Noname manuscript No.
(will be inserted by the editor)

Supplementary Material: Feature Embedding for KNN-based Tag Prediction

Jianming Zhang · Shugao Ma · Mehrnoosh Sameki · Stan Sclaroff · Margrit Betke · Zhe Lin · Xiaohui Shen · Brian Price · Radomír Měch

Received: date / Accepted: date

This supplementary material explains how we learn the CNN feature embedding for the tag prediction system used in Sec. 6.2 of our manuscript of Salient Object Subitizing.

We learn the feature embedding for the K-Nearest-Neighbor (KNN) image search using the 6M training images from the Adobe Stock Image website (the same images as used for the KNN retrieval database). Each training image has 30-50 user provided tags. We pick about 18K most frequent tags for our dictionary. For each image, we keep the first 20 tags and extract a *term frequencyinverse document frequency* (TF-IDF) representation out of these tags. Formally, we have

$$\mathbf{tfidf}(t, d, D) = \delta(t, d) \log(1 + \frac{N}{n_t}), \tag{1}$$

where t is a word in the dictionary, d is the tag set associated with the given image, and D is the overall corpus containing all the tag sets associated with the images in our dataset. $\delta(t, d) = 1$ when $t \in d$ and 0 otherwise. N equals the number of images and n_t is the number of images that have the tag t.

Then the tag set associated with an image can be represented by a ∼18K-D vector where each entry is the TF-IDF value calculated by Eqn. 1. We L2-normalize these tag TF-IDF vectors and group these vectors into 6000 clusters using k-means. We denote each cluster as a pseudo-class and assign the pseudo-class label to the images belonging to this cluster. To obtain the image feature embedding, we train a CNN image classifier for the pseudo-classes. We adopt the GoogleNet architecture (Szegedy et al., 2015) and train the model using the softmax loss function. Instead of training the model from scratch, we perform fine-tuning on the GoogleNet model pre-train on ImageNet. We set the batch size to 32 and fine-tune the model for 3 epochs. The fine-tuning starts with a learning rate of 0.01 and we multiply it by 0.1 after each epoch. After that, the 1024D average pooling layer of the fine-tuned model gives the feature embedding for the KNN image retrieval.

Jianming Zhang, Mehrnoosh Sameki, Stan Sclaroff, Margrit Betke
Computer Science Dept., Boston Univ., Boston, MA USA
E-mail: {jmzhang,sameki,sclaroff,betke}@bu.edu

Shugao Ma
Oculus Research Pittsburgh, Pittsburgh, PA USA
E-mail: shugao.ma@oculus.com

Zhe Lin, Xiaohui Shen, Brian Price, Radomír Měch
Adobe Research, San Jose, CA USA
E-mail: {zlin,xshen,bprice,rmech}@adobe.com

References

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.