

Pragmatically Informative Color Generation by Grounding Contextual Modifiers

Zhengxuan Wu¹, Desmond C. Ong^{2,3}

¹Symbolic Systems Program, Stanford University

²Institute of High Performance Computing, Agency for Science, Technology and Research, Singapore

³Department of Information Systems and Analytics, National University of Singapore
wuzhengx@stanford.edu, dco@comp.nus.edu.sg

Abstract

Grounding language in contextual information is crucial for fine-grained natural language understanding. One important task that involves grounding contextual modifiers is color generation. Given a reference color “green”, and a modifier “bluey”, how does one generate a color that could represent “bluey green”? We propose a computational pragmatics model that formulates this color generation task as a recursive game between speakers and listeners. In our model, a pragmatic speaker reasons about the inferences that a listener would make, and thus generates a modified color that is maximally informative to help the listener recover the original referents. In this paper, we show that incorporating pragmatic information provides significant improvements in performance compared with other state-of-the-art deep learning models where pragmatic inference and flexibility in representing colors from a large continuous space are lacking.

1 Introduction

When describing colors, people rely on comparative adjectives such as “pale”, or “bright” (Lassiter and Goodman, 2017). Understanding how these comparative words modify the referent color, it requires extensive language grounding, in this case in color space (Monroe et al., 2017). For instance, understanding (i.e., correctly generating) a color that corresponds to “pale yellow” requires some knowledge about the meaning of “yellow” and “pale” in color space, and how the latter modifier modifies the former referent color. Understanding such modifiers and how they are grounded in image space is imperative for fine-grained attribute learning (Farhadi et al., 2009; Russakovsky and Fei-Fei, 2010; Vedaldi et al., 2014).

Color provides a simple and tractable space in which to study natural language grounding. Winn




Ref. Color		Modifier
[184, 71, 69]	“dirty”	
[124, 46, 153]	“dusty”	
[124, 46, 153]	“reddish”	

Figure 1: Examples of the color modification task, shown in RGB space. Given the reference color (Ref. Color), the modifier changes the color towards a target color. We represent this modification by going from left to right of the colored bar; the target color is shown at the right-end of the bar. The reference color for the top row is *green*, and *purple* for the middle and bottom rows.

and Muresan (2018) proposed a new paradigm focusing on using machine learning to produce grounded comparative adjectives in color description. As shown in Fig. 1, given a color “green” represented by the RGB vector [184, 71, 69] and a modifier “dirty”, the task is to generate RGB vectors for the color “dirty green”. Previous studies have trained deep learning models to generate modified colors by grounding comparative adjectives (Winn and Muresan, 2018; Han et al., 2019), which produce strong results, but these models lack a notion of pragmatics, which is important for modelling tasks that require extensive language grounding. In parallel, all previous works admit an important limitation where they ignore the richness of the color representation space by only considering a single RGB vector to represent a color label, which is not satisfying as a color label may be represented by a region in the color space.

In this paper, we propose adding *pragmatic informativeness* with leveraging with the richness of the color representations, a feature present in human contextualized color generation process, but which has been lacking in deep learning approaches to contextualized color modification tasks. The pragmatic approaches have been successfully applied to cognitive science and computational linguistics tasks such as reference games, where one has to understand pragmatics grounded in local context in order to correctly select the target in the presence of distractors (Andreas and Klein, 2016; Monroe et al., 2017; Cohn-Gordon et al., 2018, 2019; Nie et al., 2020). Specifically, we propose a *reconstructor-based pragmatic speaker* model that learns color generation via pragmatically grounding comparative modifiers.

Our work differs from most RSA-based models because we do not have a choice of distractors and referent targets: Rather, the task is to generate a target. Given a reference color (a vector in RGB color space) and a comparative modifier (e.g., “dirty”), a *literal speaker* model first generates modified colors using deep learning models as in (Winn and Muresan, 2018; Han et al., 2019). We kept our literal speaker close to previous, state-of-the-art deep learning models that learn a nonlinear mapping from words to (changes in) color space. Building on the literal speaker, we propose a listener that reasons about the literal speaker and tries to guess the original referent color. Finally, our *reconstructor-based pragmatic speaker* reasons about the listener, and produces modified colors that are maximally informative, which would help the listener correctly recover the referent. We find that our pragmatic speaker model performs significantly better than the state-of-the-art models on a variety of test scenarios ¹.

2 Data

We use the dataset² generated by Winn and Muresan (2018). This dataset was initially based on results of a color description survey collected by Munroe (2010) in which participants were asked to provide free-form labels for colors, giving a collection of mappings between color labels and colors. As a result, this alludes the fact that a single color label may be represented by different colors. This initial

¹Code is available at <https://github.com/frankaging/Pragmatic-Color-Generation>

²https://bitbucket.org/o_winn/comparative_colors

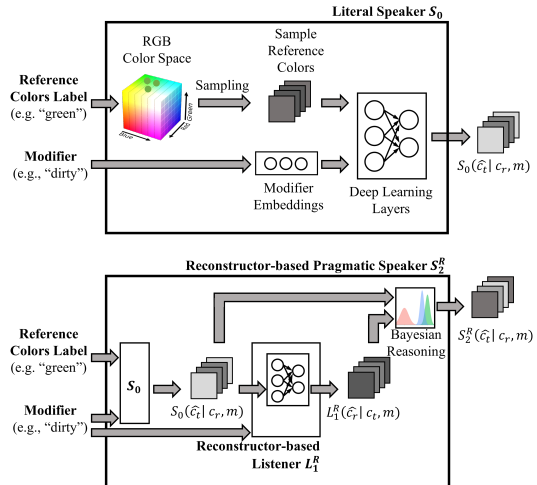


Figure 2: Model architecture for our *restructor-based pragmatic speaker* model, which consists a literal speaker model and a implicit reconstructor-based listener model.

survey was then cleaned and filtered by McMahan and Stone (2015), producing a set of 821 unique color labels, with an average of 600 RGB vectors per label. The dataset we use contains 415 triples, which includes 79 unique reference color labels and 81 unique modifiers. We later partitioned the dataset into Train, Dev and Test set as in Winn and Muresan (2018). See Appendix A for details.

3 Pragmatic Model

The task is formulated as a conditional color generation task. Given a set of sampled RGB vectors \mathbf{c}_r for a reference color label and a modifier m , our objective is to generate a RGB vector \hat{c}_t that is close to the gold-standard RGB vector c_t for the target color label. This process involves two main steps as shown in Fig. 2.

Literal Speaker We first describe the base *literal speaker* model, S_0 , which generates a probability distribution $S_0(\hat{c}_t | c_r, m)$ over target colors \hat{c}_t given a reference color $c_r \in \mathbf{c}_r$ and a modifier m (represented using word embeddings). To parameterize S_0 , we build our *literal speaker* model using the deep neural network proposed by Winn and Muresan (2018). See Appendix B for details.

To leverage samples per color label, we randomly sample n RGB vectors c_r for each color label, as \mathbf{c}_r .³ We pass each sample through the deep learning

³We randomly sample 100 RGB vectors for a particular color, and use the mean RGB values of these 100 RGB vectors as the RGB values for one sample. We iterate this process n

Test Set	Models			
	WM18	HSC19-HSV _{retrain}	S ₀	S ₂ ^R
Cosine Similarity (Std. Dev.); Higher = Better				
Seen Pairings (SP)	.680 (-)	.869 (.023)	.926 (.002)	.934 (.001)
Unseen Pairings (UP)	.680 (-)	.739 (.115)	.823 (.005)	.834 (.003)
Unseen Ref. Color (URC)	.400 (-)	.424 (.088)	.751 (.007)	.803 (.004)
Unseen Modifier (UM)	.410 (-)	.620 (.132)	.603 (.002)	.690 (.002)
Fully Unseen (FUN)	-.210 (-)	.408 (.099)	.552 (.011)	.564 (.009)
Overall (OR)	.650 (-)	.758 (.056)	.855 (.003)	.859 (.002)
Delta-E Distance (Std. Dev.); Lower = Better				
Seen Pairings (SP)	6.10 (-)	5.58 (.022)	5.03 (.012)	4.79 (.011)
Unseen Pairings (UP)	7.90 (-)	8.00 (.082)	5.79 (.008)	5.72 (.006)
Unseen Ref. Color (URC)	11.4 (-)	20.5 (1.33)	10.1 (.011)	9.49 (.013)
Unseen Modifier (UM)	10.5 (-)	16.0 (1.28)	13.2 (.072)	13.2 (.066)
Fully Unseen (FUN)	15.9 (-)	16.4 (.665)	15.0 (.011)	15.0 (.006)
Overall (OR)	6.80 (-)	8.96 (.324)	6.88 (.005)	6.57 (.003)

Table 1: Average cosine similarity score and Delta-E distance over 10 runs, of our literal speaker (S_0) and our reconstructor-based pragmatic speaker (S_2^R) in comparison to previous models: WM18 proposed by Winn and Muresan (2018), and HSC19-HSV proposed by Han et al. (2019). To facilitate comparison with our models, we retrained HSC19-HSV, with more details in the main text. (-) indicates that standard deviations were not reported in the original paper. **Bolded** values indicate the best performances.

layers, to generate multiple target colors. As a result, the model predicts n modified RGB vectors $\hat{c}_t \in \hat{\mathbf{c}}_t$ for a target color label. In our experiments, we set n to 10.

To evaluate generated \hat{c}_t via sampling, we formulate the probability distribution as $S_0(\hat{c}_t|c_r, m)$, by using the distances between RGB vectors in RGB space:

$$S_0(\hat{c}_t|c_r, m) \propto \frac{e^{\Delta(\hat{c}_t, \bar{\mathbf{c}}_r)}}{\sum_{\hat{c}'_t} e^{\Delta(\hat{c}'_t, \bar{\mathbf{c}}_r)}} \quad (1)$$

where $\bar{\mathbf{c}}_r$ is the mean RGB vector associated with the reference color label. See Appendix C for our formulation of distance function $\Delta(\cdot)$ in Eqn. 1 and Eqn. 2.

Reconstructor-Based Pragmatic Speaker To increase informativeness of our outputs from S_0 , we further propose a “listener” that maximizes the probability of reconstructing inputs from the outputs of our *literal speaker* model. The listener produces a probability distribution $L_1^R(c_r|\hat{c}_t, m)$ by reconstructing c_r given a \hat{c}_t and m . We parameterize L_1^R using a separate neural network which has the same model architecture as in S_0 trained

times.

with the objective of predicting c_r given c_t and word embeddings for m .

Similar to Equation 1, we formulate the probability distribution for $L_1^R(c_r|\hat{c}_t, m)$ by considering reconstruction error using the distance between RGB vectors in the RGB space:

$$L_1^R(c_r|\hat{c}_t, m) \propto \left(\frac{e^{\Delta(\hat{c}_r, \bar{\mathbf{c}}_r)}}{\sum_{\hat{c}'_r} e^{\Delta(\hat{c}'_r, \bar{\mathbf{c}}_r)}} \right)^{-1} \quad (2)$$

where \hat{c}_r is the reconstructed RGB vector for a reference color label. Unlike Equation 1, we have the probability proportional to the *inverse* of the distance function as we want the reconstructed reference color to be close to the true reference color.

Now, we introduce our pragmatic model, the *reconstructor-based pragmatic speaker* model, which combines the *literal speaker* and the *reconstructor-based listener* (Fig. 2). Formally, the reconstructor-based pragmatic speaker generates a probability distribution over \hat{c}_t , weighting both the L_1^R and S_0 terms:

$$S_2^R(\hat{c}_t|c_r, m) = L_1^R(c_r|\hat{c}_t, m)^\lambda \cdot S_0(\hat{c}_t|c_r, m)^{1-\lambda} \quad (3)$$

where λ is a pragmatic reasoning parameter that

Test Set	Ref.	Modifier	Target	Cos. Sim.	Delta-E
SP		"light seafoam"	HSC19-HSV _{retrain}	.882	5.33
			S_2^R	.974	4.99
UP		"faded"	HSC19-HSV _{retrain}	.826	6.43
			S_2^R	.872	6.41
URC		"sandier"	HSC19-HSV _{retrain}	.954	15.9
			S_2^R	.982	12.0
UM		"vibrant"	HSC19-HSV _{retrain}	-.185	9.21
			S_2^R	.223	5.63
FUN		"paler"	HSC19-HSV _{retrain}	.646	18.9
			S_2^R	.777	17.4

Figure 3: Examples of predictions for our retrained model (Han et al., 2019) HSC19-HSV, and our *reconstructor-based pragmatic speaker* model S_2^R . See Appendix H for details about retraining.

controls how much the model optimizes for discriminative outputs, following (Monroe et al., 2017). The predicted RGB vector with the highest probability in Eqn. 3 is our pragmatic prediction for \hat{c}_t . We found an optimal value of λ at 0.33 using grid search by evaluating with the *Validation* set.

4 Experiment Setup

We evaluate our models under 5 different test sets as in Winn and Muresan (2018); Han et al. (2019): Seen Pairings (SP), Unseen Pairings (UP), Unseen Reference Color (URC), Unseen Modifiers (UM), Fully Unseen (FUN) and Overall (OR) sets. See Appendix D to F for details about evaluation splits, model configurations and evaluation metrics.

5 Results

Table 1 shows test results, in comparison with previous best performing models. We retrain the best performing model HSC19-HSV (Han et al., 2019) with our evaluation pipeline to ensure fairness.

Compared to the both models, our literal model S_0 achieves better performance across most test cases while maintains similar performance in others as shown in the Table 1. This is expected given the fact that S_0 maximizes information gain via additional evaluating with sampling procedure. Our pragmatic model S_2^R further increases performances across all *Test* sets by outperforming our literal model S_0 with significant gains, which is expected given the model adds pragmatic informativeness. For all the test sets, S_2^R exceeds the state-of-the-art (SOTA) deep learning based models across

both evaluation metrics. For more difficult test cases where one of the input is unseen, our models perform extremely well. Fig. 3 shows examples from our models. See Appendix G for explanations and error analysis. We also investigated whether different distance functions in Eqn. 1 and Eqn. 2 affect our performance results, as in Appendix I.

6 Conclusion and Future Work

In this paper, we propose a novel *reconstructor-based pragmatic speaker* model, and apply it to a color generation task that requires extensive grounding of contextual modifiers in color space. Our base *literal speaker* model, with no pragmatics and which adapts previous deep learning models with distractor sampling, performs respectably on modifiers and colors that it was trained on, but performs poorly when testing on previously unseen modifiers and colors. Our pragmatic speaker model reasons about a listener that tries to guess the speaker’s input, and hence provides pragmatically informative utterances (colors). The output of the pragmatic speaker model shows consistent improvements in performance over the base literal speaker model across multiple testing conditions, unseen test words (e.g. unseen colors). We note that in this case (and in many other datasets as well), the data was generated by people—perhaps people may be applying some pragmatic reasoning when labeling colors. We speculate that this might both (i) explain the performance of the pragmatic speaker model, and (ii) justify the pragmatic assumptions we make in building our model.

References

- Jacob Andreas and Dan Klein. 2016. Reasoning about pragmatics with neural listeners and speakers. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1173–1182.
- Reuben Cohn-Gordon, Noah Goodman, and Christopher Potts. 2018. Pragmatically informative image captioning with character-level inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 439–443.
- Reuben Cohn-Gordon, Noah Goodman, and Christopher Potts. 2019. An incremental iterated response model of pragmatics. In *Proceedings of the Society for Computation in Linguistics (SciL) 2019*, pages 81–90.
- Ali Farhadi, Ian Endres, Derek Hoiem, and David Forsyth. 2009. Describing objects by their attributes. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1778–1785. IEEE.
- Xudong Han, Philip Schulz, and Trevor Cohn. 2019. Grounding learning of modifier dynamics: An application to color naming. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1488–1493.
- Daniel Lassiter and Noah D Goodman. 2017. Adjectival vagueness in a bayesian model of interpretation. *Synthese*, 194(10):3801–3836.
- M Ronnier Luo, Guihua Cui, and Bryan Rigg. 2001. The development of the cie 2000 colour-difference formula: Ciede2000. *Color Research & Application: Endorsed by Inter-Society Color Council, The Colour Group (Great Britain), Canadian Society for Color, Color Science Association of Japan, Dutch Society for the Study of Color, The Swedish Colour Centre Foundation, Colour Society of Australia, Centre Français de la Couleur*, 26(5):340–350.
- K McLaren. 1976. Xiii—the development of the cie 1976 (1* a* b*) uniform colour space and colour-difference formula. *Journal of the Society of Dyers and Colourists*, 92(9):338–341.
- Brian McMahan and Matthew Stone. 2015. A Bayesian model of grounded color semantics. *Transactions of the Association for Computational Linguistics*, 3:103–115.
- Will Monroe, Robert XD Hawkins, Noah D Goodman, and Christopher Potts. 2017. Colors in context: A pragmatic neural model for grounded language understanding. *Transactions of the Association for Computational Linguistics*, 5:325–338.
- Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gasic, Lina M Rojas Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Counter-fitting word vectors to linguistic constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 142–148.
- Randall Munroe. 2010. Color survey results. Online at <http://blog.xkcd.com/2010/05/03/color-survey-results>.
- Allen Nie, Reuben Cohn-Gordon, and Christopher Potts. 2020. Pragmatic issue-sensitive image captioning. *arXiv preprint arXiv:2004.14451*.
- Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.
- Olga Russakovsky and Li Fei-Fei. 2010. Attribute learning in large-scale datasets. In *European Conference on Computer Vision*, pages 1–14. Springer.
- Andrea Vedaldi, Siddharth Mahendran, Stavros Tsogkas, Subhransu Maji, Ross Girshick, Juho Kannala, Esa Rahtu, Iasonas Kokkinos, Matthew B Blaschko, David Weiss, et al. 2014. Understanding objects in detail with fine-grained attributes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3622–3629.
- Olivia Winn and Smaranda Muresan. 2018. ‘lighter’ can still be dark: Modeling comparative color descriptions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 790–795.

Appendix

A Data

We partition the datasets into different sets includes a *Train* set, a *Validation* set and a collection of *Test* sets. The *Train* set (271 triples) and the *Test* sets (415 triples, described in Sec. 4) are partitioned as in the original paper (Winn and Muresan, 2018). The *Validation* set (100 triples) contains triples sampled from the *Train* set for hyper-parameter tuning (not used for training). Additionally, RGB vectors of a color are partitioned across these three sets so that different sets use distinct RGB vectors to represent a color label; thus, if a color label (e.g. “red”) appears as a reference color in both training and testing sets, we will use different sets of RGB vectors for them. We use the same approach presented in Winn and Muresan (2018): using the mean value of a set of RGB vectors to represent the gold-standard RGB vector for a target color label for performance comparison. Note that color labels may contain adjective modifiers, such as “dirty green”. Winn and Muresan (2018) converted these mappings to triples of (reference color label, modifier, target color label), such as (“green”, “dirty”, “dirty green”), where both the reference color labels and target color labels are included in the original dataset.

B Literal Model

The model takes two inputs as shown in Fig. 4: the modifier m and the RGB vector c_r for the reference color label. An input modifier is represented by using 300-dimensional GloVe word embeddings (Pennington et al., 2014). We represent the modifiers as a bi-gram to account for comparatives that need “more” (e.g. “more vibrant”); single-word modifiers are padded with the zero vector. Both the word embeddings m of modifiers and the color RGB vector c_r are concatenated and fed into a fully connected feedforward layer with a hidden size 30:

$$f_\phi(c_r, m) = \mathbf{W}_1[c_r, m] + \mathbf{b}_1 \quad (4)$$

where \mathbf{W}_1 and \mathbf{b}_1 are learnt parameters. The output hidden state o_1 of the first layer is concatenated with the color RGB vector, and fed into another fully connected layer f_ψ with a hidden size 3 to predict the RGB vector \hat{c}_t for a target color label:

$$\begin{aligned} f_\psi(o_1, m) &= \mathbf{W}_2[o_1, c_r] + \mathbf{b}_2 \\ \hat{c}_t &= f_\psi(o_1, m) \end{aligned} \quad (5)$$

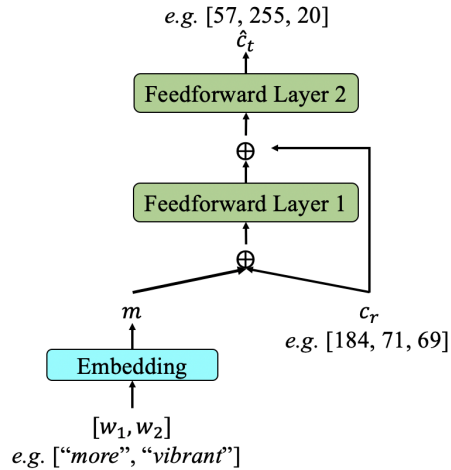


Figure 4: Model architecture for our base *literal speaker* S_0 model, which is based on the model proposed by Winn and Muresan (2018).

where \mathbf{W}_2 and \mathbf{b}_2 are learnt parameters. Following previous works (Winn and Muresan, 2018), our loss function has two parts: (1) minimizing the cosine distance between the “true” color modification in RGB space, $(c_t - c_r)$, and the predicted modification $(\hat{c}_t - c_r)$. (2) minimizing the mean square error between c_t and \hat{c}_t . Intuitively, the first part supervises the model to learn contextual meanings of modifiers, while the second part supervises the model to draw colors accurately.

C Distance Function

We propose that modified target color is expected to diverge from the reference color in the color space as in Winn and Muresan (2018). We use Delta-E 2000 in CIELAB color space as $\Delta(\cdot)$ in Eqn. 1 and Eqn. 2 (See Sec. 5 for the description and other distance metrics). The RGB vector with the highest probability is considered as the final output for S_0 . We use Delta-E 2000 in CIELAB which is also commonly used in previous works for evaluating model performances in color generation (Winn and Muresan, 2018) (See Sec. 4 for the definitions). Note that this distance measurements can be replaced by other metrics.

D Evaluation Splits

We evaluate our models under 5 different test sets as in Winn and Muresan (2018); Han et al. (2019): (1) 271 Seen Pairings (SP): The triple (reference color label, modifier, target color label) has been seen in the training set. (2) 29 Unseen Pairings

(UP): The reference color label and modifier have been seen in the training set, but not their pairing (i.e., they appeared separately). (3) 63 Unseen Reference Color (URC): reference color label is not present in the training set, but modifier is. (4) 41 Unseen Modifiers (UM): modifier is not present in the training set, but the reference color label is. (5) Fully Unseen (FUN): Neither reference color label or modifier has been seen in the training data. (6) Overall (OR): All samples in the testing set. As mentioned in Sec. 2, we use a different set of RGB vectors in the test sets for a color label that has been seen during training to ensure our model is not overfitting with the training set. We averaged our performance results over 10 runs with distinct random seeds.

E Model Configurations

To train our models, we use a standard laptop with 2.9 GHz Intel Core i7 and 16 GB 2133 MHz LPDDR3 with GPU training disabled. With this computing infrastructure, it takes about 10 minutes to train all of our models, where each model is trained with 500 epochs. Our *literal speaker* model S_0 contains 18,222 trainable parameters which is the same as *reconstructor-based listener* model L_1^R . Our reconstructor-based listener pragmatic speaker S_2^R contains 36,445 trainable parameters.

F Evaluation Metrics

We use two different metrics to evaluate our models, following Winn and Muresan (2018); Han et al. (2019): (1) Cosine similarity scores between the two difference vectors ($c_t - c_r$) and ($\hat{c}_t - c_r$); Higher scores correspond to better performance. (2) Delta-E distance in CIELAB color space between c_t and \hat{c}_t (see Winn and Muresan (2018); Han et al. (2019) for discussions). Delta-E is a non-uniformity metric for measuring color differences, which was first presented as the Euclidean Distance in CIELAB color space (McLaren, 1976). Luo et al. (2001) present the latest and most accurate CIE color difference metrics, Delta-E 2000, which improves the original formula by taking into account weighting factors and fixing inaccuracies in lightness. Lower Delta-E 2000 values correspond to better performance.

G Error Analysis

In Fig. 3, we provide selected illustrative examples from each of the five *Test* sets. We discuss the two cases from the UM and FUN test sets. For the

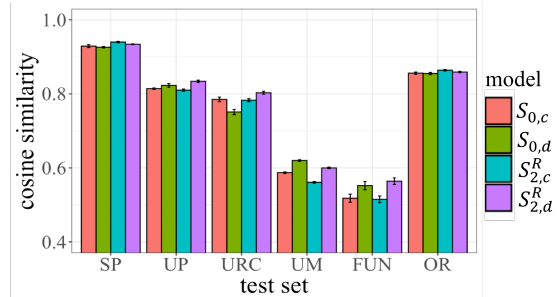


Figure 5: Test result for each *Test* set evaluated by cosine similarity with different distance-based probability estimations over 10 runs. Error bars represents standard deviation of the cosine similarity.

UM example, the reference color is “green” and the modifier is “vibrant”. Since in our training, the word embedding for “vibrant” is unseen, the non-pragmatic (literal) model results in a negative cosine similarity.

On the other hand, our pragmatic model S_2^R is capable of making more accurate predictions when presented with previously unseen test cases. This is expected as, by performing additional recursive evaluation, the pragmatic model is able to choose more “informative” target colors. This is consistent across all *Test* sets where we have unseen test examples. For FUN test set, both S_0 and S_2^R models make poor predictions. We expect that for unseen modifiers, the predictions should be based on modifiers with similar meanings. For example, “paler” should be similar to some seen modifiers, e.g. “whiten”, “faded”. However, the predicted modification is more close to “sandier” in this case. As described in Mrkšić et al. (2016), this may related to closeness in the word embedding space for these modifiers.

H Retraining

We note that in the original HSC19 paper, they computed the cosine similarity between vectors in HSV space and RGB space. We think this could have been an inadvertent mistake, and so we retrained HSC19-HSV and used cosine similarities between RGB vectors, which results in some differences in our table. Details can be found in their public code repository <https://github.com/HanXudong/GLoM>.

I Distance Metrics

As mentioned, we use Delta-E 2000 distance in the CIELAB color space as $\Delta(\cdot)$ in Eqn. 1 and Eqn. 2 by default. Additionally, we use cosine distance

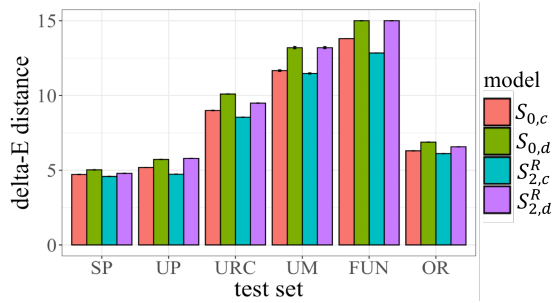


Figure 6: Test result for each *Test* set evaluated by Delta-E 2000 distance (Luo et al., 2001) in CIELAB color space (McLaren, 1976) with different distance-based probability estimations over 10 runs. Error bars represents standard deviation of the Delta-E 2000 distance.

in the RGB color space as our distance metrics to formulate our distance-based probability estimations. We denote $S_{0,c}$ and $S_{2,c}^R$ for cosine distance based estimation, and $S_{0,d}$ and $S_{2,d}^R$ for Delta-E 2000 based estimation. Results are shown in Fig. 5 and Fig. 6. Our result suggests that Delta-E 2000 based models perform better in most of test cases. This result corroborates with previous works suggesting Delta-E 2000 based distance in CIELAB color space is more accurate in describing nuanced difference between colors (Luo et al., 2001). As shown in Fig. 5 and Fig. 6, *reconstructor-based pragmatic speaker* models are consistently outperforming *literal speaker* models across multiple *Test* sets.