Learning Deep Representations of Fine-Grained Visual Descriptions

Liangjie Cao

27 May 2018

1. Deep Structured Joint Embedding

Intuitively, the maximize the compatibility between a description and its matching image, and minimize compatibility with images from other classes. The objective is: given data $S = \{(v_n, t_n, y_n), n=1,..., N\}$ containing visual information $v \in \vartheta$, text descriptions $t \in \tau$ and class labels $y \in \Upsilon$. They seek to learn functions $f_v: \vartheta \rightarrow \Upsilon$ and $f_t: \tau \rightarrow \Upsilon$ minimize the empirical risk.

$\frac{1}{N} \sum \triangle (y_n, f_n(v_n)) + \triangle(y_n, f_n(t_n))$

where $\triangle: \Upsilon x \Upsilon \rightarrow \Re$ is the 0-1 loss. Note that N is the number of image and text pairs in the training set, and so a given image can have multiple corresponding captions. In practice they have many visual descriptions and many images per class. During training, in each mini-batch we ?rst sample an image from each class, and then sample one of its ten corresponding captions. Since their text encoder models are all differentiable, they backpropagate (sub)-gradients through all text network parameters for end-to-end training. For the image encoder, they keep the network weights ?xed to the original GoogLeNet.

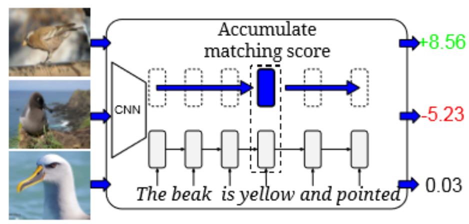

Figure 1. **Their model learns a scoring function between images and text descriptions.**

2. Text encoder models

In this section the authors describe the deep neural language models that we use for representing ?ne-grained visual descriptions. They compare the performance on zero-shot prediction tasks in Section 3. Text-based convolutional neural networks were studied in depth in [3] for the task of document classi?cation. The text-based CNN can be viewed as a standard CNN for images, except that the image width is 1 pixel and the number of channels is equal to the alphabet size. The 2D convolution and spatial max-pooling are replaced by temporal (1D) convolution and temporal max-pooling. After each convolution layer, they use recti?ed linear activation unit (ReLU), the overall network is constructed using convolution, pooling and thresholding activation function layers, followed by fullyconnected layers to project onto the embedding space. The text embedding function is thus simply ψ(t)=CNN(t); the ?nal hidden layer of the CN-N. Figure 2 illustrates the convolutional-recurrent approach. The ?nal encoded feature is the average hidden unit activation over the sequence. The resulting scoring function can be viewed as a linear accumulation of evidence for compatibility with a query image (illustrated in Figure 1). It is also a linearized version of attention over the text sequence. This has the advantage that at test time for classi?cation or retrieval, one can use the averaged hidden units as a feature, but for diagnostic purposes one can backtrace the score computation to each time step of text processing. They also evaluate a baseline that represents descriptions using unsupervised word embeddings learned by word2vec [2]. Previous works on visual-semantic embedding have directly used the word embedings of target classes for zero-shot learning tasks. However, in their case we have access to many visual descriptions, and they would like to extract vector representations of them in real time; *i.e.* without re-running word2vec training. A very simple way to do this is to average the word embeddings of each word in the visual description. Although this loses the structure of the sentence, this nevertheless yields a strong baseline and in practice performs similarly to bag of words. The CUB dataset also has per-image attributes, but they found that using these does not improve performance compared to using a single averaged attribute vector per class.

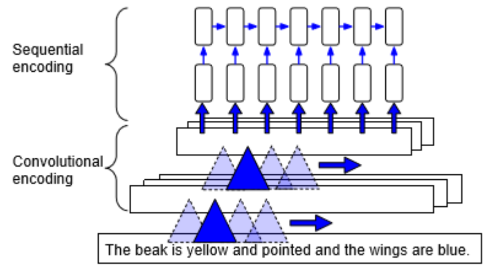

Figure 2. **Their proposed convolutional-recurrent net**

3. Experimental results

In this section they describe our experiments on the Caltech-UCSD Birds dataset (CUB) and Oxford Flowers102 (Flowers) dataset. CUB contains 11,788 bird images from 200 different categories. Flowers contains 8189 ?ower images from 102 different categories. Following [1], the images in CUB are split into 100 training, 50 validation, and 50 disjoint test categories. The CNN input size (sequence length) was set to 30 for word-level and 201 for character-level models; longer text inputs are cut off at this point and shorter ones are zeropadded. All text embeddings used a 1024-dimensional embedding layer to match the size of the image embedding. They kept the image encoder ?xed, and used RMSprop with base learning rate 0.0007 and minibatch size 40. I will continue learning in the following days.

References

[1] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Label-embedding for image classification. *IEEE TPAMI*, 38(7):1425–1438, 2016.

[2] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. *In NIPS*, pages 3111–3119, 2013.

[3] X. Zhang, J. J. Zhao, and Y. Lecun. Character-level convolutional networks for text classification. *In NIPS*, pages 649–657, 2015.