DeepCamera: A Unified Framework for Recognizing Places-of-Interest based on Deep ConvNets

Pai Peng Hongxiang Chen Lidan Shou Ke Chen Gang Chen Chang Xu
Zhejiang University, Hangzhou, China
{pengpai_sh, hxchen, should, chenk, cg, Xc0903}@zju.edu.cn

ABSTRACT

In this work, we present a novel project called DeepCamera(DC) for recognizing places-of-interest(POI) with smartphones. Our framework is based on deep convolutional neural networks(ConvNets) which are currently state-of-the-art solutions to vision recognition tasks such as our mission. We propose a novel ConvNet by introducing a new layer called "spatial layer" which captures spatial knowledge from a geographic view. As a result, both spatial and visual knowledge contribute to generating a hybrid probability distribution over all possible POI candidates. Furthermore, we compress multiple trained deep ConvNets into one single shallow net called "shNet" which achieves competitive performance with ensemble methods. Our preliminary experiments conducted on real-world dataset have shown promising POI recognition results.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Image Database

Keywords

places-of-interest; deep learning; image recognition

1. INTRODUCTION

The KPCB annual Internet Trends report(2014)[3] states that the global smartphone user base has hit 1.5 billion and all internet-connected citizens share over 1.8 billion photos every day via image-sharing social media platforms(e.g. Flickr and Facebook). Such large amounts of images have been a rich treasure to be explored, making a more intelligent and convenient life for human beings.

When a tourist is visiting a scenic spot for the first time, the fastest way to know the history of places-of-interest(POI) in front of him is using his accompanied smartphone to take a picture toward that POI. Then a pop-up of place name is expected to display as well as its related further detailed information. Our recent work called *DeepCamera*(DC) describes a technical framework to meet the demand of such scenario. The aim of DC is developing a system based on deep neural networks for recognizing outdoor POIs with the help of rich sensors embedded in smartphones.

Our solution relies on two assumptions: (1) a POI database which records the POI names and their exact geo-locations, (2) an image database which maps image IDs to their semantic POIs. These two assumptions do seem reasonable with availability of gazetteer such as [2] and numerous photos shared on the web. Then our goal is to identify the target POI for a query photo with camera geometries in smartphones, also known as *field-of-view*(FOV for short). A typical FOV includes four parameters: the GPS location, the angles of view, the maximum visible distance, and the direction of the smartphone camera.

Deep *Convolutional Neural Networks*(ConvNets) have shown powerful abilities to deal with vision recognition tasks since it cut a figure in ImageNet 2012[5] as the champion team's solution. The idea behind ConvNet is a layer-wise model which mimics human visual system and brain structure. Its major contribution is concluded as mapping low level raw pixels into high level semantic object categories via bottom-up layers. Different from handcrafted features based model such as bag-of-visual-words(BOW)[9], ConvNet automatically learns progressive features in different layers in the same way as we human beings see the world.

In spite of strong discriminative power of ConvNet, it still would be at loss what to do when images are too similar in visual content. For example, temples in China and Japan share high degree of outdoor appearance similarites. As a result, the cell in the probability distribution generated by ConvNet would be quite close in Chinese and Japanese temples categories. Then, it comes very naturally that the smartphone locations would help to eliminate such ambiguities. The FOV geometries can further restrict the POI candidates within a small portion of entire POIs, filtering out most POIs which are unlikely to be captured by camera(e.g. ones which are behind or too far away from current location).

Interestingly, recent work *KC*[6, 7] is similar to ours with the same goal of recognizing POIs. However, KC represents images as BOW, which means that local features have to be extracted previously. Moreover, they adopt sparse coding to compute visual similarities which is known to be computationally expensive. We propose a novel model which equips ConvNet with geographic features by adding a new layer called "spatial layer" in ordinary ConvNet. Such layer includes a spatial probability distribution over all POIs based on FOV parameters from a geographic view. Thus, both spatial and visual knowledge contribute to generating a hybrid

CIKM'15, October 19–23, 2015, Melbourne, Australia.

DOI: http://dx.doi.org/10.1145/2806416.2806620.

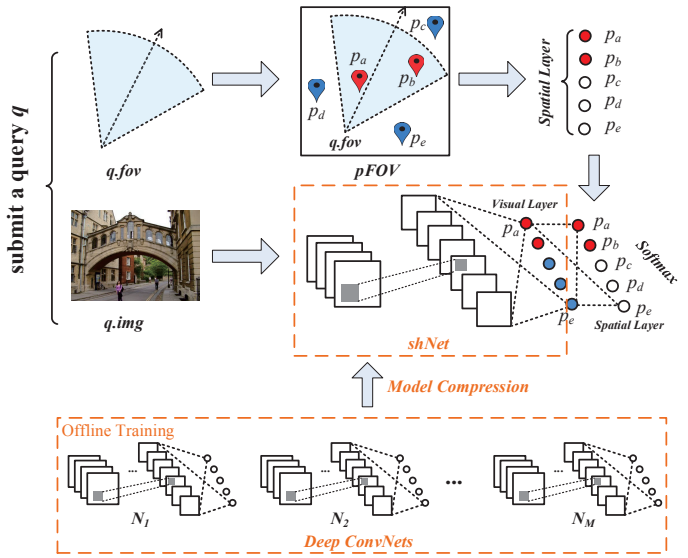


Figure 1: A toy example of DeepCamera framework. When a query q is issued, $q.fov$ and $q.img$ is processed in parallel. A shallow net called shNet compressed by multiple trained deep ConvNets(Section 2.2) will give a visual probability distribution for a given $q.img$(Section 2.1). The pFOV model filters out some unlikely POIs(p_c, p_d, p_e in our case) and outputs a spatial probability distribution which will be plugged into shNet(Section 2.3).

probability distribution over all possible POI candidates. One common way to boost performance of deep learning model is training multiple models by varying model parameters such as learning rate or different initializations and combine these pretrained models by ensemble methods such as bagging. However, such methods are not applicable to our mission since these ensemble methods usually decrease error variance by sacrificing online query efficiency while POI identification is expected to be done in real-time. Inspired by [4], we compress those pretrained deep models into one single shallow net called *shNet* which achieves competitive performance with ensemble methods.

Summary of contributions:

- We present a simple and effective framework for recognizing POIs in smartphones;
- We propose a novel ConvNet by adding a "spatial layer", thus equiping ordinary ConvNet with geographic features;
- We compress multiple deep ConvNets into one single shallow net with competitive performance in order to meet the demand of real-time;
- Experiments on real-world dataset has shown our model achieves as high as 97% accuracy for POI recognition.

2. DEEPCAMERA FRAMEWORK

We shall start by a definition of mathematical notation for DeepCamera. Suppose that we already have two databases: (1) a POI database P, where each POI $p \in P$ has geo-location attribute $p.loc$; (2) an image database S, where each image $s \in S$ has two attributes: a geo-location $s.loc$ and a POI tag $s.poi$ for its geographic and semantic label. A query q consists of two parts, namely the query image $q.img$ as the visual part and the query FOV $q.fov$ as the spatial part. We show a toy example of query processing in Figure 1.

We dive into the details of our DC framework in the remaining section as follows. *Deep ConvNet Model* outputs a visual probability distribution over all POIs. Then we train a shallow net via *Model Compression* in order to simultaneously increase accuracy and efficiency. Finally, we plug in the *Spatial Layer* which captures a geographic probability based on a query FOV.

2.1 Deep ConvNet Model

In this subsection, we describe our deep ConvNet model in two parts, namely the *Network Architecture*(detailed in Section 2.1.1) and *Network Training*(detailed in Section 2.1.2).

2.1.1 Network Architecture

We summarized 3 similar deep network in Table 1. The receptive fields in convolution layer are all set to 3×3 since deep model with small kernels has two advantages [8]: (1) makes the final decision function more descriminative; (2) descreases the number of parameters. The input size of our ConvNet is fixed as $3 \times 128 \times 128$(3 channels for RGB) subtracted from the mean RGB value of training set. The convolution stride is fixed to 1 pixel and the padding is also 1 pixel so that the output size is preserved after convolution operation.

All together, we add 5 MaxPooling layers as a downsampling technique among the stack of convolutional layers. The MaxPooling window is 2×2, with stride 2. Three fully-connected(FC) layers are stacked on top of convolutional layers. The first two FC layers have 1024 neurons with a Dropout[10] probability 0.5 and the last one's size depends on the number of POIs. The non-linear transformation we use is Rectified Linear Unit(ReLU)[5] after each convolution layer.

2.1.2 Network Training

The network training procedure is optimized by mini-batch gradient descent with RMSProp[1] which gains popularity very recently. The mini-batch size is set to 128. In addition to Dropout layers, we also introduce a L_2 weight decay as regularization(the penalty coefficient is 0.001). The initialization of learning rate is set 0.01 and it is decreased by multiplying 0.98 for every other 10 epochs until it reaches 0.0001. The data augmentation strategy is cropping rescaled images for each epoch iteration and then conduct randomly horizontal flipping and contrast(between 0.8 to 1.2).

2.2 Model Compression

A simple but effective way to boost a machine learning algorithm is to train many models previously and then combine these models to obtain a single model with higher accuracy. Such combination is called *ensemble methods*. However, the increasing accuracy is gained at the expense of efficiency since it will compute multiple models. Thus, we are motivated to obtain a shallow net(which we call *shNet*) but still with competitive performance. Next, we will briefly introduce two simple ensemble methods in Section 2.2.1 and describe how to train the shNet in Section 2.2.2.

2.2.1 Ensemble Methods

Simple Averaging Suppose we have already trained n models M_1, M_2, ..., M_n and there are m POIs(or classes) in total and N validation images. Then for a given query image

Table 1: Deep ConvNet Architecture(ReLU is not shown, XXX represents the number of POIs).

ConvNet Name	ConvNet Architecture
N_1(9 layers)	Input → Conv3-64 → MaxPool → Conv3-128 → MaxPool → Conv3-256 → Conv3-256 → MaxPool → Conv3-512 → MaxPool → Conv3-512 → MaxPool → FC-1024 → FC-1024 → FC-XXX → Softmax
N_2(11 layers)	Input → Conv3-64 → MaxPool → Conv3-128 → MaxPool → Conv3-256 → Conv3-256 → MaxPool → Conv3-512 → **Conv3-512** → MaxPool → Conv3-512 → **Conv3-512** → MaxPool → FC-1024 → FC-1024 → FC-XXX → Softmax
N_3(13 layers)	Input → Conv3-64 → **Conv3-64** → MaxPool → Conv3-128 → **Conv3-128** → MaxPool → Conv3-256 → Conv3-256 → MaxPool → Conv3-512 → **Conv3-512** → MaxPool → Conv3-512 → **Conv3-512** → MaxPool → FC-1024 → FC-1024 → FC-XXX → Softmax

q and a model M_i, let $\mathbf{P}(q = p_j; M_i)$ denote the probability that the query image q is predicted as POI $p_j (j = 1, 2, ..., m)$ under the model M_i. Then the simple averaging ensemble is:

$$\mathbf{P}(q = p_j; M_1, ..., M_n) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{P}(q = p_j; M_i) \quad (1)$$

Weighted Averaging The simple averaging method consider each model as equal importance. However, models with high accuracy may be assigned with high votes which leads to a weighted averaging method. Assume α_i is the accuracy of model M_i on validation dataset, then the weighted averaging ensemble is:

$$\mathbf{P}(q = p_j; M_1, ..., M_n) = \sum_{i=1}^{n} \frac{\alpha_i}{\sum_{i=1}^{n} \alpha_i} \mathbf{P}(q = p_j; M_i) \quad (2)$$

2.2.2 *shNet Training*

After we have obtained an ensembled deep model M using methods in Section 2.2.1, we would like to train a shallow net shNet which mimics M. More concretely, shNet is trained to fit the *function* that learned by M. Note that such function is so complex that it cannot be learned by a shallow model on the original labels. Hence, we have to learn the function with the help of intermediate deep model first and then train another shallow model to mimic such function.

The last layer in M is a Softmax normalization to produce a probability distribution over different POIs. However, it is not reasonable to directly train the shNet on these probabilities since they are sometimes too difficult to be learned by cross-entropy loss function[4]. Thus the shNet is trained on *logits* which are values before the Softmax transformation. Training in the logit space will be much easier for shNet to capture the relationships between different POIs than probability space. For example, consider three POI probabilities output by M after Softmax [$2e{-}9$, $4e{-}5$, 0.9999]. It can be seen that the third POI dominate the probability space, hence, the information hidden in the first two POIs will not be captured if we train shNet on these probabilities, e.g. although the second POI is unlikely to be the target but it is still more likely than the first POI which may indicate that the last two POIs share similarities to some extent. The corresponding logit values are [10, 20, 30], which mitigates the serious bias in probability space, making shNet easier to learn.

The last important thing to note is that cross-entropy is not any longer applicable for training shNet as cost function since it will only focus on the true label's cell. Thus, we formulate the learning process as a L_2 loss regression problem in the logit space as detailed in [4].

Figure 2: Example of query photos in two dataset.

2.3 Spatial Layer

In work [6], it has been proposed a *probabilistic FOV model*(pFOV) which can estimate the likelihood of some POI being captured by a smartphone camera. The idea behind is to leverage the uncertainty of the phone's GPS location and build up a gaussian-based model to capture such uncertainty. In our case, we will obtain a geo-spatial probability distribution over different POIs via pFOV model which forms our "spatial layer". We call the last layer after Softmax in shNet "visual layer" since it represents probabilities in the visual space. Then we plug a new "spatial layer" on top of the "visual layer". The weight matrix $\mathbf{W}$ between these two layers is:

$$\mathbf{W} = \beta \cdot \mathbf{\Sigma}(1, 1, ..., 1) \quad (3)$$

where β is a weighted preference for spatial phase and $\mathbf{\Sigma}$ is a diagonal matrix with all ones in the diagonal. β is a hyper-parameter which should be tuned in validation set.

3. EXPERIMENTAL RESULTS

3.1 Dataset Description

We conduct our experiments on two different dataset, namely *Singapore 151K*(S) and *NewYork 296K*(N).

Singapore 151K Singapore 151K is provided by [6] which contains 151,193 geotagged images with 2,256 distinct POIs. They also collect a query set consisting of 680 smartphone captured photos(with the ground truth POIs) and each photo is associated with FOV parameters.

NewYork 296K In total, 296,156 geotagged images associated with 1,512 different POIs are downloaded from Flickr. We get the POI list in NewYork region from [2] and use these POI names as query keywords via Flickr API. Unlike Singapore 151K, we were not able collect an extra query set acquired by smartphones. We select 500 images as query photos, which do not contain camera direction obviously. In such case, we slightly modify equation 1 in [6] by remov-

Table 2: Results of Different Models

Model	Acc(S)	Acc(N)	Time(S)	Time(N)
KC[6]: Spatial($\lambda = 1$)	48%	-	-	-
KC: Visual($\lambda = 0$)	77%	-	-	-
KC: Spatial+Visual($\lambda = 0.5$)	92%	-	0.68s	-
N_1	85.4%	78.2%	0.0030s	0.0017s
N_2	87.5%	82.7%	0.0041s	0.0026s
N_3	88.3%	83.1%	0.0048s	0.0035s
Averaged Ensembles(AE)	89.0%	84.3%	0.0962s	0.0645s
Weighted Ensembles(WE)	91.2%	85.8%	0.1138s	0.0769s
WE+SpatialLayer	**97.0%**	**91.1%**	0.1619s	0.1084s
shNet	88.4%	81.2%	**0.0013s**	**0.0011s**
shNet+SpatialLayer	95.3%	88.8%	0.0494s	0.0326s

ing the first item, i.e. the probability distribution now only depends on the distance d.

3.2 Performance Results

We summarize the performance results of different models in Table 2. Note that these results are averaged over all query set. Experiments are conducted on a Linux Server with 64GB memory and GPU mode is NVIDIA-TITAN X.

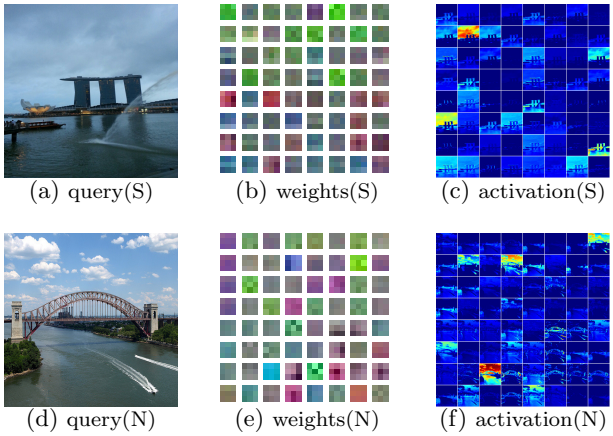
The first three models(as our baseline) and results are reported in KC[6] in the case of query box = 0.6 km, which performs best when cross validated. Our best single model of deep ConvNet N_1, N_2, N_3 acquires 85%, 87%, 88% recognition accuracy in Singapore dataset, all exceeding the pure visual phase in KC. In order to obtain averaged and weighted ensembles, we vary the ConvNet with different initialization, learning rate, gradient descent algorithm(such as normal SGD, Momentum and AdaGrad). The weighted ensembles perform better than simple averaging as expected, rise to 91%. The shNet architecture is as suggested in [4], a 3-layer network with one pooling layer. For the sake of saving space, we only report the best shNet which possesses roughly 32M parameters. Note that such best shNet accuracy is still lower than ensembles since shNet is a model which attempts to *approximate* the function learned by ensembles. Nevertheless, it considerably reduces online query time compared with ensembles. Finally, we combine spatial layer into ensembles and shNet. As a result, it increases recognition accuracy significantly due to geo-spatial remediation embedded in pFOV model. Overall, our shNet+SpatialLayer increases recognition accuracy and query response time is $\times 13$ faster than [6]. Same conclusions can be drawn from New York dataset.

3.3 ConvNet Visualization

Due to limited space, we plot the weights in the first layer learned by network N_1 for Singapore dataset in Figure 3(b) and NewYork dataset in Figure 3(e). It can be seen that there are 64 weights, which corresponds to the number of feature maps in the first convolutional layer, with kernel size 3×3. We also show 2 query sample images and their 64 activations in the first layer in Figure 3. Note that with 1 pixel padding, each activation output size keeps the same as input.

4. CONCLUSION

We have presented a novel framework DeepCamera for recognizing places-of-interest with smartphones. Our framework is based on Convolutional Neural Networks. In order to respond user queries in real-time, we compressed multiple



(a) query(S) (b) weights(S) (c) activation(S)

(d) query(N) (e) weights(N) (f) activation(N)

Figure 3: Visualizing deep ConvNet N_1.

trained ConvNets into one single model shNet which proves to achieve competative performance with ensemble methods. A spatial layer is stacked on top of shNet to combine the geographic features. Our framework is evaluated on real-world dataset and beats the baseline.

Acknowledgments

The work is supported by the National Key Basic Research Program of China (GrantNo. 2015CB352400), National High Technology Research Development Program of China (GrantNo. 2013AA040601), and the National Science Foundation of China (GrantNo. 61170034).

5. REFERENCES

[1] http://www.cs.toronto.edu/~tijmen/csc321/slides/.
[2] http://www.geonames.org/.
[3] http://www.kpcb.com/internet-trends.
[4] L. J. Ba and R. Caurana. Do deep nets really need to be deep? *CoRR*, abs/1312.6184, 2013.
[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105. 2012.
[6] P. Peng, L. Shou, K. Chen, G. Chen, and S. Wu. The knowing camera: Recognizing places-of-interest in smartphone photos. In *SIGIR*, pages 969–972, 2013.
[7] P. Peng, L. Shou, K. Chen, G. Chen, and S. Wu. The knowing camera 2: recognizing and annotating places-of-interest in smartphone photos. In *SIGIR*, pages 707–716, 2014.
[8] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.
[9] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *ICCV*, pages 1470–1477, 2003.
[10] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JLMR*, 15:1929–1958, 2014.