Estimating Intended Color from its name

Lyndon White, Roberto Togneri, Wei Liu,
Mohammed Bennamoun
lyndon.white@research.uwa.edu.au, roberto.togneri@uwa.edu.au,
wei.liu@uwa.edu.au,
mohammed.bennamoun@uwa.edu.au
The University of Western Australia. 35 Stirling Highway, Crawley, Western Australia

2018-06-07

Abstract

When a speaker says the name of a color, the color that they picture is not necessarily the same as the listener imagines. Color is a grounded semantic task, but that grounding is not a mapping of a single word (or phrase) to a single point in color-space. Proper understanding of color language requires the capacity to map a sequence of words to a probability distribution in color-space. A distribution is required as there is no clear agreement between people as to what a particular color describes – different people have a different idea of what it means to be "very dark orange".

Learning how each word in a color name contributes to the color described, allows for knowledge sharing between uses of the words in different color names. This knowledge sharing significantly improves predicative capacity for color names with sparse training data. The extreme case of this challenge in data sparsity is for color names without any direct training data. Our model is able to predict reasonable distributions for these cases, as evaluated on a held-out dataset consisting only of such terms.

1 Method

1.1 Tokenization

For all the term based methods, we perform tokenization. Tokenization

1.2 HSV color-space

2 Distribution Estimation

2.1 The conditional independence assumption

2.2 Discretization

For distribution estimation, our models are trained to output histograms. By making use of the conditional independence assumption Section 2.1, we output one histogram per channel.

2.3 Kernel-Density Based Smoothing

We make use of the **?**

2.4 Mean Squared Error on HSV

3 Experimental Setup

3.1 Implementation

The implementation of the CDEST and baseline models was in the Julia programming language (**?**). The full implementation is included in the supplementary materials. can be downloaded from the GitHub repository.[1] It makes heavy use of the MLDataUtils.jl[2] and TensorFlow.jl,[3] packages. the latter of which we enhanced significantly to allow for this work to be carried out.

3.2 Common Network Features

Dropout(**?**) is used on all layers, other than the embedding layer, with threshold of 0.5 during training. The network is optimized using Adam **?**, using a learning rate of 0.001. Early stopping is checked every 10 epochs using the development dataset. Distribution estimation methods are trained using full batch (where each observation is a distribution) for every epoch. Point Estimation trains using randomized mini-batches of size $2^1$6 observations (which are each color-space triples). All hidden-layers, except the penultimate layer in the point estimation network, have the same width 300, as does the the embedding layer.

[1] Implementation source is at `https://github.com/oxinabox/ColoringNames.jl`

[2] MLDataUtils.jl is available from `https://github.com/JuliaML/MLDataUtils.jl`

[3] TensorFlow.jl is available from `https://github.com/malmaud/TensorFlow.jl`

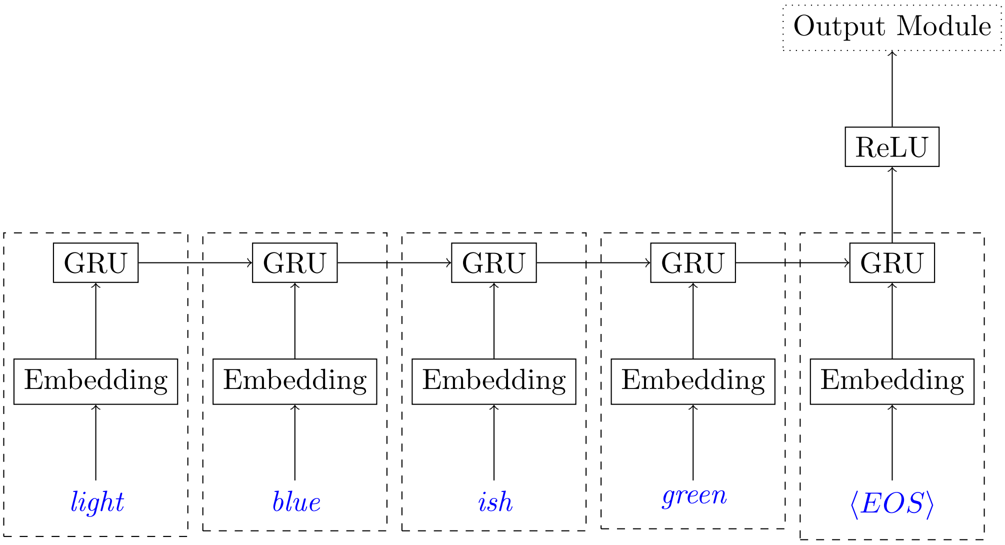

Figure 1: The RNN Input module for the example input `light greenish blue`. Each dashed box represents 1 time-step.

3.2.1 Embeddings

All our neural network based solutions incorperate an embedding layer. This embedding layer maps from tokenized words to vectors. We make use of 300d pretrained FastText embeddings ?[4].

The embeddings are not trained during the task, but are kept fixed. The layers above allow for arbitrary non-linear transform of the result.

3.3 Input Modules

3.3.1 Recurrent Neural Network(RNN)

A Recurrent Neural Network is a common choice for this kind of task, due to the variable length of the input. The general structure of this network, shown in ?? is similar to ?, or indeed to most other word sequence learning models. Each word is first transformed to an embedding representation. This representation is trained with the rest of the network allowing per word information to be efficiently learned. The embedding is used as the input for a Gated Recurrent Unit (GRU) (?). The output of the last time-step is fed to a Rectified Linear Unit (ReLU) (?).

3.3.2 Sum of Word Embeddings (SOWE)

Using a simple sum of word embeddings as a layer in a neural network is less typical than an RNN structure. Though it is well established as a

[4] Available from `https://fasttext.cc/docs/en/english-vectors.html`

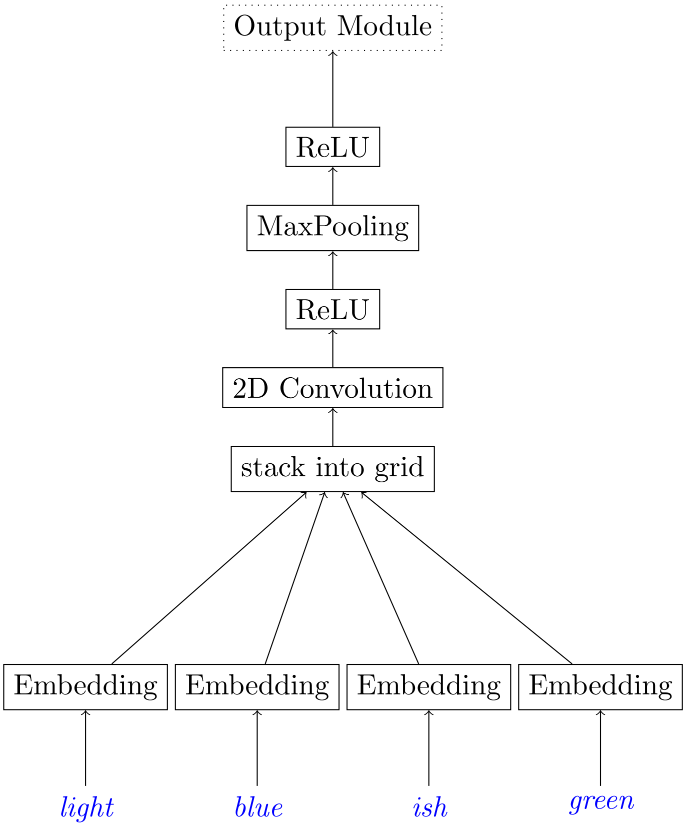

Figure 2: The SOWE input module for the example input `light greenish blue`

useful representation, and has been used an input to other classifiers such as support vector machines. Any number of word embeddings can be added to the sum. However, it has no representation of the order. The structure we used is shown in Figure 3

3.3.3 Convolutional Neural Network(CNN)

We apply a convulutional neural network to the task by applying 2D convolution over the stacked word embeddings. **??** We use 64 filters of size between 1 and the length of the longest padded embedding (5).

3.3.4 Non-term based Baseline

To baseline the performance of our models we propose

Histogram Baseline (Distribution Estimation, only)

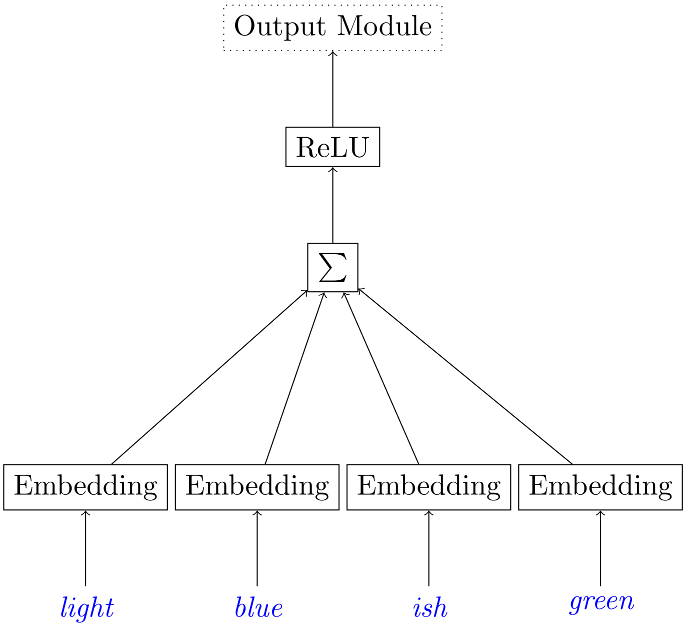

Figure 3: The SOWE input module for the example input `light greenish blue`

Mean Squared Error Baseline (Point Estimation, only)

3.4 Datasets

3.4.1 Full Training and Testing set

3.4.2 Extrapolation Training and Testing Set

The primary goal in constructing using the term based models is to be able to make predictions for never before seen descriptions of colors. For example, based on the learned understanding of `salmon` and of `bright`, from examples like `bright green` and `bright red`, we wish for the systems to make predictions about `bright salmon`, even though that description never occurs in the training data. To evaluate this generalisation capacity, we define an extrapolation sub-dataset for both testing and training. This is defined by selecting the rarest 100 color descriptions from the full dataset, with the restriction that every token in a selected description must still have at least 8 uses in other descriptions in the training set. The selected examples include multi-token descriptions such as: `bright yellow green` and also single tokens that occur more commonly as modifiers than as standalone descriptions such as `pale`.

The extrapolation training set is made up of the data from the full training set, excluding those corresponding the the rare descriptions. Similar is done for the development set, so as no direct knowledge of the combined terms can leak during early-stopping. Conversely, the extrapolation test set is made up of only the observations from the full test set that do use those rare descriptions.

By training on the extrapolation training set and testing on the extrapolation test set, we can assess the capacity of the models to make predictions for color descriptions not seen in training. A similar approach was used in ?. We contrast this to the same models when trained on the full training set, but tested on the extrapolation test set, to see how much accuracy was lost.

3.5 Order Testset

It is known that the order of words in a color description to some extent matters. `greenish brown` and `brownish green` are distinct, if similar, colors. To assess the models on there ability to make predictions when order matters we construct the order testset. This is a subset of the full test set containing only descriptions with terms that occur in multiple different orders. There are 76 such descriptions in the full dataset. Each of which has exactly one alternate ordering. This is unsurprising as while color descriptions may have more than 2 terms, normally one of the terms is a joining token such as `ish` or `-`.

3.6 Output Modules

3.6.1 Distribution Estimation

For all the distribution estimation systems we investigate here, we consider training both on the binned-data, and on the smoothed data (as described in Section 2.3). Making use of the conditional independence assumption (see Section 2.1), we output the three discretized distributions. This is done using 3 softmax output layers.

The output module for distribution estimation

Contrasting to estimating continuous conditional distributions, estimating a discrete conditional distributions is a significantly more studied application of neural networks – this is the basic function of any softmax classifier. To simplify the problem, we therefore transform it to be a discrete distribution estimation task, by discretizing the color-space. Discretization to a resolution of 64 and 256 bins per channel is considered.

For the case of the machine learning models, the output is produced using softmax layers.

3.6.2 Point Estimation

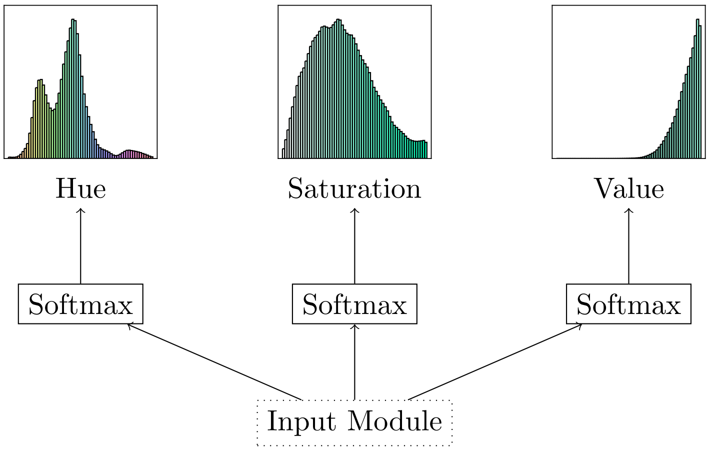

Figure 4: The Distribution Output Module

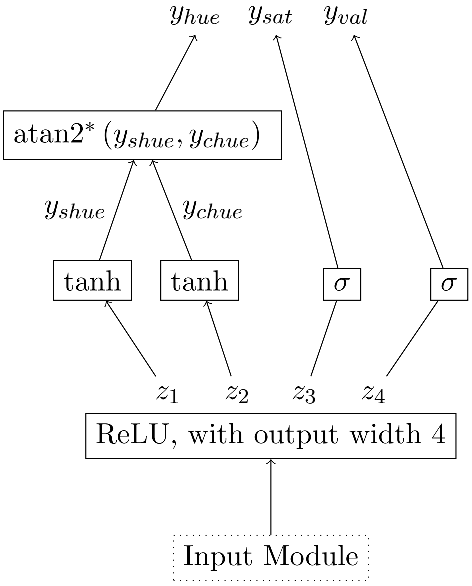

Figure 5: The Point Estimate Output Module. Here atan2* is the quadrant preserving arctangent, outputting as a regularized angle (as per in all evaluations)

A On the Conditional Independence of Color Channels given a Color Name

As discussed in the main text, we conducted a superficial investigation into the truth of our assumption that given a color name, the distributions of the hue, value and saturation are statistically independent.

We note that this investigation is, by no means, conclusive though it is suggestive. The investigation focusses around the use of Spearman's rank correlation. This correlation measures the monotonicity of the relationship between the random variables. A key limitation is that the relationship may exist but be non-monotonic. This is almost certainly true for any relationship involving channels, such as hue, which wrap around. In the case of such relationships Spearman's correlation will underestimate the true strength of the relationship. Thus, this test is of limited use in proving the conditional independence. However, it is a quick test to perform and does suggest that the conditional independence assumption may not be so incorrect as one might assume.

For the Monroe Color Dataset training data given by $V \subset \mathbb{R}^3 \times T$, where $\mathbb{R}^3$ is the value in the color-space under consideration, and T is the natural language space. The subset of the training data for the description $t \in T$ is given by $V_{|t} = \{(\tilde{v}_i,\, t_i) \in V \;\mid\; t_i = t\}$. Further let $T_V = \{t_i \;\mid\; (\tilde{v}, t_i) \in V$ be the set of color names used in the training set. Let $V_{\alpha|t}$ be the α channel component of $V_{|t}$, i.e. $V_{\alpha|t} = \big\{v_\alpha \mid ((v_1, v_2, v_3), t) \in V_{|t}\big\}$.

The set of absolute Spearman's rank correlations between channels a and b for each color name is given by $S_{ab} = \big\{\big|\rho(V_{a|t}, V_{b|t})\;\big|\; t \in T_V\big\}$.

Color-Space	$Q3(S_{12})$	$Q3(S_{13})$	$Q3(S_{23})$	max
HSV	0.1861	0.1867	0.1628	0.1867
HSL	0.1655	0.2147	0.3113	0.3113
YCbCr	0.4005	0.4393	0.3377	0.4393
YIQ	0.4088	0.4975	0.4064	0.4975
LCHab	0.5258	0.411	0.3688	0.5258
DIN99d	0.5442	0.4426	0.4803	0.5442
DIN99	0.5449	0.4931	0.5235	0.5449
DIN99o	0.5608	0.4082	0.5211	0.5608
RGB	0.603	0.4472	0.5656	0.603
Luv	0.5598	0.6112	0.4379	0.6112
LCHuv	0.6124	0.4072	0.3416	0.6124
HSI	0.2446	0.2391	0.6302	0.6302
CIELab	0.573	0.4597	0.639	0.639
xyY	0.723	0.5024	0.4165	0.723
LMS	0.968	0.7458	0.779	0.968
XYZ	0.9726	0.8167	0.7844	0.9726

Table 1: The third quartile for the pairwise Spearman's correlation of the color channels given the color name.

We consider the third quartile of that correlation as the indicative statistic in Table 1. That is to say for 75% of all color names, for the given color-space, the correlation is less than this value.

Of the 16 color-spaces considered, it can be seen that the HSV exhibits the strongest signs of conditional independence – under this (mildly flawed) metric. More properly put, it exhibits the weakest signs of non-independence. This includes being significantly less correlated than other spaces featuring circular channels such as HSL and HSI.

Our overall work makes the conditional independence assumption, much like n-gram language models making Markov assumption. The success of the main work indicates that the assumption does not cause substantial issues.