Weakly supervised learning of semantic colour terms

David Hanwell, Majid Mirmehdi

Visual Information Laboratory, Department of Computer Science, University of Bristol, Bristol, UK
E-mail: David.Hanwell@Bristol.ac.uk

Abstract: Recognition of visual attributes in images allows an image’s information content to be expressed textually. This has benefits for web search and image archiving, especially since visual attributes transcend language barriers. Classifiers are traditionally trained using manually segmented images, which are expensive and time consuming to produce. The authors propose a method which uses raw, noisy and unsegmented results of web image searches, to learn semantic colour terms. They use probabilistic graphical models on continuous domain, both for weakly supervised learning, and for segmentation of novel images. Experiments show that the authors methods give better results than the current state of the art in colour naming using noisy, weakly labelled training data.

*Note: Colour figures are available in the online version of this paper.

1 Introduction

Much work has been done in computer vision on recognising objects or the categories to which they belong. More recently, focus has broadened to recognising semantic attributes of objects and image regions. The distinction between these is perhaps most easily understood in terms of language: Objects are things which correspond to nouns, whereas semantic attributes are those which correspond to adjectives, such as colours (e.g. red, blue), textures (e.g. rough, smooth) and patterns (e.g. stripy, spotty). Recognition and modelling of such visual semantic attributes has several important applications. Firstly, modelling both attributes and objects together can increase the performance of object categorisation [1], as can exploiting correlations between object and attribute occurrence [2]. Some, such as [3, 4, 5], take advantage of correlations between attributes to recognise non-visual attributes, such as ‘fast’, ‘weak’ or ‘eats fish’. Secondly, because attributes transcend object classes, they allow fewer training examples to be used [4, 5], and even allow the generation of textual descriptions of previously unseen objects, or the learning of new categories from few [6] or no training examples [3, 5]. Thirdly, and perhaps most importantly, is automated annotation of large image sets. Image search engines predict an image’s semantic content using text appearing near the image or in its file name. For example, in some cases, such as online retail, there will be a high correlation between an image’s content and its textual description. However, such descriptions will probably not fully convey the entire semantic content of the image, will not use multiple synonyms of a term and are typically written in only a single language. This means that only a small subset of the terms that could be used to describe the image will actually be associated with it, for example, if we search for ‘Draper 30-Litre Dry Bag’ on Amazon UK, which is a very blue object, we see the word ‘blue’ appears nowhere on its page and searching again for ‘blue 30 L bag’, ‘blue 30 L waterproof bag’ or ‘blue dry bag’ does not return the item, although it does return other blue bags. Automated image annotation would allow a much larger proportion of an image’s information content to be expressed textually. This would allow (a) more accurate search results using more specific search terms, (b) searching web pages using cues from their images and (c) images to be found using terms in languages other than that of their host web-page. Much of the previous work in recognising visual attributes is highly supervised, in that it uses datasets which have been manually annotated and segmented, either by the authors themselves [7], or using services such as Amazon’s Mechanical Turk [1, 5] which farms out the task to private individuals. Compiling such datasets is expensive, time consuming and if too few human labellers are used, may be biased. Furthermore, learning a multitude of terms, rather than simply the most basic ones, would require a gargantuan feat of manual image segmenting and labelling. Recent work [8] proposed to use ImageNet [9], a large collection of images of many objects and concepts, each with a textual description. A more readily scalable approach is to use the results of a web image search engine such as Google Image search. This would provide cheap examples of a large number of attributes, in any language. Its results are aggregated from many websites, and hence are not biased by any one person’s idea or notion of an attribute (although they may conceivably be biased by assumptions made by the search algorithm). For example, van de Weijer et al. [10] conclude that learning colours from web image search results yields substantially better models than more supervised methods, where humans label hundreds of coloured chips from a predetermined set of names. Although this is an almost effortless method of building a dataset, the resulting images cannot ‘all’ be guaranteed to contain the attribute intended. Even those images which do
contain at least one image region with the attribute will probably also contain other regions without it. For this reason we describe the data as being ‘weakly labelled’ and refer to the learning task as ‘weakly supervised’. Training data obtained using Mechanical Turk may contain noise from the labelling process, and so even this could be seen as weakly labelled.

One of the first works to consider weakly supervised learning of visual semantic attributes was [11], which considers two types of attribute; unary (an attribute as a property of a single segment such as ‘red’) and binary (an attribute as a property of a pair of segments such as ‘stripy’). Two datasets are used; a noisy positive set of images with the attribute, and a negative set of images without the attribute. Attributes are learned by maximising a likelihood ratio using an expectation maximisation (EM)-like algorithm. The principal drawback of this approach is its reliance on a negative set. In their experiments, the authors used for the negative set of an attribute, the combined positive sets of other attributes to be learned. However, this would only work when the attributes are mutually exclusive. For example, while learning the colours scarlet, red and crimson, the assumption that a colour is more strongly represented in its positive set than the negative set may no longer be true.

Another example of weakly supervised attribute learning is the work of van de Weijer et al. [10, 12], in which modified versions of probabilistic latent semantic analysis (PLSA) [13] are used to train simple colour models from weakly labelled data. It is against this work that we compare our own methods. In [12], the three-dimensional feature space (CIE L*a*b*) is divided into 4000 sub-cubes, which are then used as the words in a document/topic model, where the colours are topics and the images are documents. This is the most weakly supervised of previous works, requiring as input only the 11 sets of images returned from searching Google Images for 11 colour terms. In [10], the same domain of sub-cubes is used, however, this time a standard PLSA model is modified with a regularisation term and prior distributions to allow the inclusion of the data labels. The parameters of the regularisation term and the priors are set using a hand-segmented validation set, thus making the supervision rather less weak than that of [12]. In both of these works, the authors consider only ‘basic colour terms’ as defined by Berlin and Kay [14], which do not have a significant overlap. In particular, they use the following set of 11 basic colour terms

\[ \text{Colour set of } [15, 16]: \text{ black, blue, brown, grey, green, orange, pink, purple, red, white, and yellow.} \]

which they assume will fill the entire Red Green Blue (RGB) colour cube. As with [11], this is unsuitable if we want to learn the overlapping attributes. For example, we would like to be able to train a system to recognise, ‘red’ as well as ‘burgundy’, ‘salmon’ and ‘maroon’.

One limitation of the methods of van de Weijer et al. is the lack of scalability of their discrete colour domain. If we wish to learn more complex attributes than simply colours, more features must be incorporated. Even if we rather crudely divide each feature, or dimension, into ten, then nine dimensions would require a billion such ‘sub-cubes’ – an unwieldy representation, especially if we are to learn many attributes. Although their assumption that 11 basic colour terms fill the entire colour cube seems reasonable, it is not clear how this may be extended to higher dimensional feature spaces. Ideally, we would like to learn a set of attributes without worrying whether or not we have compiled a complete set of terms.

A final example of learning visual attributes in a weakly supervised setting is [15], in which visual attributes and object classes are learned concurrently using images collected from web image searches on the term ‘attributes’ + object’. The method used is not as weakly supervised as others described here, as it requires false positives to be manually removed from the training set. We propose a method of learning colour terms from weakly labelled Google Image search results, which is readily extendible to higher dimensional feature spaces. Our method can be used to learn the overlapping attributes, and does not require a complete set of attributes to be specified. It has no parameters and hence requires no validation set. Furthermore, it does not require the specification of a negative set. We show that our method produces state of the art results in semantic colour naming from weakly labelled training data, and present both quantitative and qualitative results to demonstrate this.

The method we propose is immediately applicable to higher dimensional feature spaces, simply by using a higher dimensional feature (such as a texture feature), and correspondingly high-dimensional Gaussian mixture models. This is especially true when compared against [10, 12]. Here, we concentrate on learning colour, both in order to have work against which to compare our own, and to provide a proof of concept of our method.

In Section 2, we describe our model used to learn colours from weakly labelled data. In Section 3, we explain how our trained models are used to assign colour names to pixels of novel images. Section 4 details the experiments we performed using our models, and presents the results of these, comparing them against previous work. Finally, in Section 5, we conclude the paper, and briefly discuss the direction of future work.

2 Proposed model

We begin with a set of $D$ colour terms which we wish to learn. We extend each colour term separately into Google Image search, and take every one of the first 100 of the resulting images for each. The resulting $D$ datasets are very noisy. Not every image contains a region of the specified colour, and even in those which do, there are regions which do not. Fig. 1 shows some examples of the results of a search for ‘blue’ on Google Images. The datasets contain an average of 68 million pixels each. To make learning more tractable, we uniformly sample $N = 200,000$ pixels from each set. The L*a*b* values of these pixels are then our training data. We use the CIE L*a*b* colour space [16] because of its perceptual uniformity, which means that the Euclidean distances in the colour space are proportional to the corresponding difference in colour perceived by a human observer. Tests showed that models learned using this colour space gave better results than either the RGB or Hue Saturation Lightness (HSL) colour spaces. We denote the data samples of model $d$ (with $d \in \{1, \ldots, D\}$) by $X^d = (x^d_1, \ldots, x^d_D)$, with $x^d_i$ being the feature vector of the data sample $i$ of colour $d$.

We do not know which image areas contain a colour or even what proportion of data has a colour, therefore we model these aspects using latent variables. We refer to the pixels that do not have the colour as background, and those
which do as foreground. For each dataset, \( x^d \), we introduce a set of latent indicator variables, \( z^d \), such that

\[
z^d_i = \begin{cases} 1 & \text{if } x^d_i \text{ is foreground} \\ 0 & \text{if } x^d_i \text{ is background} \end{cases}
\]

Although we do not know what is the background, we infer its distribution by assuming, as in [12], that each dataset has the same, or similar distribution of background colours. More specifically, we model each dataset as a mixture of its foreground distribution, and a common background distribution. To this end, we also introduce the vector \( \pi = (\pi_1, \ldots, \pi_D) \) of latent mixture weights, where \( \pi_d = P(z^d_i = 1) \) represents the probability that a pixel of set \( d \) belongs to the foreground class. By modelling the data as a mixture in this way, this background distribution is simultaneously learned, and ‘subtracted’ from the noisy input data’s distributions, yielding the foreground models. Given a pixel \( x^d \) from set \( d \in \{1, \ldots, D\} \), we model it by

\[
P(x^d|\theta_d) = \pi_d M_d(x^d, \theta_d) + (1 - \pi_d) B(x^d, \theta_B)
\]

where \( M_d(x^d, \theta_d) \) represents the model we wish to learn (the foreground model), \( B(x^d, \theta_B) \) represents the model of the background, \( \theta_d \) are the parameters of foreground model \( d \), and \( \theta_B \) are the parameters of the background. Fig. 2 shows a graphical representation of this generative model.

We use no prior on \( \pi \), preferring not to make unnecessary assumptions about data composition, and to keep our model as general as possible. Furthermore, we avoid requiring extra validation data to fit prior parameters, as in [10]. Since we are trying to find attributes corresponding to the semantic concepts (colour terms), we do not leave attribute discovery to the model as the topics are in PLSA-like models. Larlus and Jurie [17] noted that topics estimated by their PLSA model tended not to agree with true classes. Furthermore, we do not propose to discover visual attribute terms, but concentrate on learning to recognise given attributes in novel images. We consider visual attribute discovery to be a distinct line of research, for example, [18, 19] which attempt to assess the ‘visualness’ of terms to determine whether they correspond to visual attributes.

Each \( M_d \) and \( B \) is a mixture of Gaussians in the colour space. \( B \) has at least as many components as there are datasets \( (D) \), in order to accurately model the background distribution. Each \( M_d \) has between 2 and 5 Gaussian components to avoid over-fitting. This leads to a compact representation of each colour. We fit our model using a standard EM algorithm to determine the values of parameters \( \theta_B \), and \( \theta_d \) and \( \pi_d \) for each \( d \).

### 3 Application to novel images

If our learnt colour models described distinct, non-overlapping colours, then segmenting a new image into colour regions may be as simple as using a maximum a posteriori (MAP) estimate to determine to which colour category each pixel belongs. However, because our models are potentially overlapping, each pixel’s value alone will not be sufficient to determine the colour of the object or image region to which it belongs. For example, an image region considered to be turquoise may well contain pixels which by themselves would be considered cyan. To avoid wrongly classifying these pixels as cyan, we need to be aware of the distribution of similar colours throughout the rest of the image. As classification ‘purely’ by region (rather than by pixel) would rely too heavily on a segmentation algorithm, we classify by pixel, but take into account the distribution of colours in the image as a whole.

We begin by introducing some notation and fundamental results used later. Suppose we have some data in a \( K \)-dimensional feature space, which we assume to be Gaussian with PDF \( f \), mean \( \mu_f \) and covariance matrix \( \Sigma_f \). Suppose further that we have a model of a class or category, also modelled as a Gaussian in the same space, with PDF \( g \), mean \( \mu_g \) and covariance matrix \( \Sigma_g \). We then ask; what is the expected PDF of data with this distribution, under this model? This is simply the functional

\[
\tilde{f}(f, g) = \int_{\mathbb{R}^K} f(x) g(x) \, dx
\]

The advantage of using Gaussians here is that they are closed under multiplication, allowing us to easily obtain

\[
\tilde{f}(f, g) = \frac{\Sigma_f (2\pi)^{-K} e^{-\frac{1}{2} (\mu_f - \mu_g)^T \Sigma_f^{-1} (\mu_f - \mu_g)}}{\sqrt{\det(\Sigma_g)}}
\]

Fig. 1 Some of the images from a search for ‘blue’ on Google images

Fig. 2 Generative model used to learn attributes from noisy data
where

\[
\begin{align*}
\mu_{jg} &= \Sigma_{ig}^{-1} \Sigma_{ig}^{-1} \mu_j + \Sigma_{ig}^{-1} \mu_g \\
\Sigma_{ig} &= (\Sigma_{ig}^{-1} + \Sigma_{ig}^{-1})^{-1}
\end{align*}
\]  

This result is also easily extendible to our case, in which both the data and model are represented by ‘mixtures’ of Gaussians.

We model an image’s colour content as a mixture of \( \tilde{C} \) Gaussian clusters, \( C_i \) in the colour space, with \( i \in \{1, \ldots, C\} \). We denote the probability, or weight, of the \( r \)th image Gaussian by \( \pi_{ir} \). By \( M^r \) and \( \omega^r \) we denote the \( n \)th learned colour model and its prior probability, respectively, for \( n \in \{1, \ldots, N\} \). We assume that each of the image clusters has a single colour, generated by a single colour model. The resulting model is shown in Fig. 3, where \( x \) represents a single pixel, and \( m(x) \) its model, or semantic colour.

We fit a mixture of Gaussians to an image’s pixels in the colour space using a method adapted from the split and merge EM method of [20]. This begins by fitting a single Gaussian to the data, splitting it and then running EM until convergence. We then alternate between splitting the most appropriate Gaussian, and running EM, until the stopping criterion is satisfied. We use the same stopping criterion as [20], but apply our own criterion to determine which Gaussian cluster to split.

Suppose we have \( L \) data \( \{x_i\} \) (each with weight \( w_i \)), for \( i \in \{1, \ldots, L\} \), which are modelled by a Gaussian with PDF \( f \). Then, the mean PDF of the data, under this Gaussian is

\[
\bar{f} = \frac{\sum_{i=1}^{L} w_i f(x_i)}{\sum_{i=1}^{L} w_i}
\]

The expected PDF of a Gaussian can be calculated using (4), to be

\[
\begin{align*}
E_f(f(x)) &= \int_{\mathbb{R}^K} f(x) \, dx = \tilde{g}(f', f) \\
&= \frac{1}{(2\pi)^{K/2} \sqrt{2|\Sigma|}}
\end{align*}
\]

Then, we split the Gaussian with the least value of

\[
\frac{E_f(f(x))}{\bar{f}}
\]

This is intended to give a measure of Gaussianity of the data represented by each Gaussian. The criterion splits the Gaussian whose underlying data is the least well represented. Our experiments showed that using our split criterion tended to result in a greater number of mixture components than the criterion of [20]. Although this may not be preferable for other tasks, for ’this’ task it allowed more accurate segmentation of images into semantic colours. In fact, any reasonable method of fitting a mixture of Gaussians to data could be used, provided it results in an appropriate number of components. Discussion or experimentation into different methods of fitting such mixtures is beyond the scope of this work.

As described above, an image to be segmented is represented as a mixture of Gaussians in feature space. We call this mixture \( C \), and it has PDF

\[
f_C(x) = \sum_{i=1}^{C} f_{C_i}(x) \pi_i^C
\]

We denote by \( M^r \), a colour model, itself a mixture of \( \tilde{M}^n \) Gaussians, \( M^r \), each with weight \( \pi_{ir}^M \), for \( j \in \{1, \ldots, M^r\} \). The probability of a model given a pixel value, \( x \), can then be expressed by marginalising the components of the image mixture model

\[
P(M^n|x) = \sum_{i=1}^{C} P(M^n|C_i, x)P(C_i|x)
\]

As implied by the model (see Fig. 3), \( M^n \) and \( x \) are conditionally independent given \( C \). Thus, we have

\[
P(M^n|C_i, x) = P(M^n|C_i) = P(M^n) f(C_i|M^n)
\]

where \( f(C_i|M^n) \) is the probability of the Gaussian mixture \( M^n \) under the Gaussian \( C_i \). This is simply the functional

\[
\tilde{g}(M^n, C_i) = \int_{\mathbb{R}^K} f_{M^n}(x) f_{C_i}(x) \, dx
\]

Hence

\[
P(M^n|x) \propto \omega^n \sum_{i=1}^{C} \sum_{j=1}^{M^n} \pi_{ij}^M \pi_{ir}^C f_M^n(x) f_{C_i}(x)
\]

We calculate \( \tilde{g}(M^n, C_i) \) using (4), and estimate \( P(M^n) = \omega^n \) using the EM algorithm, keeping the colour models and image mixture model fixed. Then, we classify each pixel, \( x \), as

\[
m(x) = \arg \max_{n \in \{1, \ldots, N\}} P(M^n|x)
\]

where \( m: \mathbb{R}^K \rightarrow \{1, \ldots, N\} \) is the classifier function.

4 Experiments and results

To obtain quantitative results, and to compare our method with that of [12], we performed tests using the same test set, which the authors of [12] have made available online. The set consists of four types of objects: cars, dresses, pottery and glass and shoes. For each object type there are 11 sets of images, one for each of the 11 colour terms used in [12]. Each of these image sets contains 10 test images, resulting in a total of 440 images (see Table 1). For the purpose of assessing the
quality of semantic segmentation, van de Weijer et al. [12] also produced a mask for each of these images, demarcating a region of the image believed to have that colour. These masks were produced by a combination of manual segmentation and some (unspecified) algorithm. However, many of them are not very precise, for example, Fig. 4 shows an example from the yellow shoe set, along with its mask and our segmentations of the mask contents, and of the whole image. The mask clearly contains a portion of the dark region below the shoe, which should ‘not’ be classified as yellow.

The images of dresses, pottery and glass, and shoes are mostly taken indoors. The majority of the car images, however, are taken outdoors. This combined with the reflective nature of cars’ surfaces makes the cars test set particularly difficult.

We begin by learning the 11 colour terms used in [12]: black, blue, brown, green, grey, orange, pink, purple, red, white and yellow. For a truly fair comparison, we learn these from the same training data used in [12], which is also the raw result of a Google Image search, and which the authors have made available online. We then took the 440 images of the dataset of [12], and applied our classification method to the portions of the images which lay inside the masks, to assign each of the pixels to one of the 11 learned colours. Comparing each resulting segmentation against its corresponding mask, we calculated the true positive rates achieved using our model. Our method outperformed that of [12] in three out of the four categories, and achieved better classification accuracy overall. We do not compare against the results of [10] here since, as we noted earlier, it requires a manually labelled validation set to fit parameters, and hence is much less weakly supervised than our work or [12]. These results are shown in Table 2. Note that in [10, 12] the overall results quoted are per-category, rather than normalised per-pixel means. We re-weight the per-category results quoted in [10, 12], such that all the percentages reported in Tables 2–4 are per-pixel percentages. For example, the value of 71.7% in Table 3 was originally quoted in [12] as 70.6%.

As described in Section 3, when classifying pixels of a novel image, we estimate the image specific prior distribution over the colour categories, \( o^* \). As in [12], we estimated this using only the pixels ‘inside’ the mask. As might be expected, this gives better quantitative results for classifying the mask content (which we expect to be of a single colour). For most applications, however, such masks would not exist, and using them during classification seems inappropriate and likely to lead to unrealistic results. Furthermore, when applied to a whole image (without a mask), an uninformative uniform prior typically gives more satisfactory ‘qualitative’ results. For example, Fig. 5 shows an image from the blue dresses category segmented using an image specific (centre), and uniform (right) prior. With the image specific prior, the colour blue is so over represented that part of the light background, and all of the dark belt are classified as blue. With the uniform prior, however, the result is much more pleasing, with both

<table>
<thead>
<tr>
<th>Categories</th>
<th>Colours</th>
<th>Images per cat. per col.</th>
<th>Total images</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>11</td>
<td>10</td>
<td>440</td>
</tr>
</tbody>
</table>

![Table 1](image1.png)

![Table 2](image2.png)

![Table 3](image3.png)

![Table 4](image4.png)

Fig. 4 An example from the test set of [12]

\( a \) Example image from the set of yellow shoe images

\( b \) Its mask from the test set of [12]

\( c \) Our colour segmentation of the content of the mask

\( d \) Of the whole image

\( \text{doi: 10.1049/iet-cvi.2012.0210} \)
background and belt correctly segmented. Fig. 6 shows more example segmentations using a uniform prior.

With this in mind, we repeated the segmentation of all 440 test images using a uniform prior over the colour models. Table 3 shows the classification results using both our method and those of [10, 12], with uniform priors. In all cases, these ‘quantitative’ results are worse than when using image specific priors, with our method showing the best overall performance.

As mentioned earlier, and demonstrated in Fig. 4, the masks used in [10, 12] are the result of quite coarse segmentation, and often misrepresentative of the colour they are intended to specify. For this reason, we took three images from each colour of each category of their dataset, and created our own masks, using hand-segmentation only. We then performed segmentation of these 132 images, ‘without’ taking the mask into account to create a prior, but instead using a uniform prior over colours. As Table 4 shows, our models give better results than the models learned using the more strongly supervised method of [10], which the authors have made available online.

The key features of our method are that it is extendible to higher dimensional feature spaces, and does not assume that colours are non-overlapping, or that they account for the entire colour space. Thus, our next experiments were performed to test the effectiveness of our method at learning a small number of overlapping colours.

We compiled the following two colour term sets:

Set 1: cyan, dark green, emerald, lime green and turquoise.
Set 2: indigo, lavender, magenta and salmon pink.

Each colour was learned using 80–100 images from Google Image searches on the colour term only. For the test data, we obtained 10 and 12 images for Set 1 and Set 2, respectively, each image corresponding to one of the colour terms. For performance quantification, we hand-segmented the images to create masks of which portions of the image we consider to have the colour. For pixel classification, we achieved 100% accuracy for Set 1, and 98.1% for Set 2.

Finally, for a more qualitative demonstration of our approach, we tried using the learned background distribution as an additional colour category, to segment images into known colours (colours for which we have learned a distribution) and unknown colours (colours for which we have not learned a distribution). We took the five learned models of Set 1, and the background model learned at the same time, and segmented images (without masks) into these six categories – see examples in Fig. 7. The top row shows a pair of patterned curtains segmented into known and unknown colours – the greens and blues of the leaves have been labelled as dark green, lime green and turquoise, whereas the grey, white and red parts of the image were labelled background (shown absolute white). Similarly with the third image, both the orange stripe and light background have been labelled as background as neither colour is a member of Set 1, as shown. We segmented the image with both an image specific prior (centre) and a uniform prior (right). Using a uniform prior, the known colours (turquoise and green) were correctly segmented, whereas the unknown colours (red and white) were labelled as background. With the image specific prior, the image was simply segmented into turquoise and background only, perhaps because the turquoise model was given too great a weight by the prior.

One particular shortcoming of our model is its ability to learn and classify achromatic colours; black, white and grey. These colours consistently scored lower than others, perhaps because the difference between white and grey, and between grey and black, depends so much upon the light level in the image as a whole. For example, see the performance of our white model in Table 5. Our model

---

**Fig. 5** Example of segmenting a novel image (left) using an image specific (centre), and uniform (right) prior

**Fig. 6** Some example segmentations using our method, with uniform priors on the colour models
could be expanded to incorporate regional or global features to help deal with this issue. Fig. 8 shows some examples of failure cases using our method. One particularly noteworthy example is the image of the black car on the first row of Fig. 8. The whole of the car, the grey tarmac on which it sits, and the scenery behind it has been classified as grey. Only the grass and sky have been correctly segmented.

Fig. 7  Examples of segmentation using the background class as a colour category in addition to the five colours of Set 1
The background class is shown absolute white in the segmentations
Note that the orange region of the third image has been classified as background, since orange is not a member of Set 1

Fig. 8  Examples of failure cases using our method
Segmentations are into the 11 basic colours, with uniform priors on the colour models

Table 5  A confusion matrix showing percentage classification rates with 11 colours, using the proposed method. Corresponds to the bottom row of Table 4.

<table>
<thead>
<tr>
<th>Classified Colour</th>
<th>Black</th>
<th>Blue</th>
<th>Brown</th>
<th>Grey</th>
<th>Green</th>
<th>Orange</th>
<th>Pink</th>
<th>Purple</th>
<th>Red</th>
<th>White</th>
<th>Yellow</th>
<th>Actual Colour</th>
</tr>
</thead>
<tbody>
<tr>
<td>87.0</td>
<td>2.7</td>
<td>0.9</td>
<td>9.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Black</td>
</tr>
<tr>
<td>2.7</td>
<td>93.9</td>
<td>1.7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Blue</td>
</tr>
<tr>
<td>0.3</td>
<td>84.6</td>
<td></td>
<td>11.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Brown</td>
</tr>
<tr>
<td>18.8</td>
<td>0.1</td>
<td>79.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Grey</td>
</tr>
<tr>
<td>0.5</td>
<td>1.3</td>
<td></td>
<td>95.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Green</td>
</tr>
<tr>
<td></td>
<td>2.2</td>
<td>91.1</td>
<td>2.7</td>
<td>1.2</td>
<td>2.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Orange</td>
</tr>
<tr>
<td></td>
<td>0.1</td>
<td>10</td>
<td></td>
<td>3.3</td>
<td>14.6</td>
<td>1.9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Pink</td>
</tr>
<tr>
<td>11.4</td>
<td>3.1</td>
<td>1.7</td>
<td>3.8</td>
<td>4.8</td>
<td>1.7</td>
<td>72.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Purple</td>
</tr>
<tr>
<td>0.8</td>
<td>1.0</td>
<td>0.4</td>
<td>2.2</td>
<td>1.9</td>
<td>93.7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Red</td>
</tr>
<tr>
<td>33.2</td>
<td>1.5</td>
<td>11.0</td>
<td>3.1</td>
<td></td>
<td>51.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>White</td>
</tr>
<tr>
<td>0.9</td>
<td>0.5</td>
<td></td>
<td>0.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>98.4</td>
<td></td>
<td>Yellow</td>
</tr>
</tbody>
</table>

5  Conclusions and future work
We have proposed a model for learning colours from noisy, weakly supervised training data, which makes few assumptions, and can learn partially overlapping colours. The model improves upon the previous work in terms of quantitative results. We have also proposed a means of applying the learned colour models to novel images, in
order to segment them into areas corresponding to semantic colour terms.

Our use of a continuous feature space has allowed us to exploit some of the convenient mathematical properties of Gaussians to model image colour content for the task of semantic colour naming in a better manner. Such probabilistic methods are necessary for realising continuous versions of usually discrete models, such as PLSA. This would make it possible to extend such models to higher dimensional feature spaces.

In our future work, we intend to progress to experiments on more complex attributes incorporating such features as texture and shape.

6 Acknowledgments

The authors would like to thank the authors of [10, 12] for making their dataset and masks available online. The authors would like to thank the UK Engineering and Physical Sciences Research Council.

7 References

8 Russakovsky, O., Fei-Fei, L.: ‘Attribute learning in large-scale datasets’. ECCV – Int. Workshop on Parts and Attributes, 2010