Describing Visual Textures Using Natural Language

Mikayla Timm
mtimm@cs.umass.edu

Abstract

Visual textures can be characterized by their color, shape, periodicity, and other attributes that can often be described using natural language. In this paper we study the problem of describing detailed properties of visual textures on a novel dataset containing rich descriptions of textures. We evaluate different deep learning approaches for mapping visual features to natural language on this dataset and find that these models are limited in their ability to handle compositionality. Our dataset also allows us to generate natural language explanations of what discriminative features are learned by deep networks for fine-grained categorization, where texture plays a key role in discrimination.

1 Introduction

Texture is ubiquitous and provides useful cues for a wide range of visual recognition tasks. We rely on texture for estimating material properties of surfaces, for fine-grained discrimination between objects of similar shape, and even for generating realistic imagery in computer graphics applications. This reliance is also mimicked by deep networks trained on current computer vision datasets which often rely on texture properties for discrimination. Texture is localized and more easily modeled than, for example, shape that is affected by pose, viewpoint, or occlusion. Despite its importance, there has been limited work on describing detailed properties of texture using natural language.

We introduce a new dataset containing rich natural language descriptions of textures taken from the Describable Textures Dataset (9). Our dataset, called the *Describable Textures Dataset in Detail*, or DTD^2, contains several descriptions of each image obtained by crowdsourcing and vastly extends the 47 attributes present in the original dataset. As

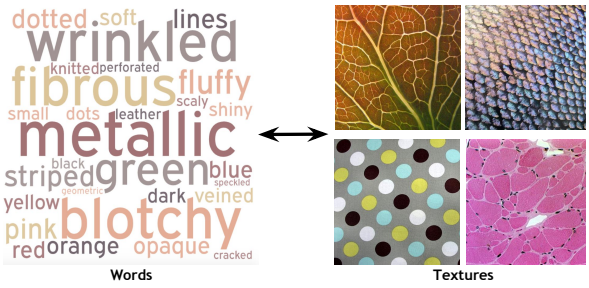

Figure 1: We introduce the Describable Textures Dataset in Detail (DTD^2), consisting of texture images with rich natural language descriptions. Attributes included in these descriptions reflect multiple aspects of visual texture, including color, material, pattern, tactile texture, and style.

seen in Figure 1, these descriptions contain a wide range of words that describe the color of the elements within the image, their shape, size, spacing, and other high-level perceptual properties.

The domain of textures is rich and poses a number of challenges for compositional language modeling. For example, to estimate the color of dots of dotted texture, the model must learn to associate the color to the foreground dots and not the background. We investigate off-the-shelf discriminative and generative models for mapping image features to attribute phrases and evaluate their performance on this dataset. We also experiment with two bilinear CNN- and NLP-based encoder models that aim to solve some of the difficulties in compositional language modeling. We find that the baseline classification model performs the best on our retrieval tasks, indicating that our recall metrics do not benefit from incorporating natural language processing techniques into training the models. The current approaches we investigated are able to retrieve relevant phrases for most images, even some relevant phrases that are not in the ground truth. Retrieving images that corre-

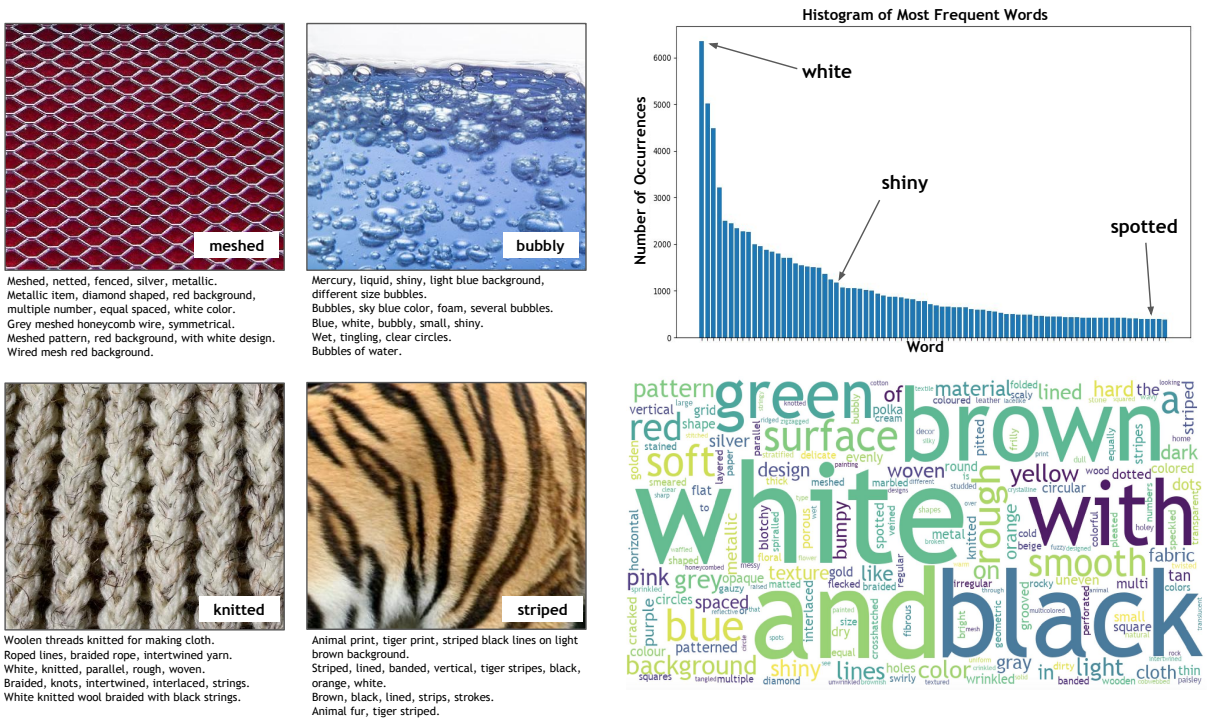

Figure 2: Four **example texture images** and their natural language descriptions from our new *Describable Textures Dataset in Detail* (left). A **histogram of the most frequently used words** in our natural language vocabulary is also shown (top right), along with a **word cloud** of the most frequently used words in the dataset (bottom right).

spond to query phrases appeared to be a harder task, as all of our experimental models struggled to retrieve meaningful images on many phrases, particularly compositional phrases such as "green background."

We also present an application of our dataset, where we visualize what discriminative texture properties are learned by existing deep networks for fine-grained classification on several natural domains such as birds, flowers, butterflies, and fungi. To do this, we generate maximal images for each category within the dataset using a state-of-the-art texture-based classifier and describe these images using models trained on DTD[2]. We find that the resulting explanations are well-aligned with the discriminative attributes of each category (e.g., the bird "Spotted Catbird" is green and dotted, as seen in Figure 7). Such descriptions of texture attributes are currently lacking in datasets for explainable AI.

2 What you proposed vs. what you accomplished

- ~~Analyze, visualize, pre-process text descriptions and attribute phrases.~~
- ~~Build baseline model for attribute phrase classification.~~
- ~~Iterate on LSTM-based captioning model.~~
- ~~Build 2 compositional classification models to learn embeddings for images and phrases.~~
- ~~Analyze applications to fine-grained datasets.~~
- *Use bilinear CNN features in all 3 models to compare to ResNet-101 image features.* Did not complete in time - elected to spend the final week building a second compositional model to compare BERT phrase embeddings to RNN phrase embeddings.
- ~~Evaluate all models on image to phrase and phrase to image retrieval.~~

3 Related Work

Visual texture is a fundamental component of human and machine perception and has been the focus for representing image features in the computer vision community for decades (15; 8). Texture representations have been used in computer vision for characterizing materials (18; 33; 6), as understanding materials in images assists with classic tasks such as object recognition and image

retrieval (25). The SIFT descriptor from (23) generates image features that are invariant to changes in scale, rotation, and translation, and have been widely used for vision tasks over the years. First and second order pooling statistics of SIFT descriptors as in VLAD (14) and Fisher vectors (28) have been used for representing textures, along with (more recently) learned image features from deep convolutional neural networks (32; 12).

Texture is a crucial factor in the success of fine-grained recognition models (Lin et al.; 19), especially where a deep, convolutional neural network is previously trained for image classification on a massive dataset (31). There is remarkable evidence that these fine-grained models rely a lot on texture, and not necessarily shape or ordering of objects. However, in previous work, we do not have the right vocabulary to describe these textures. The current explainable AI models for birds output phrases such as, "This is a large black bird with a pointy black beak," but the fine-grained model for classifying this bird would actually be focusing on textures such as small stripes on the bird's back that we are not really describing (30).

Describing images using natural language from the perspective of computer vision and language generation has been primarily examined in the context of image captioning (16; 34). For example, Show, Attend and Tell (36) is a widely used neural image captioning model that utilizes a convolutional neural network encoder and an visual attention-based LSTM network (13) decoder to generate descriptions of scene and object images. Existing vision and language datasets focus mostly on scenes and objects, such as MS COCO (20) and Flickr30k (29). However, there is this domain of texture images in particular that has been somewhat neglected in the image captioning literature.

The Describable Textures Dataset (9) is a rich dataset of 5640 texture images collected from Google and Flickr, equally distributed across a range of 47 different canonical texture categories, such as "dotted", "fibrous", "interlaced", and "waffled". These 47 canonical categories were selected based off previous work in psychology (7) that studied how humans recognize textures. Though informative for understanding the general category of a texture image, the Describable Textures Dataset does not include any information about the texture's color or any other distinguishing characteristics, only the canonical category. Specific details on each individual texture image are lacking, as all images are grouped into this limited span of canonical categories. This makes it impossible to distinguish unique texture characteristics for the images within each canonical category.

In our work, we aim to address the lack of rich, detailed information on texture in the literature by collecting an enhanced natural language dataset of texture images.

4 Dataset and Tasks

We introduce a new dataset expanding on the original Describable Textures Dataset that provides natural language descriptions to help address the lack of rich, distinguishing attributes in previous texture datasets. We call this dataset *Describable Textures Dataset in Detail*, or DTD^2.

4.1 Description collection

We collected our descriptions by presenting each DTD image and its corresponding DTD texture category to 5 Amazon Mechanical Turk workers, asking them to describe the texture spanning most of the image with as much detail as possible using natural language. We prompted them to provide at least 5 descriptive words or phrases in their description of each texture. Describable aspects of each image included texture, color, shape, pattern, style, and material. We allowed the annotators to include the original DTD attribute in their description if they felt it was appropriate, but to specifically provide additional details of these attributes in their descriptions.

Using this strategy, we collected 5 descriptions (with no less than 5 words each) from 5 different workers for each texture image, providing detailed information for each texture. After collecting the raw annotations, we manually verified that all descriptions for each image were appropriate and relevant, and we removed descriptions that were deemed unfit (not a description of the texture).

In this verification step, we found that a select number of the DTD images were difficult to describe in more detail without talking about the object present in the image, for example, a "waffled" image containing a breakfast waffle may have had descriptions that talked about the waffle food item instead of the texture of the waffle itself. We elected to remove these unfit descriptions from our

dataset. We excluded images from the dataset with fewer than 3 valid descriptions after removing the unfit descriptions. An additional observation we noticed during this verification process is that the "freckled" category from DTD contained many images of human faces, and the "potholed" category contained many images of scenes of roads not necessarily zoomed in on a single texture. As a result, many of the descriptions we collected for these two categories described features of the faces and scenes in the images, not necessarily a description of a single texture. We elected to remove the "freckled" and "potholed" categories from our dataset completely to ensure all images and descriptions focused on textures only.

4.2 Attribute phrases

From the collected annotations (as demonstrated in Figure 2), we noticed that people describe various aspects of texture often by a set of attribute phrases rather than a complete, grammatically correct sentence. This is in contrast to standard image captioning datasets, where images of scenes are often described using locations of objects. As texture itself is relatively unordered when spanning an image, it is interesting to note that permuting the order of the attribute phrases often does not affect the description semantics, and can often be read as a list of unordered phrases. Therefore, we can further split the collected texture descriptions into a list of attribute phrases by splitting each description on commas (','). This give us a set of attribute phrases for each texture image, which we can concatenate to form a full description.

We conducted an experiment to verify that the list of attribute phrases could actually be represented in an unordered fashion without losing meaning when generating texture descriptions. We randomly sampled 100 texture images from the dataset, split each image's description into a list of attribute phrases, then permuted the order of the phrases by shuffling them and re-concatenating with comma separators. For each of the randomly sampled images, we had three people try to guess whether the *original* list of attribute phrases or the *permuted* list of attribute phrases was the original annotation from the dataset - for example, for the "meshed" image shown in Figure 2, we want to identify whether *"Meshed, netted, fenced, silver, metallic"* (original list) or *"Silver, netted, fenced, metallic, meshed"* (permuted list) is the original annotation. Our experiment showed that people were only able to identify the correct original annotation around 50% of the time, indicating that it is essentially random chance. Hence, this confirmed our hypothesis that our texture descriptions can be represented as an unordered list of phrases without losing significant semantic meaning.

# images	5369
# unique words	827
# unique phrases	940
# descriptions per image	4.6
# attribute phrases per image	18.5
# images train	3212
# images val	791
# images test	1366

Table 1: Important statistics for the *Describable Textures Dataset in Detail* (DTD^2).

4.3 Dataset statistics

The final *Describable Textures Dataset in Detail* contains 5369 texture images, with an average of 4.6 descriptions (18.5 attribute phrases) per image. The whole dataset contains 22854 unique attribute phrases and 7681 unique word types. For our experiments, we used attribute phrases that occur in at least 10 different images across DTD^2. After this filtering, we have 940 unique phrases and 827 unique words. A summary of our experimental dataset statistics is shown in Table 1.

4.4 Tasks

For our tasks, we have created a train, validation, and test benchmark for this dataset. Each split contains the same proportion of DTD attributes for an even distribution of texture images, randomly sampled for each attribute (stratified sampling). 60% of the images for each DTD attribute are assigned to the training set, 15% to validation, and 25% to test. Table 1 shows the distribution of images across our train, validation, and test sets. We use these splits to train and evaluate our models on various tasks.

On top of our new dataset, we propose a set of retrieval and generation tasks as follows:

Image to phrase retrieval: Given an image, we retrieve the attribute phrases from DTD^2 that best describe the texture in the image. We can evaluate how well different models perform on our test images by calculating the *top-k recall*: the percentage

of ground-truth attribute phrases that are included in the k retrieved phrases. As many of the 22854 unique attribute phrases are sparse and do not appear for more than one image across the dataset, we decide to only retrieve attribute phrases that occur in at least 10 different images across DTD^2, resulting in a set of 940 frequently-used attribute phrases to retrieve from.

Phrase to image retrieval: One can retrieve an image from a candidate image set according to a given attribute phrase. That is, given an attribute phrase, retrieve the texture images from DTD^2 that correspond most to the query phrase. We evaluate this image retrieval task using *top-k recall*: the percentage of ground-truth images that are included in the k retrieved images.

Description generation: This task aims to generate a whole description on a given image in a generative fashion, as opposed to retrieving from a list of attribute phrases. We evaluate the description generation task on the BLEU (27) captioning metric.

These tasks are open-ended by nature: there are many more descriptions and attribute phrases than the labeled ground-truth that are true for any given image, and many more images that match with the given descriptions than are labeled. Therefore, qualitative visualizations are important to fully understand and evaluate the performance.

5 Methods

We investigate four different methodologies for describing visual textures in DTD^2. All models were trained using PyTorch, running on the Gypsum GPU cluster.

5.1 Discriminative Classification Model

The first baseline model we examine for the retrieval tasks is a classification-based approach. Here, we frame the description task as a classification problem, where we train a discriminative model to output class scores across the 940 frequently-used attribute phrases for each DTD^2 image. Given a texture image, we compute a set of image features using an ImageNet (10) pre-trained ResNet101 model (12). We then pass these image features through a fully-connected hidden layer of dimension 1024, then a final fully-connected output layer of size 940, giving us an output vector $x \in \mathcal{R}^{940}$ representing the 940 attribute phrases we are predicting scores for. The loss function used for this classifier was a weighted binary cross entropy with logits loss, where the weight (indicated by w_i) corresponds to the phrase frequency for each position i in the output vector x. The weighted binary cross entropy loss is given by $l_i = -w_i[t_i * \log \sigma(x_i) + (1 - t_i) * \log(1 - \sigma(x_i))]$, where t_i is the target value for that phrase/image pair in the ground truth (1 if the phrase is in the ground truth for that image, 0 if not), and σ is the sigmoid function.

We train this model on our training split for 200 epochs using the Adam optimizer (17), with a base learning rate of 0.001 and weight decay set at 0.0001. Corresponding code files: models/img_encoder.py, models/phrase_classify.py, run/train_phrase_classify.py.

Classification models are easy to train and lend themselves naturally to retrieval tasks, which are the main experiments of interest in this paper. However, this discriminative model will be unable to generalize to new, unseen attribute phrases, as it is limited to the phrase classes it was trained on.

5.2 Generative Language Model (Show, Attend and Tell)

The second baseline model we have experimented with for the texture description task is a generative language model that outputs natural language descriptions of texture images. The training objective of this model is to learn embeddings of words in our descriptions and output a texture "caption" that captures the information in our ground truth descriptions. Our base architecture for this approach is the Show, Attend and Tell (SAT) model from (34). This is a widely used neural image captioning model that utilizes a convolutional neural network encoder and an attention-based LSTM (13) decoder to generate descriptions of images. The benefit of using a generative language model over a classification model is that it will allow us to generalize to arbitrary unseen attribute phrases, while the classification model can only handle a limited set of 940 possible phrases.

We trained this language model using the images and the *full descriptions* from DTD^2. We chose to freeze the weights of the image encoder, which is an ImageNet (10) pre-trained ResNet-101 (12) model, so it uses the same image features as the discriminative classification approach. That is, we only update the

parameters of the LSTM decoder at training time, leaving the image encoder parameters frozen. We used an initial learning rate of 0.0004 with the Adam optimizer in PyTorch, and trained for 11 epochs with a batch size of 64 for the best validation set performance. Corresponding code files: models/language_generation.py, run_new/train_language_generation.py, run_new/eval_language_generation.py (these files were adapted from https://github.com/sgrvinod/a-PyTorch-Tutorial-to-Image-Captioning), run_new/scores_language_generation.py, run_new/evaluate_scores.py.

5.3 Bilinear RNN Classifier

The third model we experimented with on the texture description task is a bilinear model that aims to improve the original classification approach by learning relationships between related attribute phrases. For example, if the model has a good understanding of what it means for an image to have "stripes", and it has also seen compositional attribute phrases such as "red stripes" or "green stripes" before, we would like for it to use some of its knowledge of these compositional attributes to extend to "blue stripes" or "yellow stripes". The idea is that, since many of these attributes have a compositional structure describing different aspects of the texture (pattern, color, etc.), we should not necessarily train a separate classifier for each phrase while ignoring all other phrases.

This compositional model aims to learn a weighted "attention" over the phrases by mapping them to an embedding space (in this case, a recurrent neural network) and computing the matrix product between the phrase encodings and image encoding. We then use a classification loss to train the model to give phrase probabilities for each image. Through experimentation of different architectures, the following setup gave best results on the validation set:

Image Encoding Stream

- Pass image x through ImageNet pre-trained ResNet-101 model to obtain 2048-dimensional image features, then perform batch normalization and ReLU on the features to get $x_{resnet} \in \mathcal{R}^{2048}$.
- Pass x_{resnet} through fully-connected hidden layer with weight matrix $W_{hid} \in \mathcal{R}^{2048\times1024}$, then perform batch normalization and ReLU to get $x_{hid} \in \mathcal{R}^{1024}$.
- Pass x_{hid} through fully-connected output layer with weight matrix $W_{out} \in \mathcal{R}^{1024\times512}$ to get $x_{out} \in \mathcal{R}^{512}$.

Attribute Phrase Encoding Stream

- Randomly initialize an embedding for each phrase $y \in \mathcal{R}^{512}$.
- Pass each attribute phrase embedding y through a 1-layer recurrent neural network. Extract the hidden state of the RNN to obtain $y_{emb} \in \mathcal{R}^{512}$, the encoding for each phrase.
- Arrange the 940 attribute phrase encodings into a phrase embedding matrix $Y_{emb} \in \mathcal{R}^{512\times940}$. Each column corresponds to the RNN embedding for one phrase.

After encoding the image and attribute phrases, we compute their **matrix product**, giving a 940-dimensional output vector. Concretely, we compute $\phi = x_{out}Y_{emb} \in \mathcal{R}^{940}$. A pictorial representation of this bilinear RNN model is shown in Figure 3.

This model was trained with a weighted binary cross entropy with logits loss, where the weight (indicated by w_i) corresponds to the phrase frequency for each output position i in ϕ. The weighted binary cross entropy loss is given by $l_i = -w_i[t_i * \log\sigma(\phi_i) + (1-t_i) * \log(1-\sigma(\phi_i))]$, where t_i is the target value for that phrase/image pair in the ground truth (0 or 1), and σ is the sigmoid function.

We trained the model for 700 epochs with an image batch size of 64 using the Adam optimizer in PyTorch. Hyperparameters were tuned using the validation set to measure image to phrase and attribute to phrase retrieval, with an initial learning rate of 0.0001 and 0.0001 weight decay. The learning rate was decayed every 200 epochs with gamma (multiplicative factor) set to 0.1. Corresponding code files: models/bilinear_compositional_models.py, run_new/train_bilinear_classifier_rnn.py.

5.4 Bilinear BERT Classifier

The fourth and final model we experimented with is similar to the Bilinear RNN Classifier, but instead of using a recurrent neural network to encode the phrases, we use the final output layer embeddings from BERT (11). The goal of

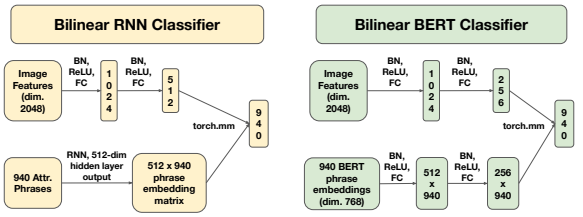

Figure 3: **Bilinear Compositional Models**. Both models embed texture images and attribute phrases in a two-stream approach before combining their embeddings through their matrix product. The final output becomes a 940-dimensional vector, which is used to assign probabilities of the 940 phrases to the given image by training with a classification cross entropy loss. The Bilinear RNN classifier embeds the 940 raw natural language phrases using a recurrent neural network, and the Bilinear BERT classifier obtains BERT embeddings for the 940 phrases and passing them through two fully connected layers.

using BERT is that the embeddings learned in BERT would provide a stronger starting point for representing the natural language semantics of the phrases, hopefully providing a better result than training an RNN from scratch. We pass these BERT embeddings through two fully connected layers before computing the matrix product between the phrase embedding matrix and the image embeddings. Concretely, we use the following setup for this model:

Image Encoding Stream

- Pass image x through ImageNet pretrained ResNet-101 model to obtain 2048-dimensional image features, then perform batch normalization and ReLU on the features to get $x_{resnet} \in \mathcal{R}^{2048}$.
- Pass x_{resnet} through fully-connected hidden layer with weight matrix $W_{hid} \in \mathcal{R}^{2048\times1024}$, then perform batch normalization and ReLU to get $x_{hid} \in \mathcal{R}^{1024}$.
- Pass x_{hid} through fully-connected output layer with weight matrix $W_{out} \in \mathcal{R}^{1024\times256}$ to get $x_{out} \in \mathcal{R}^{256}$.

Attribute Phrase Encoding Stream

- Compute BERT embeddings by taking the output of the last layer of BERT for each attribute phrase y, then perform batch normalization and ReLU on each embedding to get $y_{BERT} \in \mathcal{R}^{768}$ for each of the 940 attribute phrases.
- Pass each phrase's BERT embedding y_{BERT} separately through a fully-connected hidden layer with weight matrix $W_{hid} \in \mathcal{R}^{768\times512}$, then perform batch normalization and ReLU to obtain $y_{hid} \in \mathcal{R}^{512}$, the intermediate embedding, for each attribute phrase.
- Pass each phrase's y_{hid} through a fully connected output layer with weight matrix $W_{out} \in \mathcal{R}^{256}$, giving a $y_{out} \in \mathcal{R}^{256}$ embedding for each attribute phrase.
- Arrange the 940 y_{out} phrase embeddings into a phrase embedding matrix $Y_{emb} \in \mathcal{R}^{256\times940}$. Each column in this matrix corresponds to the encoding for one attribute phrase.

After encoding the image and attribute phrases, we compute their **matrix product**, giving a 940-dimensional output vector. Concretely, we compute $\phi = x_{out}Y_{emb} \in \mathcal{R}^{940}$, the same size output as the previous bilinear RNN model. A pictorial representation of the bilinear BERT model is shown in Figure 3.

This model was also trained with a weighted binary cross entropy with logits loss, the same objective as the bilinear RNN classifier. We trained this model for 530 epochs with an image batch size of 64 using the Adam optimizer in PyTorch. Hyperparameters were tuned using the validation set to measure image to phrase and attribute to phrase retrieval, with an initial learning rate of 0.0001 and 0.0001 weight decay. The learning rate was decayed every 200 epochs with gamma (multiplicative factor) set to 0.1. Corresponding code files: models/bilinear_compositional_models.py, run_new/train_bilinear_classifier_bert.py.

6 Experiments

All experiments in this section are carried out on the held-out DTD[2] test set, described in Section 3.4.

6.1 Image to Phrase Retrieval

For our experiments, we first focus on generating descriptions that are more detailed than existing attributes as categories. We accomplish this by retrieving attribute phrases for each image that are scored highly by the models and observing the top-k recall on the test set. Concretely, we measure recall as:

$$recall = \frac{truepositives}{truepositives + falsenegatives}$$

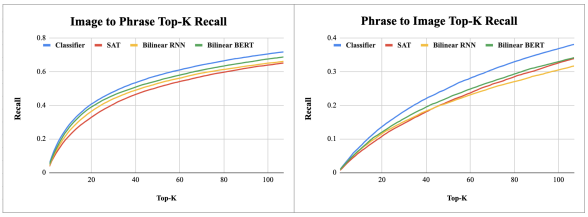

Figure 4: **Top-k Recall Curves** for Phrase Retrieval and Image Retrieval tasks. We plot recall scores for $k = 1$ to 110. The basic classifier model achieves higher recall than the Show, Attend, and Tell model (SAT) as well as both bilinear NLP-based models.

The phrase scores for the baseline classification model and bilinear compositional models are obtained for each image by taking the output vector from the last layer, where each element of the vector corresponds to a score for each attribute phrase.

To get phrase scores from the Show, Attend and Tell model, we look at the probability for each phrase given its image encoding. The LSTM encodes a probability distribution over all the words in our vocabulary at each time step given an input image x and previous words $P(y_t|x, y_1, y_2, ..., y_{(t-1)})$. Hence, we can compute the probability of any phrase y given an image x as $P(y|x) = P(y_1|x) * P(y_2|x, y_1) * P(y_3|x, y_1, y_2) * ... * P(y_t|x, y_1, ..., y_{(t-1)})$.

The top-k recall results for this image to phrase retrieval task on DTD^2 are shown in Figure 4. Some representative example outputs are shown in Figure 5. For the qualitative part of this study, we examined 102 different query images, retrieving the top 20 attribute phrases (10 visualized in the figure) on each for all four models. From this study, we noticed that color is an example of something that is hard to do with our current models. For example, "black and white" is not the same as "black" or "white". Classification of attributes helps provide additional information on the texture, but does not capture colors of specific things like lines, checkers, or dots. Perhaps more data is needed to improve performance. However, despite the flaws of the experimental models, they perform reasonably well in retrieving relevant phrases and images, as seen in our qualitative visualizations.

6.2 Phrase to Image Retrieval

Another task we explore with our new dataset is phrase to image retrieval. In this task, we retrieve the images that score the highest when querying on a specific attribute phrase of interest. For each model, we retrieve the top-k images given a phrase by ranking all the images by their output score for that phrase. The top-k recall curves for each of the models evaluated on phrase to image retrieval are shown in Figure 4. Representative example outputs for all the models on phrase to image retrieval are shown in Figure 6. In our qualitative study on phrase to image retrieval, we examined 102 query phrases and retrieved the top 20 images (4 visualized in the figure) corresponding to each phrase for all four models. We noticed that all of the models often had a hard time distinguishing the background from the foreground, such as in the "green background" example shown in Figure 6, where we are lucky to obtain some shades of green in our retrieved images, but not all images have a green **background**. It is also noteworthy to mention that some phrases and visual features may actually be easier to model with our phrase and image representations than others. For example, a phrase like "abstract" may be harder for the models to learn a representation for due to its open-ended and somewhat subjective nature, while "geometric" (seen in Figure 6) is more concrete, leading all of the models to perform better on these easier phrases.

It may be reasonable to assume that having insufficient labels for these phrases would contribute to this bad performance, as not every person who labels each image may put the same set of phrases. There are an infinite number of possible phrases any person could write for a given image, as we did not force them to select from a predefined set of phrases in our labeling form during data collection.

Our cutoff for only evaluating phrases that occur in at least 10 different images across DTD^2 helps ensure that the models have some chance to learn relationships between the image features and the phrases. We acknowledge that certain phrases may be easier to learn qualities about than others in our dataset, especially given the long-tail distribution of phrase frequency as shown in Figure 2. For example, the phrase "white" occurs significantly more across the images in the dataset than the phrase "spotted".

To examine the effect this long-tail distribution has on our models, we conducted a study where we separate the set of evaluation phrases according to their frequency in the dataset. We split the histogram of phrases into 5 bins, then measure phrase to image retrieval recall performance for all mod-

Query Texture Image	Ground Truth Phrases	Classifier Phrase Ranking	SAT Phrase Ranking	SAT Output Caption	Bilinear RNN Phrase Ranking	Bilinear BERT Phrase Ranking
	blue, red, braided, yellow, multi color, twisted, knotted, intertwined, interwoven, white background, equal size, woven	braided, twisted, regular, soft, colorful, red, woven, unwrinkled, white, blue	wooly, soft, rope, yarn, twisted, knitted, painting, woven, fabric, multicolored	blue and yellow colored knitted material	braided, woven, soft, twisted, lined, multi color, red, regular white, patterned	twisted, soft, braided, blue, white, red, yellow, strands, black, rough
	bumpy, yellow, rough, uneven, green, fruit, vegetable	bumpy, green, white, smooth, uneven, rough, metallic, shiny, gray, yellow	bumpy, food item, frilly, uneven, green, bubbles, bubbly, yellow, surface, shiny	bumpy uneven rough opaque dull yellow surface	bumpy, rough, green, yellow, lumpy, uneven, metallic, blue, wrinkled, rugged	bumpy, green, rough, yellow, bubbly, hard, shiny, organic, white, smooth
	polka-dotted, pink background, evenly spaced	dotted, dots, soft, polka-dotted, white, circles, spotted, smooth, unwrinkled, pink	polka dots, polka-dotted, cloth, white dots, dots, circles, pink background, dotted, fabric, spotted	polka dotted spotted pink white circles	fabric, evenly spaced, dotted, spotted, dots, clean, unwrinkled, soft, pattern, white	unwrinkled, polka-dotted, textile, dotted, circles, pink background, soft, spotted, fabric, pink

Figure 5: **Image to Phrase Retrieval Results for Sample Query Images.** Given an image, we retrieve the top 20 phrases with each model. The phrases are concatenated with commas and listed in descending order corresponding to their score for each model, with the highest scoring phrase appearing first in each list. Retrieved phrases that are also in the ground truth phrases for each image are highlighted in blue for comparison. Additionally, the generated output caption from the Show, Attend and Tell (SAT) model is shown for each image.

els across each of the 5 bins. Fixing the number of images we retrieve to 20, we show the recall@20 scores for each model across the 5 bins in Table 2. These results show that the baseline classification model performs the best on all phrase frequency buckets, indicating that the NLP-based models do not offer much improvement even on the more rare phrase buckets. Additionally, it is interesting to note that recall scores did not necessarily decrease when measured on bins with lower average phrase frequency compared to bins with higher average phrase frequency. This may be explained by the fact that some phrases are inherently easier to understand and are more well-defined than others ("geometric" versus "abstract").

6.3 Description Generation

We can also generate new descriptions for texture images using the Show, Attend and Tell model. We do this by passing in an unseen image from the test set into the image encoder, and passing the image embedding through the LSTM decoder to obtain a sequence of words using beam search. Using this setup, we report a BLEU score of 0.04645. This is relatively low, and could be due to insufficient training data as the word embeddings passed into the LSTM were trained from scratch. A potential improvement to this experiment could involve initializing the word embeddings of the LSTM with BERT. Example output captions from the Show, Attend and Tell model (SAT) are shown in Figure 5.

7 Application

7.1 Visualizing Textures of Fine-grained Categories

We analyze how categories from recent FGVC challenges (4; 5) can be described by their *textural content*. The motivation is that subtle differences between species of birds or butterflies can often be described in terms of the texture associated with them and that several top-performing networks are inspired by texture-based representations. These representations are characterized by orderless pooling of second-order filter activations such as in bilinear CNNs (Lin et al.) and the winner of the iNaturalist 2018 challenge (19).

Concretely, for each category we (i) visualize the "maximal images" by obtaining inputs $\mathbf{x}$ that maximize the probability of the particular class according to a texture-based deep network $C_\theta(\mathbf{x})$, and (ii) automatically describe the maximal images using a set of texture attributes. We use C_θ as a multi-layer bilinear CNN as described in previous work on visualizing deep texture representations (21).

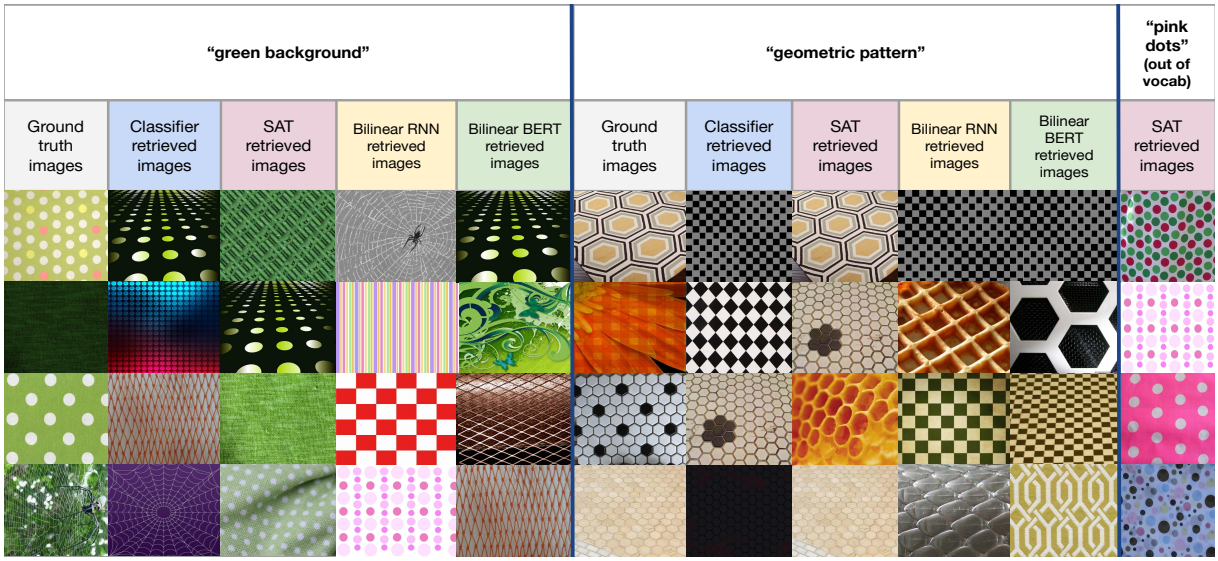

Figure 6: **Phrase to Image Retrieval Results for Sample Query Phrases.** Given a phrase, we retrieve the top 4 images with each model. Ground truth images for each of the phrases are shown for reference. An out-of-vocabulary phrase ("pink dots") can be used as a query for the Show, Attend and Tell (SAT) model, as it is able to generalize to new phrases unlike the classification-based approaches.

	Avg Phrase Freq	Classifier r@20	SAT r@20	Bilinear RNN r@20	Bilinear BERT r@20
Bin 1	300.7	**0.1281**	0.1011	0.112	0.1178
Bin 2	49.5	**0.1476**	0.13	0.1296	0.1286
Bin 3	25.1	**0.1774**	0.1217	0.1311	0.1328
Bin 4	16.0	**0.1611**	0.1181	0.1275	0.1262
Bin 5	11.3	**0.1385**	0.1119	0.0888	0.1012

Table 2: **Results of experiment examining the effect of phrase frequency on image retrieval performance.** We evaluate the models' recall performance on different bins of the test set, where each bin corresponds to sets of phrases with varying frequencies. Bin 1 contains the highest frequency phrases, and Bin 5 contains the lowest frequency phrases. The baseline classifier performs the best out of all the models across all frequency bins.

Visualizing categories as maximal textures. We visualize the categories from Caltech-UCSD birds (35), Oxford flowers (26), FGVC flowers (2), FGVC fungi (3) and FGVC butterflies and moths (1) datasets. Following the approach of (Lin et al.) we extract the covariance matrix followed by signed square-root and ℓ_2 normalization from *relu*{*2_2,3_3,4_3, 5_3*} layers of VGG-16 network (32) and train a softmax layer to predict class labels. We train the model on the standard training split for birds and Oxford flowers and randomly select 100 images from the 200 categories with the most images for FGVC fungi, flowers, and butterflies.

Let C_i be the predicted probability from layer i. Then the maximal inverse image for a target class $\hat{C}$ is obtained as: $\min_{\mathbf{x}} \sum_{i=1}^{m} L\left(C_i, \hat{C}\right) + \gamma\Gamma(\mathbf{x})$. Here L is the softmax loss and $\Gamma(\mathbf{x})$ is the TV norm that acts as a smoothness prior. This technique was also used to visualize inverse images in (24). Figure 7 show the maximal images for various fine-grained categories along with their texture attributes. The maximal images indicate what discriminative texture properties are learned from training images for classification of instances which often appear in clutter, with wide ranges of pose and lighting variations, and under occlusions.

Describing maximal textures. We provide the preliminary experiments on describing these textures using attribute phrases that provide a language-based explanation of discriminative texture properties.

Using the phrase classification model trained on the dataset introduced in this paper, for each maximal texture image from the fine-grained categories, we generate a "phrase cloud" showing the top 20 attribute phrases, with the font size proportional to the predicted probability.

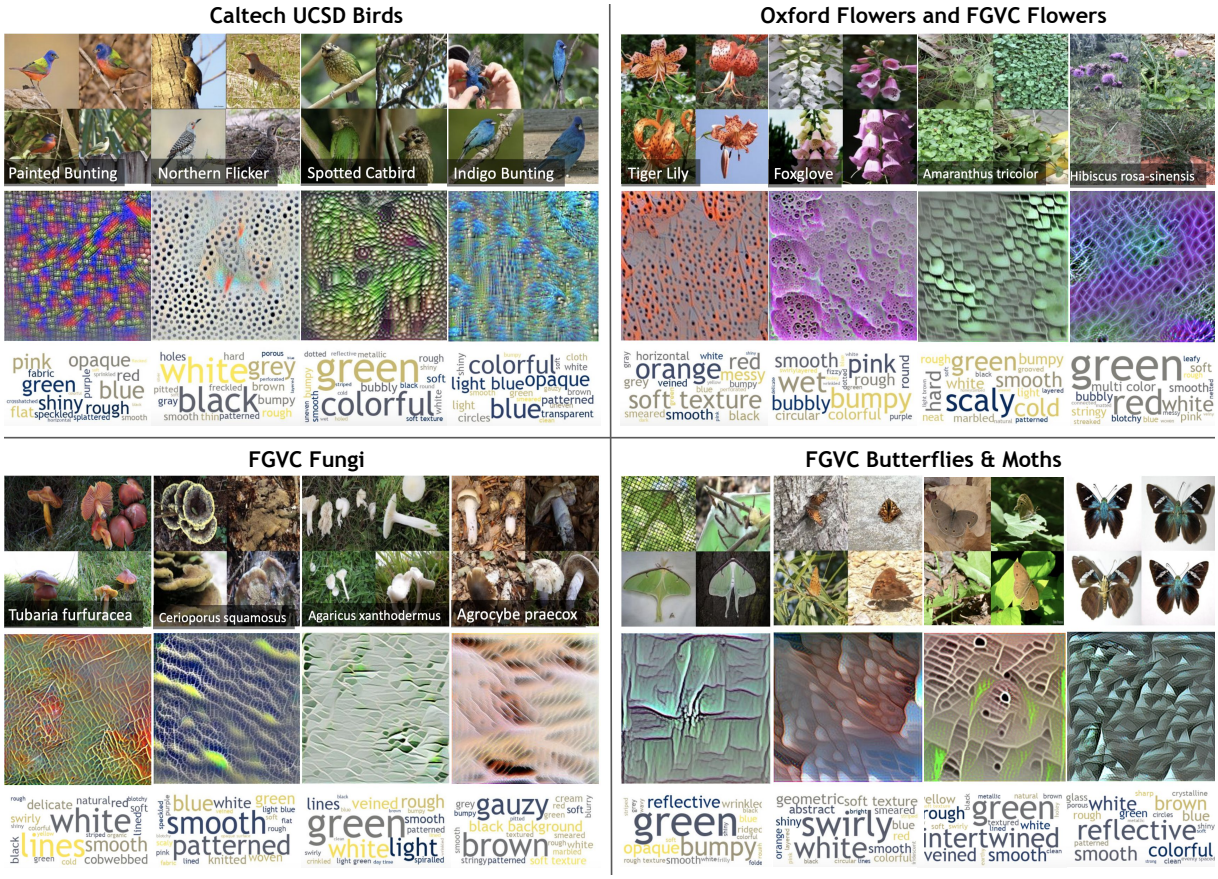

Figure 7: Fine-grained categories visualized as their training images (top row in each quadrant), maximal texture images (middle row in each quadrant), and texture attributes (bottom row in each quadrant). The size of each phrase in the cloud reflects its likelihood of being associated with the maximal texture.

As seen in Figure 7, these visualizations indicate what aspects of the texture are most discriminative for each fine-grained category while the descriptions provide a language-based explanation of the same.

8 Conclusion

Our new DTD^2 dataset provides a solid starting point for examining different methods for describing texture in images with rich natural language. Our experimental models do not currently have a good understanding of the compositional difficulties of visual textures. However, the models do perform reasonably well when performing image to phrase retrieval, and still decently well (though less so) on phrase to image retrieval. Additionally, we can use these models trained on DTD^2 to describe fine-grained categories of different flora and fauna using attribute phrases, giving us insight into what discriminative properties of texture help most in fine-grained classification.

Future work on this project will aim to explore different image representations for texture, including using the bilinear CNN to generate image embeddings as opposed to just ResNet-101. Another modeling approach to explore would be to train a metric learning model to minimize the distance between texture representations and phrase embeddings, but there are challenges associated with this task in choosing which negative phrases to sample given our missing data problem (just because a phrase is not in the ground truth for an image does not necessarily mean that it does not apply). Describing textures in different domains, such as fashion and home decor, would also be a worthwhile application to explore, as images in these domains have interesting materials, colors, and patterns.

References

[1] FGVC Butterflies and Moths Dataset, https://sites.google.com/view/fgvc6/competitions/butterflies-moths-2019.

[2] FGVC Flowers Dataset, https://sites.google.com/view/fgvc5/competitions/fgvcx/

flowers.

[3] FGVC Fungi Dataset https://sites.google.com/view/fgvc5/competitions/fgvcx/fungi.

[4] The Fifth Fine-Grained Visual Categorization (FGVC) Workshop https://sites.google.com/view/fgvc5.

[5] The Sixth Fine-Grained Visual Categorization (FGVC) Workshop https://sites.google.com/view/fgvc6.

[6] Bell, S., Upchurch, P., Snavely, N., and Bala, K. (2015). Material recognition in the wild with the materials in context database. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3479–3487.

[7] Bhushan, N., Rao, A. R., and Lohse, G. L. (1997). The texture lexicon: Understanding the categorization of visual texture terms and their relationship to texture images. *Cognitive Science*, 21(2):219–246.

[8] Chang, T. and Kuo, C.-C. (1993). Texture analysis and classification with tree-structured wavelet transform. *IEEE Transactions on image processing*, 2(4):429–441.

[9] Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., , and Vedaldi, A. (2014). Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

[10] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.

[11] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[12] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

[13] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

[14] Jégou, H., Douze, M., Schmid, C., and Pérez, P. (2010). Aggregating local descriptors into a compact image representation. In *CVPR 2010-23rd IEEE Conference on Computer Vision & Pattern Recognition*, pages 3304–3311. IEEE Computer Society.

[15] Julesz, B. (1981). Textons, the elements of texture perception, and their interactions. *Nature*, 290(5802):91.

[16] Karpathy, A. and Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137.

[17] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[18] Leung, T. and Malik, J. (2001). Representing and recognizing the visual appearance of materials using three-dimensional textons. *International journal of computer vision*, 43(1):29–44.

[19] Li, P., Xie, J., Wang, Q., and Gao, Z. (2018). Towards faster training of global covariance pooling networks by iterative matrix square root normalization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[20] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

[21] Lin, T.-Y. and Maji, S. (2016). Visualizing and Understanding Deep Texture Representations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2791–2799.

[Lin et al.] Lin, T.-Y., RoyChowdhury, A., and Maji, S. Bilinear Convolutional Neural Networks for Fine-grained Visual Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), volume=40, number=6, pages=1309–1322, year=2018, publisher=IEEE.*

[23] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.

[24] Mahendran, A. and Vedaldi, A. (2016). Visualizing deep convolutional neural networks using natural pre-images. *International Journal of Computer Vision (IJCV)*.

[25] Manjunath, B. S. and Ma, W.-Y. (1996). Texture features for browsing and retrieval of image data. *IEEE Transactions on pattern analysis and machine intelligence*, 18(8):837–842.

[26] Nilsback, M.-E. and Zisserman, A. (2008). Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP)*.

[27] Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

[28] Perronnin, F., Sánchez, J., and Mensink, T. (2010). Improving the fisher kernel for large-scale image classification. In *European conference on computer vision*, pages 143–156. Springer.

[29] Plummer, B. A., Wang, L., Cervantes, C. M., Caicedo, J. C., Hockenmaier, J., and Lazebnik, S. (2015). Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE international conference on computer vision*, pages 2641–2649.

[30] Reed, S., Akata, Z., Lee, H., and Schiele, B. (2016). Learning deep representations of fine-grained visual descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 49–58.

[31] Sharif Razavian, A., Azizpour, H., Sullivan, J., and Carlsson, S. (2014). Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 806–813.

[32] Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

[33] Varma, M. and Zisserman, A. (2005). A statistical approach to texture classification from single images. *International journal of computer vision*, 62(1-2):61–81.

[34] Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. (2015). Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164.

[35] Wah, C., Branson, S., Welinder, P., Perona, P., and Belongie, S. (2011). The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology.

[36] Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., and Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057.