the *sum* operator, c) first providing its PC-AE projected code followed at the *next time* step by its VGG one. We compared these approaches by using the optimal hyper-parameters for an

LSTM Size	Learning rate	L_2-reg.	Word-Dropout	Image-Dropout	LSTM-out Dropout
200	0.003	0.005	0.8	0.5	0.9

Table 2: Optimal hyper parameters for *literal* image-based neural-speaker. The dropout numbers reflect keep probabilities and the Image-Dropout refers to the dropout applied at the VGG-image codes, before the FC-projection layer.

Approach	Listener's Accuracy
Concat (100D)	$65.1 \pm 0.51\%$
Concat (200D)	$78.2 \pm 0.95\%$
Sum	$77.9 \pm 0.38\%$
Serial	$\mathbf{79.0} \pm 0.32\%$

Table 3: Ablating approaches for incorporating simultaneously point-clouds with images in a *literal* neural-speakers. *Sum*: Summing the two latent codes for each object. *Concat*: Concatenating the codes. *Serial*: Feeding them one after the other in the LSTM. Concatenation naturally doubles the input-dimensions of the LSTM (Concat 200D). To keep them the same as with all other experiments (100D) we also tested reducing the VGG/PC-AE projection layers to 50 dimensions for each modality (Concat 100D). Results are averages of 5 samples of utterances for a fixed test dataset.

image-based speaker and only vary the amount of dropout applied to the point-cloud before the projection layer ([1.0 0.8, 0.6] keep probability). In all cases, avoiding dropout was best. The final results for a single random-seed and the object-generalization task are reported in Table 3. We note that while the optimal speaker that used two modalities performed slightly better than the image-based speaker, per neural-listener evaluation, it did not improve the attained performance in preliminary experiments with of human listeners in AMT.

6. Further quantitative results

6.1. Listeners: context incorporation

In Table 4 we complement the results presented in the Main Paper at Table 1, by including two more subpopulations ('Negative' and 'Split'). In Table 5, we repeat this study for listeners trained and tested on the *language generalization* task. 'Negative' is a subpopulation of utterances that contain at least one word of negative content e.g. 'not', 'but' etc. and is comprised by $\sim 15.0\%$ of all test utterances. 'Split' is smaller subpopulation ($\sim 3.2\%$ of test data) that includes language the explicitly contrasts the target with the distractors e.g. 'from the two that have thin legs, the one...'. We used an ad hoc set of search queries to find such utterances among the test set and found that the *Early-Context* architecture does perform noticeably better on these utterances. However, given the low occurrence of such cases, the resulting effects were not significant and we decided the gains of *Early-Context* architecture were not worth the increase in model complexity and rigidity with respect to context size.

6.2. Listeners: part-lesion

We complement Table 3 of the Main Paper, with a similar study (Table 6) where we ablate our neural listeners with regards to their sensitivity in referential utterances based on object parts, when *both* geometric modalities are used. We have observed that the PC-AE attempts to reconstruct (decode) noisy but *c*omplete models, even when the input is a partial, which could explain the gains seen in Table 6 compared to Table 3.

6.3. Speakers: length penalty and listener awareness

To find the optimal length-penalty value (α, Main Paper Eq.1) for image-based *literal* and a *context-unaware* speaker variants, we used our best-performing listener to simultaneously score and evaluate the utterances produced by the speakers for different values of α (Fig. 2a). The best performing length penalty for a context-unaware speaker is 0.7, and for a literal 0.6. Given the optimal α values, for these models we show the effect of using different degrees of listener-awareness (β) in Fig. 2b. It is interesting to observe that even the context-unaware speaker can generate utterances that an evaluating listener can find them very discriminative, as long as is allows to rank them.

In Fig. 3 we demonstrate the effect that the relative (training) size of the evaluating listener vs. the 'internal' listener used by a *pragmatic* speaker has for the evaluating accuracy, for two values of β. In either case we observe a slow decline in evaluating accuracy as the training size for the evaluating listener increases (from 0.5 to 0.9) and consequently the training size for the 'internal' listener decreases (from 0.5 to 0.1).

6.4. Understanding out-of-class reference

We complement the Table 5 (Main) with the standard-deviations of the underlying accuracies in Table 8. We also report simple statistics regarding the underlying transfer classes in Table 7. We note that the transfer learning accuracies acquired by listeners operating with both point-clouds and images for these experiments were significantly lower ($\sim 7\%$ on average). We hypothesize that this is due to the fact that our (chair-trained) listener models that utilize

Architecture	Overall	Subpopulations				
		Hard	Easy	Sup-Comp	Negative	Split
Combined-Interpretation	$75.9 \pm 0.5\%$	$67.4 \pm 1.0\%$	$83.8 \pm 0.6\%$	$74.4 \pm 1.5\%$	$77.3 \pm 1.5\%$	$65.8 \pm 5.2\%$
Early-Context	$79.4 \pm 0.8\%$	**70.1** $\pm 1.3\%$	$88.1 \pm 0.6\%$	$75.6 \pm 2.2\%$	**78.9** $\pm 1.4\%$	**67.4** $\pm 3.6\%$
Baseline	**79.6** $\pm 0.8\%$	$69.9 \pm 1.3\%$	**88.8** $\pm 0.4\%$	**76.3** $\pm 1.3\%$	$77.5 \pm 1.2\%$	$62.5 \pm 3.7\%$

Table 4: Comparing the effect of context inspection for listening on various (test) subpopulations of the *object generalization* task. The listeners use images, point-clouds and word-attention. Reporting averages of five random seeds controlling the split populations and the network's initialization.

Architecture	Overall	Subpopulations				
		Hard	Easy	Sup-Comp	Negative	Split
Combined-Interpretation	$78.4 \pm 0.2\%$	$71.5 \pm 0.6\%$	$85.2 \pm 0.3\%$	$75.8 \pm 0.9\%$	$77.6 \pm 0.8\%$	$61.8 \pm 3.0\%$
Early-Context	**84.4** $\pm 0.5\%$	**78.5** $\pm 0.8\%$	$90.2 \pm 0.7\%$	**80.9** $\pm 0.6\%$	**82.6** $\pm 1.1\%$	**68.9** $\pm 2.3\%$
Baseline	$83.7 \pm 0.2\%$	$77.0 \pm 0.8\%$	**90.3** $\pm 0.3\%$	$80.8 \pm 0.8\%$	$80.5 \pm 1.0\%$	$64.6 \pm 3.7\%$

Table 5: Comparing the effect of context inspection for listening on various (test) subpopulations of the *language generalization* task. The listeners use images, point-clouds and word-attention. Reporting averages of five random seeds controlling the split populations and the network's initialization.

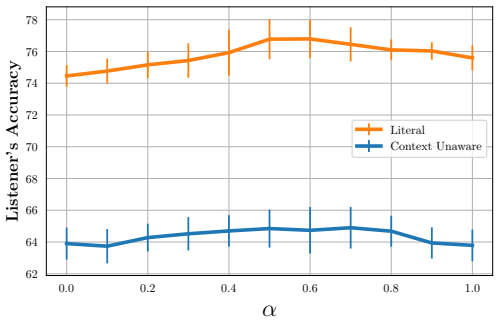
	Single Part Lesioned	**Single Part Present**
Mentioned Part	$44.9\% \pm 1.2$	$67.2\% \pm 1.1$
Random Part	$68.9\% \pm 1.3$	$42.3\% \pm 1.3$

Table 6: Evaluating the part-awareness of neural listeners by lesioning object *parts*. Results shown are for listeners using **both** point-clouds and images, with average accuracy of 78.8% when *intact* objects are used.

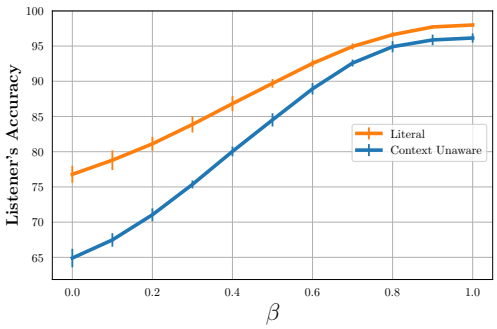
	Population		
Class	entire	with part	without part
chair	7.1	8.0 (77%)	4.7 (21%)
bed	6.4	7.0 (26%)	5.3 (48%)
lamp	7.3	11.0 (20%)	5.9 (37%)
sofa	10.1	11.0 (72%)	5.9 (15%)
table	6.6	8.0 (40%)	4.9 (42%)
average	7.6	9.3 (39.5%)	5.5 (35.5%)

Table 7: Average length of utterances for various transfer classes (complementing Table 5, Main Paper). Between parentheses is reported the percentage of the entire population that is captured by its specific sub-population. The average (last row) is wrt. the transfer classes only; the chair-category is displayed for reference.

point-clouds, rely on a pre-trained *single*-class PC-AE, unlike the pre-trained VGG (image encoder) which was fine-tuned with multiple ShapeNet classes. Also, for these experiments, [~1% ~7%] (depending on the transfer class) of the tokens were not in the chair-vocabulary, and we chose to ignore them i.e. treat them as white-space. Last, per Table 7 in *all* transfer classes the *with-part* population contains quite larger utterances than the *without-part* (9.3 vs. 5.5 on average) and that even in the case of lamps, arguably the most dissimilar category from chairs, $20 + 37 = 57\%$ of the collected utterances are in the *known* population.

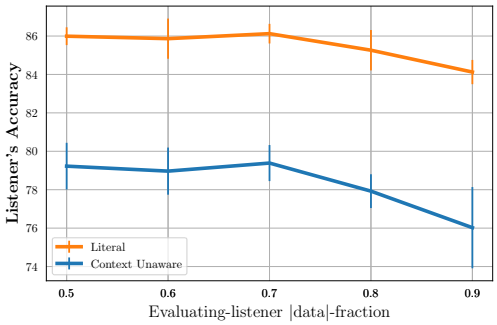


(a) Effect of the length-penalty.

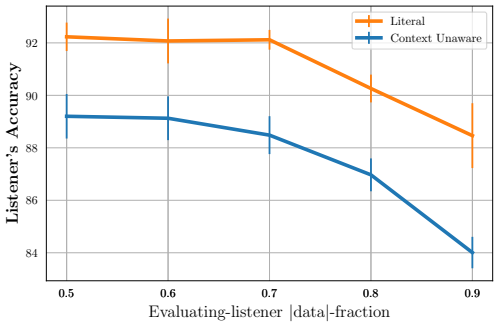


(b) Effect of increasing listener's opinion (β).

Figure 2: Left: Measuring the effect of using different length-penalty (α) values to select the top-1 scoring utterance for context-unaware and pragmatic speakers for contexts of the *object generalization* validation split (left). Right, measuring the the effect of various β-values used in turning the context-unaware and literal speakers ($\beta = 0.0$) to *pragmatic* speakers, under the optimal α of the left figure. In both plots, the y-axis reflects the performance of a listener who is used to rank *and* evaluate the utterances. Averages are with respect to 5 random seeds controlling the data splits and the initializations of the neural-networks.



(a) Speakers using a modest $\beta = 0.5$ value.



(b) Speakers using the most aggressive $\beta = 1.0$ value.

Figure 3: Effect of partitioning the training data for the evaluating and 'internal' listeners. Here, we turn context-unaware and literal speakers into pragmatic ones under two β values. The x-axis shows the fraction (f) of the training data that was used to train the *evaluating* listener (the remaining $100 - f\%$ is used to train the *internal* listener) of the resulting pragmatic speaker. On the y-axis we display the performance of the evaluating listener for the top-scoring model-generated utterance.

Population \ Class	bed	chair	lamp	sofa	table
entire	56.4 ±2.0%	77.4 ±0.9%	50.1 ±1.3%	53.6 ±2.0%	63.7 ±1.2%
known	55.8 ±1.5%	77.8 ±0.8%	51.9 ±1.8%	55.0 ±2.0%	65.5 ±0.9%
with part	**63.8** ±4.2%	77.0 ±0.8%	**60.3** ±4.4%	**55.1** ±2.5%	**68.3** ±2.6%
without part	51.5 ±3.0%	**80.5** ±1.2%	47.1 ±2.8%	54.7 ±5.5%	62.7 ±0.9%

Table 8: Transfer-learning of neural listeners in novel object *classes*: average accuracies *with* standard deviations (complementing Table 8, Main Paper). The sub-populations denote *entire*: all collected utterances, *known*: utterances containing *only* chair-training-vocabulary words, *with-part*: subset of *known*, with utterances containing at least one part-related word, *without-part* subset of *known* and complement of *with-part*. For reference the test-chair statistics are shown (first row) but not included in the reported average (last row).

Figure 4: **Examples of attention weights on human utterances**. The listener's LSTM appears to learn attention weights that emphasize the more informative words disambiguating the referent. For these results the *Baseline* listener is used and the attention-scores are extracted when the target object is grounding the LSTM.

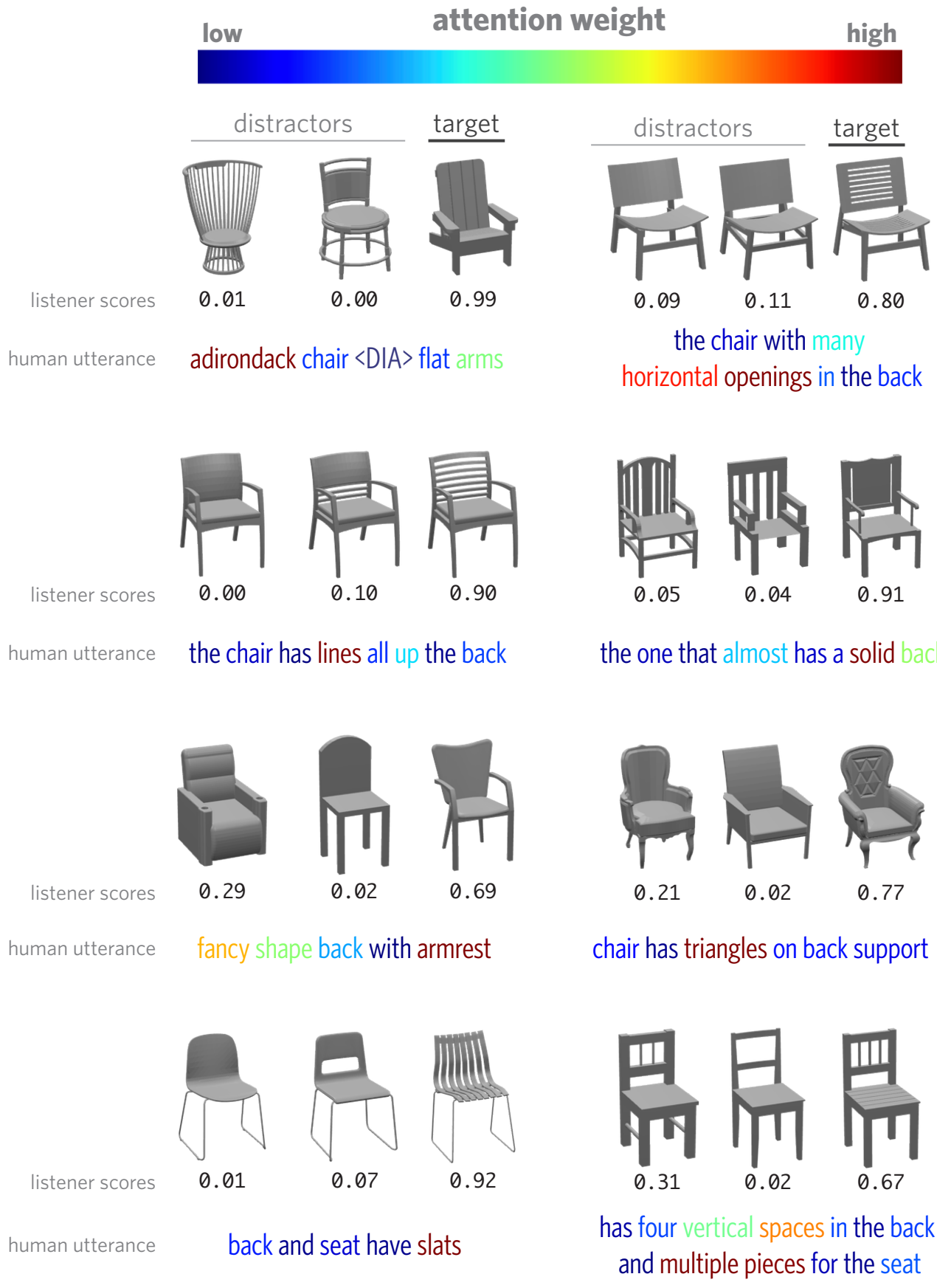


Figure 5: **Examples of lesioning all but the mentioned part**. Here, we show the response of a *Baseline* listener tested with visual representations of entire objects (left column, three chairs) vs. its response when it receives **only** the visual features corresponding to the referred semantic-part (right column). The corresponding utterance is shown left-most of each row. In these examples the listener assigns higher confidence to the actual target when the isolated parts are considered instead of the entire objects, implying that further performance gains can occur with an explicit part-aware visual attention mechanism.

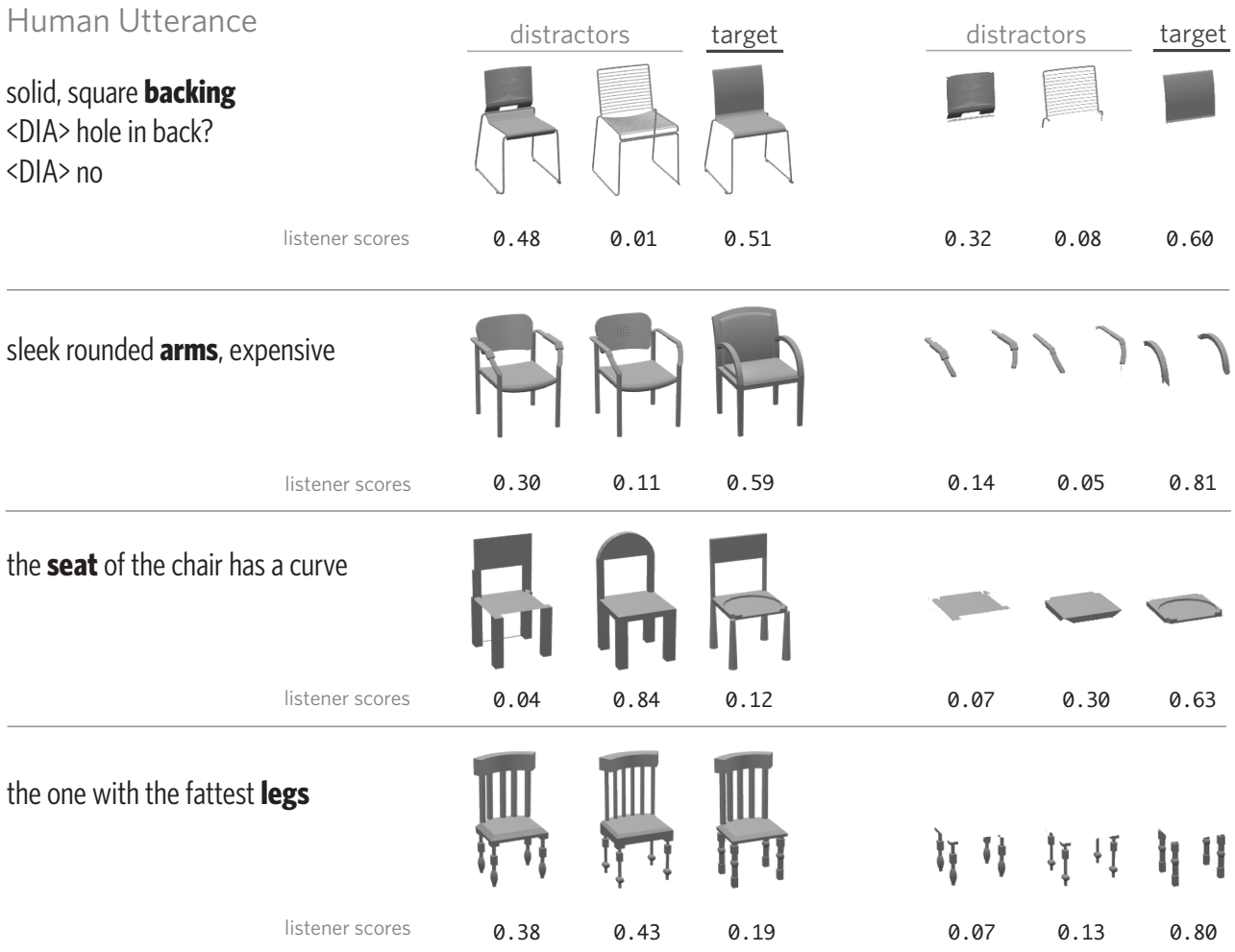
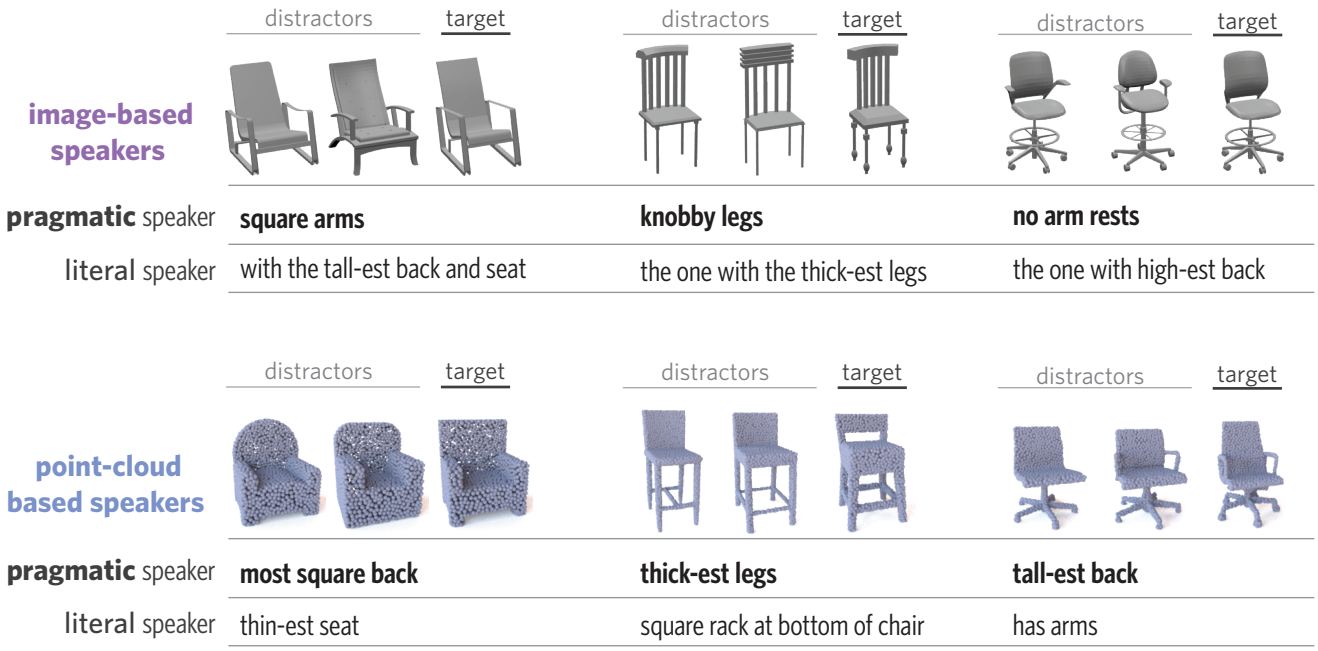
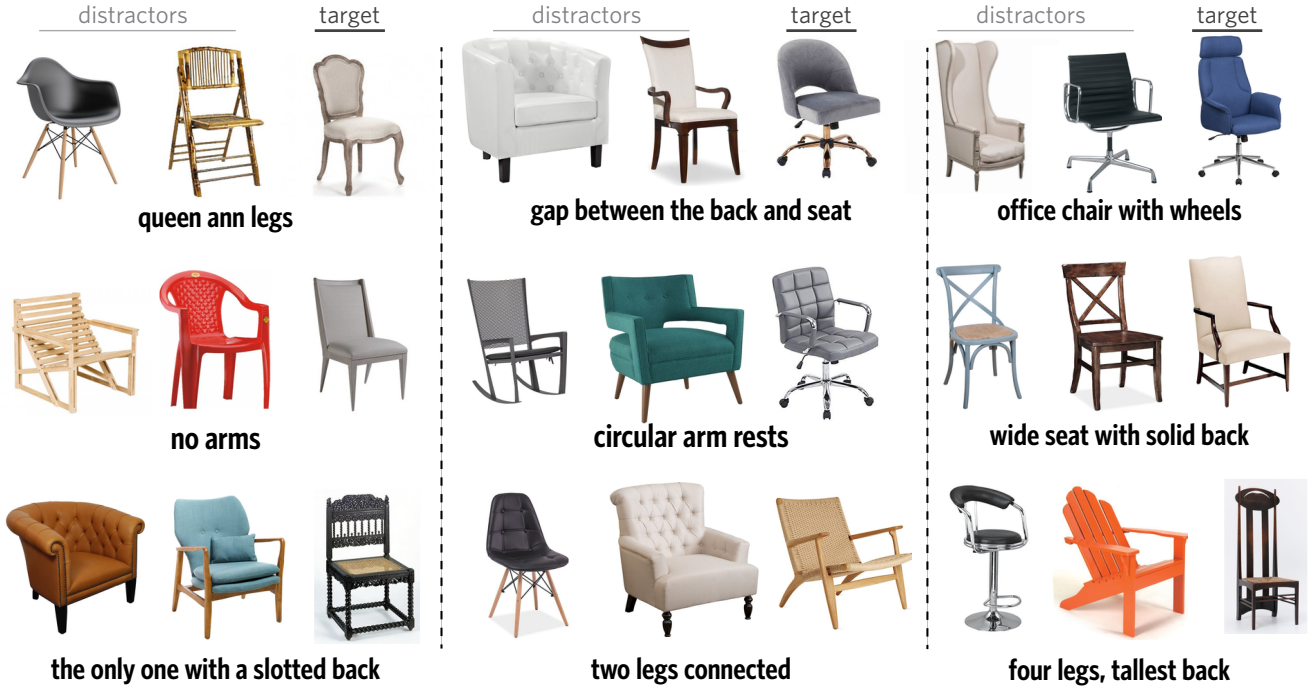


Figure 6: **Pragmatic vs. literal speakers for two modalities**. More examples of pragmatic vs. literal generations in Hard contexts. Tor-row includes examples from image-based speakers. Bottom-row from point-based ones.



(a) **Model generations with real images.** The top-scoring utterance of a pragmatic model is displayed under each context.



(b) **Human-utterance comprehension with unseen object classes.** The human utterance is color-coded according to the attention placed by a chair-trained listener who also evaluates the object-utterance compatibility (scores shown under its context).

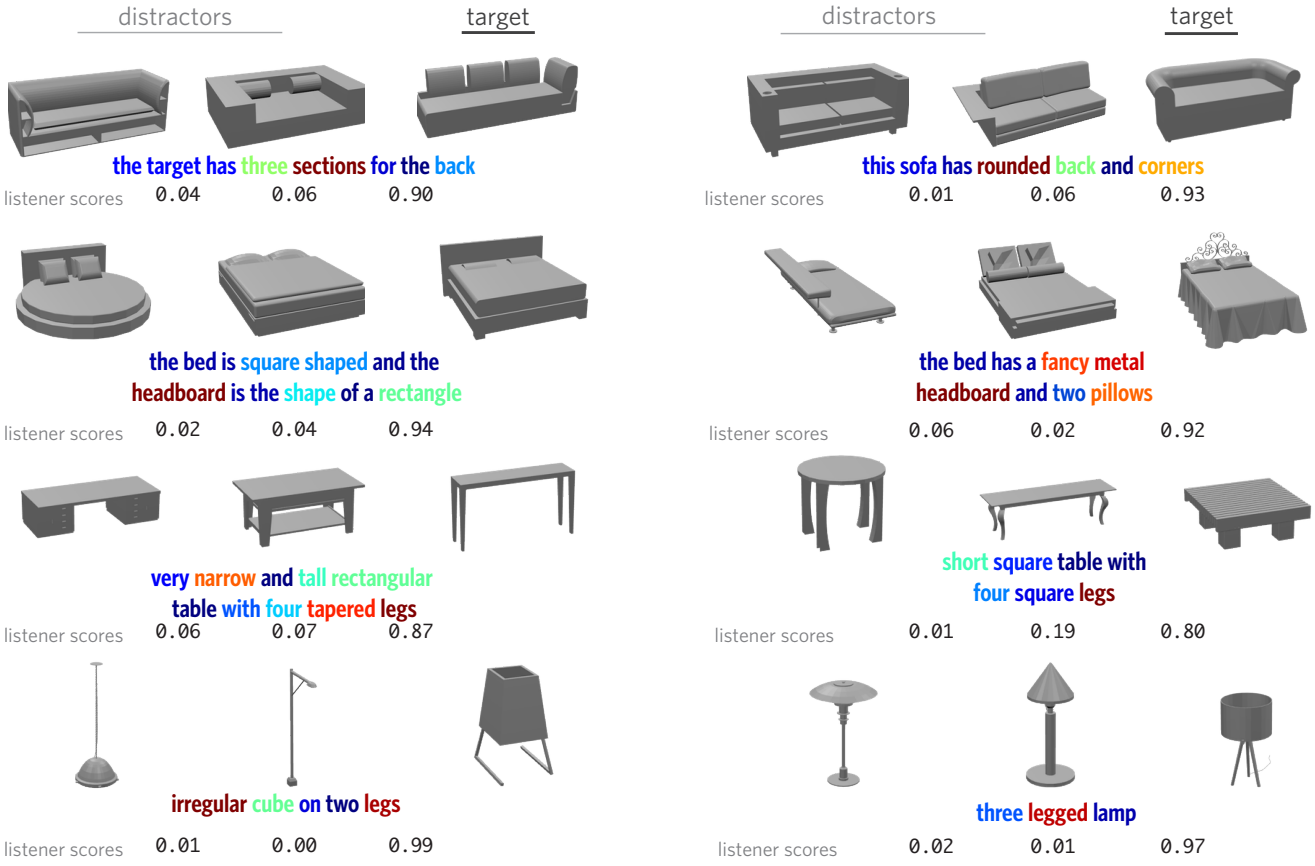
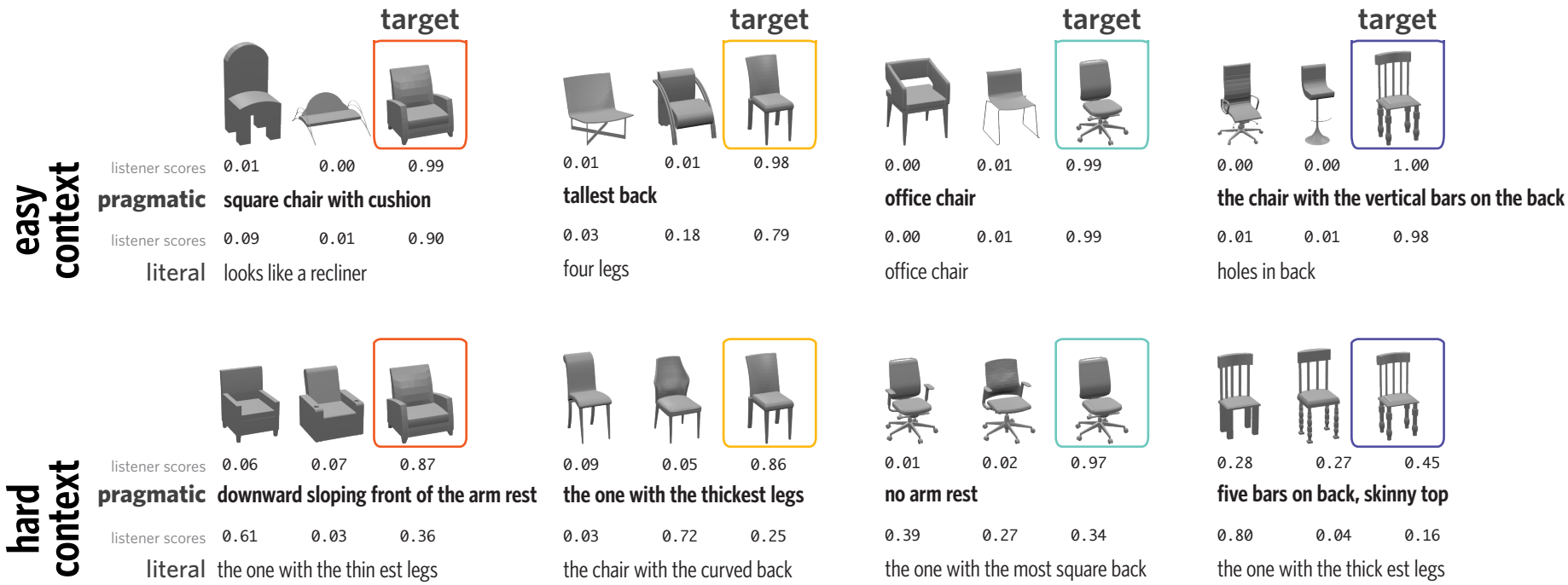
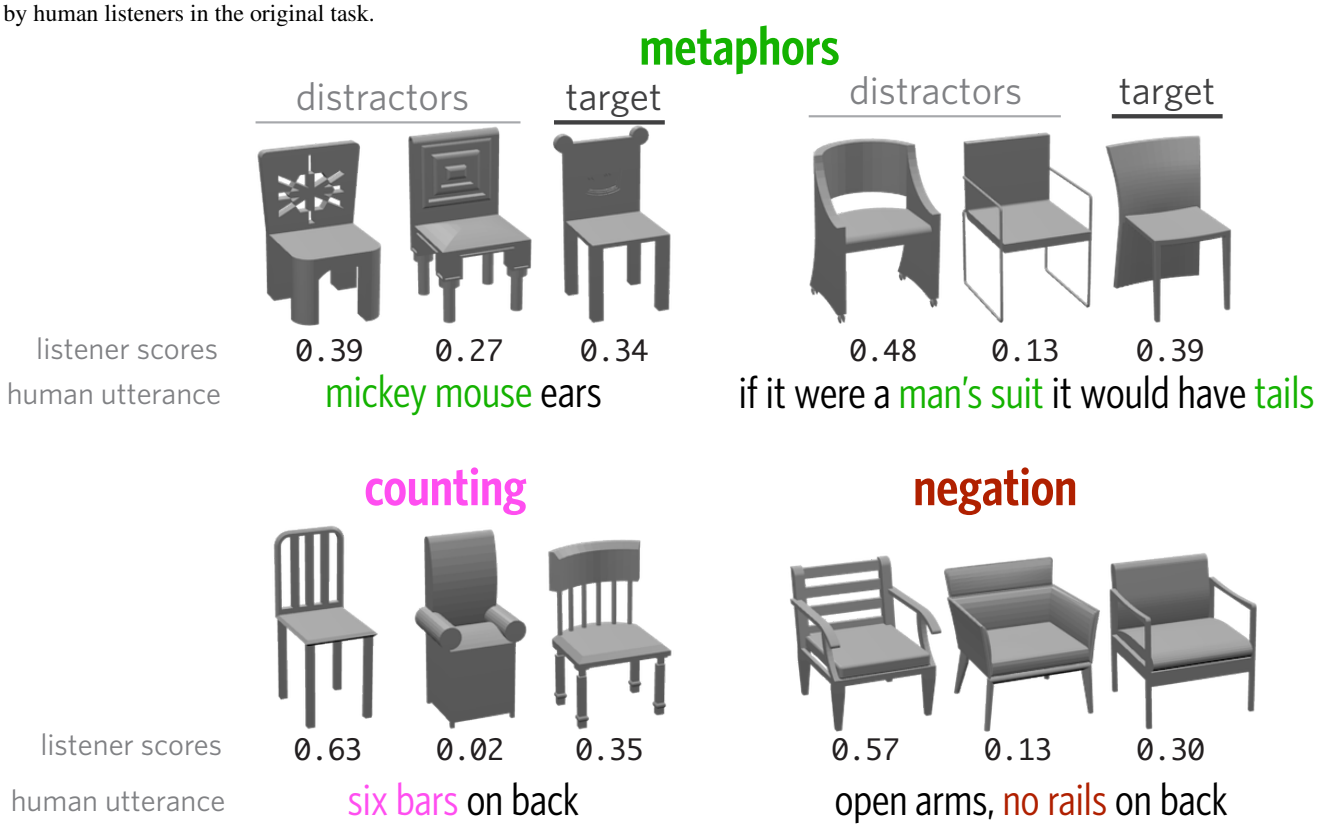


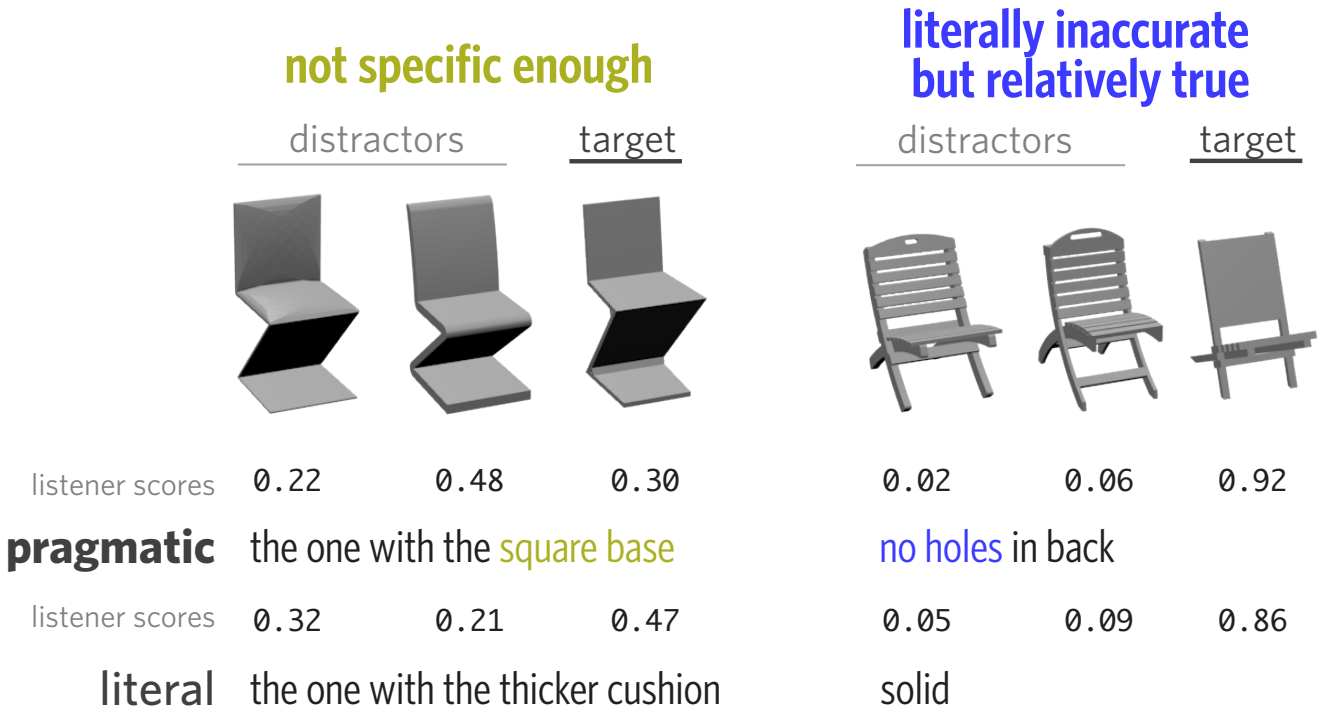
Figure 8: **Effect of context on production**: Synthetic utterances generated by a *literal* and *pragmatic* image-based speaker. The top and bottom rows show utterances produced for the same target in a Easy and Hard context, respectively. The *Baseline* (with point-clouds and images and attention) listener is used to predict the target and its confidence is displayed above each utterance. While both speaker models produce similarly effective utterances in Easy contexts, the literal speaker fails to produce effective utterances in Hard contexts.



(a) **Neural-listener failure cases**. Our top-performing listener model appears to struggle to interpret referential language that relies on metaphors, precisely counting parts, or (to a less degree) negations. All examples are drawn from the test set and were correctly classified by human listeners in the original task.



(b) **Neural-speaker failure cases**. Sometimes even the *pragmatic* speaker produces insufficiently specific utterances that mention only undiagnostic features, or produces utterances that are literally false of the target (e.g. there technically *is* a hole in the back) while still succeeding in distinguishing the objects.



Easy	**word**	office	sofa	regular	folding	wooden	stool	wheels	metal	normal	rocking
	pmi	-1.70	-0.94	-0.88	-0.84	-0.83	-0.79	-0.78	-0.71	-0.67	-0.66
Hard	**word**	alike	identical	thickness	texture	darker	skinnier	thicker	perfect	similar	larger
	pmi	0.69	0.67	0.67	0.66	0.65	0.64	0.63	0.62	0.62	0.61

Table 9: Most distinctive words in each context type according to point-wise mutual information (excluding tokens that appeared fewer than 30 times in the dataset). Lower numbers are more distinctive of Easy and higher numbers are more distinctive of Hard.

7. Miscellaneous

Each game consisted of 69 trials (unique triplets) and participants swapped speaker and listener roles with the conclusion of each trial. The game's interface is depicted in Figure 10. Participants were allowed to play multiple games, but most participants in our dataset played exactly one game (81% of participants). The most distinctive words in each triplet type (as measured by point-wise mutual information) are shown in Table 9).

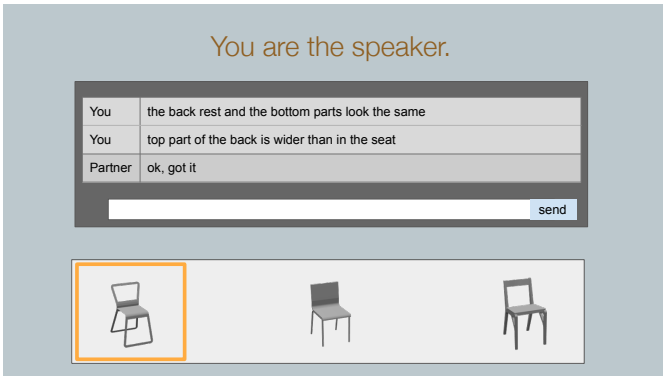


Figure 10: Reference game interface. Communication was natural without any system constraints being imposed.

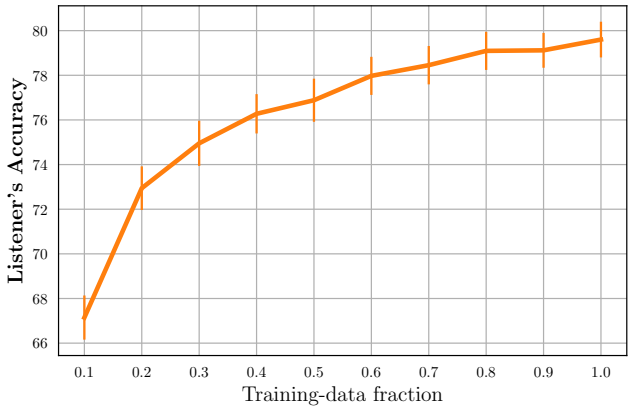


Figure 11: Listener's accuracy for different sizes of training data, under the *object* generalization task. The original split includes [80%, 10%, 10%] for training/test/val purposes, thus the maximum size of training data is 0.8 of the entire dataset corresponding to the value (fraction) 1.0 in the x-axis. The listener model uses the *Baseline* architecture with word attention, images and point-clouds and its accuracy is measured on the original (10%) test split. Results are averages of 5 random seeds controlling the original data split and the neural-net's initialization.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas J. Guibas. Learning representations and generative models for 3d point clouds. *Proceedings of the 35th International Conference on Machine Learning*, 2018. 2

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016. 3

[3] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository. *CoRR*, abs/1512.03012, 2015. 2

[4] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 2

[5] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015. 4

[6] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 4

[7] Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, 2013. 2

[8] Takeru Miyato, Dai M. Andrew, and Goodfellow Ian. Adversarial training methods for semi-supervised text classification. *International Conference on Learning*, 2017. 3

[9] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1), 2014. 3

[10] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015. 3