Deep-Crowd-Label: A Deep-Learning based Crowd-Assisted System for Location Labeling

Mohammad-Mahdi Moazzami
Computer Science Innovation Center
Samsung Research America

Jasvinder Singh
Computer Science Innovation Center
Samsung Research America

Vijay Srinivasan
Computer Science Innovation Center
Samsung Research America

Guoliagn Xing
Computer Science Department
Michigan State University

***Abstract*—Semantic labels are crucial parts of many location-based applications. Previous efforts in location-based systems have mostly paid attention to achieve high accuracy in localization or navigation, with the assumption that the mapping between the locations and the semantic labels are given or will be done manually. In this paper, we propose a system called Deep-Crowd-Label that automatically assigns semantic labels to locations. We propose a novel transfer learning method that leverages deep learning models deployed on many crowd-workers to assign semantic labels to locations by classifying associated visual data. Deep-Crowd-Label uses the power of the crowd to aggregate the individual predictions done by the model across the crowd-workers visiting the same location. Our preliminary experiments with 26 different types of locations show that, our method and our prototype system is able to find the right label for the locations i.e., coffee shop to the Starbucks.**

I. Introduction

The ubiquity of smartphones and their multi-modal sensing capabilities have enabled a wide spectrum of mobile sensing applications. These applications are usually human-centric in that the smartphone utilizes on-board sensors to sense people and characteristics of their contexts [16],[15]. On the other hand, location is one of the most widely used contextual signal in everyday activities. Therefore, one of the most important people-centric pervasive applications is in providing location-based services with contextual information about the location.

Prior work has shown how places can be discovered from temporal streams of user location coordinates [6]. However, if we can automatically characterize places by linking them with attributes, such as place categories (e.g., clothing store, restaurant), we can realize powerful location- and context-based scenarios [19]. Indoor maps and semantic understanding of a place are equally crucial pieces of the bigger localization puzzle, but only recently researchers began to focus on these aspects. In this paper, we present *Deep-Crowd-Label* a framework for semantically labeling the user's location. Despite the generality of our approach we focus on the processing of indoor locations, in which the people spent a big portion of their time. In our approach we focus on the location-tagged visual data i.e., images and videos and use deep neural networks [12] as the core of our processing pipeline. The choice of deep neural networks is due to the availability of large-scale image datasets and the promising performance of the recent methods developed in this area that outperform the other data mining and machine learning methods. Examples include the recent algorithms developed for object detection and scene understanding [29], [22], [9], [13].

Fig. 1: An indoor area with semantic labels

On the other hand, the advantages of crowd-sourcing in this work is two folded. First, there are already available big crowd-sourced datasets like ImageNet [7] and places [1] that makes the training of the deep neural models possible. Second, in addition to the crowd-sourced datasets, by deploying the trained models in many crowd-workers, crowd-sourcing enables us to have a much more reliable labeling via aggregating the predictions done by the crowd-workers (crowd-prediction). Therefore, crowd-sourcing is used both in the training (crowd-**sensed** training data) and in the inference (crowd-**prediction**). In particular, Deep-Crowd-Label leverages the crowd-sensed data to build a big dataset suitable for training deep neural network, and proposes novel techniques based on *model adaptation* and *model extension via transfer learning* to exploit the pre-trained models (which they are also trained on the crowd-sourced datasets) to build a robust and accurate prediction model for semantically labeling the user's location. Since training deep neural networks requires a lot of data to prevent overfitting[4], Deep-Crowd-Label retrofits the pre-trained models via novel transfer learning techniques and builds ensemble

978-1-5090-4338-5/17/$31.00 ©2017 IEEE

of models to cope the over-fitting problem. Moreover, Deep-Crowd-Label proposes a probabilistic averaging algorithm to use the power of crowd in aggregating the individual crowd-workers predictions in generating the final location labels.

II. Related Work

Various semantic localization schemes based on crowd-sourcing by smartphones have been developed recently. SurroundSense [3] and SenseLock [10] utilize sensor data from smartphones to characterize the ambiance and translate them into a semantic information about the user's location. Authors in [21] and [26] attempt to categorize places by training a model on WiFi, GSM and sensor data collected from frequently visited places. These approaches are based on the assumption of availability of labeled ambiance data. Work by [5] tries to connect the text in the crowd-sensed pictures with the posts in social networks to infer business names. AutoLabel [14] aims to automatically identify the name of the store by correlating the words inside the store's WiFi-tagged pictures with the keywords found in the store's website to produce a WiFi AP to StoreName table. Perhaps more closer to our approach is work by [27] that attempts to identify the stores that appear in a photo by matching it against the images of the exteriors of the nearby stores extracted from the web. This approach relies on the conventional computer vision techniques and are neither scalable nor robust. In contrast, on the one hand, Deep-Crowd-Label does not rely on the conventional computer vision techniques and uses deep neural networks that has recently driven remarkable performance in computer vision and machine learning. On the other hand, Deep-Crowd-Label relies on crowd-sourcing on both training and inference phases that increases the generality and robustness of the system. For these two reasons, we believe Deep-Crowd-Label addresses many shortcomings of the existing approaches while not replacing them but playing as a valuable complementary technique to them.

III. Method

Deep neural networks learn a number of hierarchical feature representations and have two major benefits. First, deep neural networks do not need hand-crafted features which allows us to deploy the models in real applications easier [12]. Second, the deep features learned by deep neural networks, in comparison with conventional hand-crafted features, are more generic and extremely robust which allow us to use the learned features in one domain (the source domain) to build the models for another domain (the target domain), e.g., object detection to scene understanding. The latter is the spirit of our method in designing of our processing pipeline and is explained as follows.

A. Domain Adaptation with Pre-Trained Models

Domain adaptation aims at training a classifier in one problem space and applying it to a related but not identical problem [28]. We adopt existing practice in DNN modeling, called pre-trained models [8], and apply them to our location

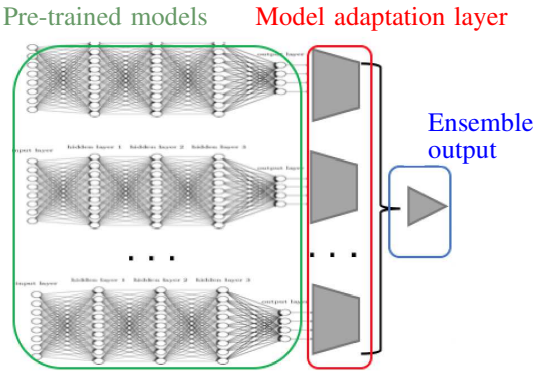

Fig. 2: Our model adaptation schema and ensemble of adapted deep neural models. Left/Green: Several deep neural models pre-trained or extended using transfer learning. Middle/Red: The adaptation layer. Right/Blue: The aggregation layer.

labeling problem. Our "domain adaptation" in this case is limited to the "label space adaptation", that is adopting the output of the final layer without tuning the learned parameters (weights) or the internal network structure. In this approach, the pre-trained models trained with arbitrary number of class labels used in the task of classifying the images that cover only a subset of classes identifying the location context i.e., store types. Table I summarizes the pre-trained models used in our domain adaptation mechanism. The final layer of these DNNs commonly use the SoftMax classifier [4]. For this layer we have:

$$P(y=j|X_i) = \frac{e^{w_j X_i^T}}{\sum_{k=1}^{n} e^{(w_k X_i^T)}} \quad (1)$$

where X_i is the feature vector extracted by the deep neural network for the input sample i (captured single image). w_i is the weight learned by the neural network. y is the predicted class label in $j \in \mathsf{N}$ the set of all the class labels a pre-trained model is trained on (the source domain). For example the size of label space, $|\mathsf{N}|$, for original AlexNet [11] trained for ImageNet [7] is 1000 labels. In order to adapt such a pre-trained model for the task of interest (the target domain), we follow the Bayesian chain rule [4] and apply the prior knowledge specific to the space to the model prediction and thus we have:

$$P_s(y=j|X_i) = \frac{1(y \in \mathbb{L}) \cdot P(y=j|X_i)}{\sum_{\mathbb{L}} 1(y \in \mathbb{L}) \cdot P(y=j|X_i)} \quad (2)$$

where $1(.)$ is the identity function and $\mathbb{L}$ is the label-set of the application the pre-trained model is adopted for (the target domain). The denominator is the normalization factor and thus $P_s(y=j|X_i)$ indicates the probability of class(label) given the feature vector X_i for application specific labels $j \in \mathbb{L}$. Therefore, given a pre-trained model $\mathbb{M}$ with the label space $\mathbb{N}$ in the source domain, and the loss function as shown in equation 1, our domain adaptation approach adapts the model $\mathbb{M}$ for the target application with label space $\mathbb{L} \subset \mathbb{N}$ using the

equation 2. Fig. 2 illustrates the layouts of our pipeline. The adaptation layer in this figure implements equation 2.

B. Model Extension with Transfer Learning

Training deep neural networks requires very large datasets and it is relatively unusual to have a proper dataset with sufficient size for many applications. Therefore, it is not always practical to train an entire deep neural network from scratch. Instead, it is common to pre-train a deep neural net. on a very large standard dataset i.e. ImageNet, which contains 1.2 million images with 1000 categories or Places dataset [25] with 500K images with 205 categories and then use the the resulting model (the pre-trained model) either as an initialization or a fixed feature extractor to build a final model for your application. In the previous section, we described how the pre-trained models are adapted to our application without further training. This approach is suitable for the cases that the target labels are a subset of original labels, $\mathbb{L} \subset \mathbb{N}$. However, in our problem, none of the original models entirely include our target label space. In other words: $\mathbb{L} \not\subset \mathbb{N}$. Therefore, we have two cases: a) There are class labels in $\mathbb{L}$ that do not have any high level representation in the pre-trained models label space $\mathbb{N}$ e.g., computer store is missing in the ImageNet label space. b) There are class labels in $\mathbb{L}$ that match with more than one class label in $\mathbb{N}$ e.g., shoe-shop has multiple corresponding categories in AlexNet-ImageNet model (i.e., shoe, loafer shoe or sport shoe). To address this issue we propose an approach to extend the pre-trained base models using "Transfer Learning" [18]. In this approach we keep the feature extractor layers[1] in the pre-trained model frozen by setting the learning rate for these layers to zero. The last fully connected layer instead is initiated with random weights and is trained with the data collected and labeled by us. This allows us to use the features extractors trained in the pre-trained models and train the final fully connected layers using our collected data and extend the model to cover the entire label space $\mathbb{L}$. This also enables us to train a deep model with limited amount of training data with no over-fitting.

C. Model Ensemble

To further improve the accuracy and increase the robustness of our label prediction model (for single image) we use an ensemble of models as illustrated in Fig. 2. Our ensemble model is simply the weighted average of prediction probabilities of the individual models that are either adapted or extended by the methods explained in sections III-A and III-B. Fig.2 illustrates this approach and Table I summarizes these models.

IV. Training

We train our network with a combination of available public datasets and our collected data using video frames and still images. The training is done when the Model Extension (ME) approach (cf. Sec. III-B) is used. Each model in this approach is trained independently regardless of what kind of pre-trained model is chosen as the base model. In the training process, the convolutional layers are taken from the convolutional layers in the base model e.g., [8], [11]. We pass the output of these convolutional layers (i.e. the pool5 features) into a single feature vector. This vector is the input to the fully connected layers taken from the base model and trained on our dataset. Finally, we re-define the last layer (i.e., softmax layer) to have outputs equal to the number of classes in our label space. During the training procedure the learning rate is set to zero for the convolutional layers. This is because we do not have enough data to train these layers and thus by freezing these layers (learning rate = 0) it will prevent our model to over-fit to the training data and at the same time taking advantages of the pre-trained model as a feature extractor. The learning rate for the fully connected layers are taken from the default for the base layers while the learning rate for the output layer is set to 10 times of the maximum of learning rates of the fully connected layers. This is because the last layer is designed specifically for the target domain and, unlike intermediate fully connected layers in the model, there is no layer in the base model corresponding to this new output layer and thus the weights in this layer has to be initialized randomly. Therefore this requires us to have the layer trained faster than other fully connected layers.

Beside the learning rate and the structure of the output layer, the other network hyper-parameters are taken from the base models. In our model we use ReLU [17] as the activation function in each convolution layer interleaved with pooling layers as in the base models. Our neural network is trained using Caffe [8].

V. Labeling and Aggregation by Crowd-Sourcing

Once the model is generated via the process explained in Sections III and IV, it's ready to be deployed and use the power of crowd for location labeling. This is basically the final stage of our labeling pipeline. In this step the individual prediction results from each crowd-worker for a location k are aggregated via the following *weighted* averaging mechanism. Let $P_I(y = j|X_i^k)$ be the prediction probability result by our ensemble of deep neural network models (cf. Sections III-C) for classifying an image i obtained by a crowd-worker at location k, represented by feature vector X_i^k . Also let Γ_k indicate the set of all images collected for location k by all crowd-workers. Therefore we have:

$$P_\Gamma(y = l|\Gamma_k) = \frac{1}{|\Gamma_k|} \sum_{X_i^k \in \Gamma_k} P_I(y = j|X_i^k) \qquad (3)$$

where $P_\Gamma(y = l|\Gamma_k)$ is the aggregated predication for location k cross over all the images obtained by all crowd-workers for that location. Since there is no difference between different crowd-workers and neither between images obtained by them, all images obtained by all crowd-workers are simply put in the set Γ_k. The final label for each location is obtained by simply picking the label with the maximum aggregated probability, in other words we have:

[1] convolutional layers

TABLE I: Our Model Adaptation and Model Extension of Deep Neural Networks

DNN Model	Architecture$^{\alpha}$	Dataset (# of classes/ total size)	Method$^{\mathfrak{M}}$
imagenet-alexnet [11]	5(CV), 3(FC), 1000(O)	ImageNet(1000/1.2M)	MA
places-alexnet [29]	5(CV), 3(FC), 205(O)	Places (205/2.5M) [29]	MA
places-hybrid [25]	5(CV), 3(FC), 1183(O)	ImageNet (978) + Places(205)	MA
places-googleNet [25]	59(CV), 5(FC),205(O)	Places (205/2.5M)	MA
shops-alexnet	5(CV), 3(FC), 26(O)	Places205 $\Longrightarrow$ shops from places + SUN397 [24] dataset (205/2.5M)	ME
indoor67-alexnet	5(CV), 3(FC), 67(O)	Indoor 67 [20] (67/15.6K)	MA
indoorshops-alexnet	5(CV), 3(FC), 26(O)	ImageNet $\Longrightarrow$ indoor shops selected from SUN + places $\Longrightarrow$ our data (15/1.5k)	ME
imagenet-stores-alexnet	5(CV), 3(FC), 9(O)	ImageNet $\Longrightarrow$ indoor shops from ImageNet (9/10k)	ME

α: FC: Fully-connected Layer, CV: Convolution Layer, O: Output (number of classes)
$\Longrightarrow$: indicates the direction of transfer learning: base model $\Longrightarrow$ new model.
$\mathfrak{M}$: MA: Model Adaptation (Sec. III-A), ME: Model Extension (Sec. III-B).

$$label_k = \arg\max_{l} P_{\Gamma}(y = l|\Gamma_k) \tag{4}$$

Note that the advantages of this approach to the voting approach is that in our approach not only it considers individual crowd-worker's result but also it takes the individual confidence levels as the prediction probability distribution over the label space into the account. Thus, if two crowd-workers label a location as a "coffee-shop" but with different probabilities, their contribution to the final prediction probability $P(y = "coffee.shop"|\Gamma_k)$ is proportional to their individual prediction probabilities.

VI. Embedding Models into the Smartphone

Although many users are uploading images to many available cloud-based services (i.e., Google places Facebook, Yelp) everyday, many other users may have stringent concerns in sharing image data because image sharing might be quite sensitive and intrusive. On the other hand, uploading every single image by smartphones is not energy efficient. Therefore, a salient feature of Deep-Crowd-Label is that it deploys the deep neural model on the smartphone of a crowd-worker and if a crowd-worker prefers to not share the images with our cloud back-end, the system performs the image labeling on the device and uploads the prediction results instead. This means the Deep-Crowd-Label performs one feed-word execution on the deep model for each input image and uploads the classification results including the probability values as a serialized json text-file to the cloud server. The server component aggregates all the inferences it receives from all the crowd-workers as discussed in Sec. V. Not only uploading a json text-file is a lot more energy efficient than uploading the image files, but also it addresses the privacy concerns that users may have.

However, embedding deep neural models into the smartphone is not straight-forward. Deep learning models are both computationally intensive and often take up a lot of space resulting unacceptable app sizes. For example, original AlexNet [11] or VGGNet [23] are more than 200 or 520 MB respectively. In our work, to shrink the file size, we apply the quantization method developed by Pete Warden and et al. [2] that takes advantage of the weights format in a trained model. In nutshell this method squeezes(quantizes) each float value (32 bits) in the model's weight matrices to the closest integer number (8 bits) resulting to a much smaller model size. By using this technique we reduce the file size of each model significantly ($\sim$ by 75%) making it easier to deploy on mobile devices. Note that this method does not improve the computation or run-time memory consumption of the models. We leave further optimization of the model deployment for our future work.

VII. Data Collection and Dataset Preparation

Data Collection: We have collected data from 26 different indoor locations, mostly shops in the malls and supermarkets. The data is collected using smart-watch and smart-phone in the form of videos and still images. The videos are converted to the frames. It is important to remove the very similar frames in the training data to prevent bias in the model. Therefore we only extract the key-frames from the video using FFMPEG. Moreover, the 80% data is left for training and the 20% of is used for inference (labeling). Since having a balanced dataset is crucial in the training phase, the number of images per class is kept balanced in the training set. This is not necessary in the inference phase.

Automatic Rotation and Noise Reduction: During data collection we observed that the camera API on each device rotates the captured image arbitrary. Since our deep neural network models are not rotation-invariant, we use the camera information in the images Exif meta-data, to rotate the images to the right orientation automatically. In addition, we apply standard denoising method to improve the quality of collected images. Images that the measure of blurriness is above a certain threshold are not used in the training data.

Data Augmentation: Deep neural networks require a lot of data to train. The easiest and the most common method to cope with data scarcity is to artificially enlarge the dataset a.k.a as data augmentation. Following the technique in [11], we augment our dataset by extracting random 224×224 patches (and their horizontal reflections) from the 256×256 images and training our network on these extracted patches. This increases the size of our training set by a factor of 2048, though the resulting training examples are, of course, highly interdependent [11]. Without this scheme, our network suffers from substantial over-fitting even with transfer learning (cf. Sec. III-B).

Fig. 3: Predictions on the real samples collected from indoor shops. Bars below each image show the top-5 model predictions using our deep learning method sorted in ascending order.

TABLE II: Location labeling results. Each table represents one store with name and grand-truth type (top row). Top-5 prediction results with confidence values (prediction probabilities) are presented in each row. Each prediction is the aggregated result of crowd-sensed images for each store (Sec. V).

(a)

Safeway	supermarket
68.52%	supermarket
6.31%	cottage-garden
5.06%	crevasse
4.89%	valley
4.81%	mountain

(b)

Macy's	clothing-store
35.71%	clothing-store
5.75%	gift-shop
3.99%	staircase
3.81%	shoe-shop
3.13%	beauty-salon

(c)

Disney Store	gift-shop
52.60%	gift-shop
11.97%	candy-store
5.18%	market
3.06%	game-room
2.42%	supermarket

(d)

Apple Store	computer store
16.47%	computer store
6.42%	food-court
6.38%	art-gallery
5.49%	cafeteria
5.26%	art-studio

(e)

Zara	clothing-store
40.40%	clothing-store
5.20%	garbage-dump
3.98%	slum
2.91%	excavation
2.83%	railroad-track

(f)

DSW	shoe-shop
19.15%	bookstore
11.39%	airport-terminal
7.32%	shoe-shop
6.32%	supermarket
4.86%	clothing-store

VIII. Evaluation

Algorithmic Performance: Our proposed method is applied to the real data collected from 26 stores. Fig. 3 shows the prediction results of our pipeline when it labels a single image. The figure shows top-5 prediction results for each image with confidence values in the bar chart in increasing order. Moreover, Tables II -a:f show the results of the aggregated predictions for 6 different indoor locations (5 different kinds of stores and 1 food court) in a shopping mall. For each location the grand-truth label is mentioned on the top-right cell and

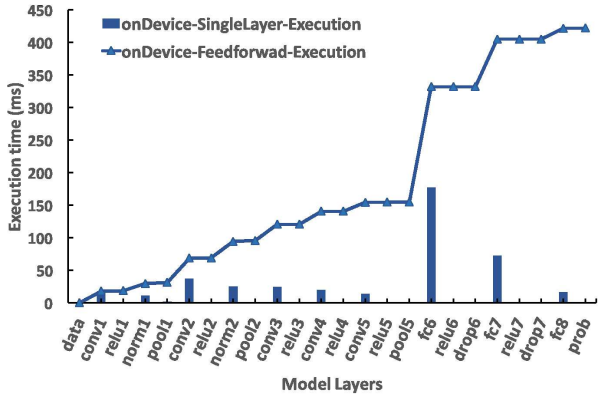

Fig. 4: The execution time of our model (base model=AlexNet) on Galaxy-S7 mobile phone with layer-wise latency breakdown.

the top-5 prediction results are reported in descending order of confidence values. Although the confidence values varies for different views, the results show the significant difference between the top-1 prediction and the other 4, verifying the applicability and generality of our method in predicting the right label for each location. Note due to limited space, only a subset of results is presented here. Moreover, as we can see in Fig. 3 there are a few examples with the incorrect top-1 prediction (false positive). However this is not the case when the results are aggregated by crowd-sourcing as it is shown in Tables II. These results show the expressivity of our method in aggregating with crowd-sensing (crowd-prediction) to improve the prediction accuracy.

System Performance: In Section VI we discussed how we embed the deep models on mobile devices to address privacy and energy consumption concerns. We benchmark the latency of running our deep neural model when performing the prediction on a Galaxy S7 smartphones as shown in Fig. **??**. This

figure shows the latency of the model at runtime with layer-wise breakdown. As we can see the latency of the layers on are not distributed uniformly which motivates us to develop more interesting deployment schema for future work. Moreover, as we see the end-to-end latency for this model is more than 420 ms for processing of 1 image (batch size=1). This results suggest that without any further improvement, the models may not be fast enough to be deployed on the smartphones for real-time applications e.g., location-based services. However, they are more suitable for applications like location labeling that are essentially an offline process. We leave further improvement of the models for real-time application for our future work.

IX. Conclusion and Future work

This paper presents Deep-Crowd-Label, a novel system to semantically label user's location. Deep-Crowd-Label is a crowd-assisted system that uses crowd-sourcing in both training and inference time. It builds deep convolutional neural models using crowd-sensed images to infer the context (label) of indoor locations. It features domain adaptation and model extension via transfer learning to efficiently build deep models for image labeling. By fully exploiting the pre-trained models and available datasets, Deep-Crowd-Label builds ensemble of models to increase the robustness and improve the accuracy of prediction. Moreover, Deep-Crowd-Label aggregates individual predictions done by the crowd-workers visiting same location to infer the contextual label of that location. In addition, to preserve privacy and energy efficiency we deploy our deep model on the crowd-workers smartphones and provide the layer-wise benchmark of the models latency running on the phone. The prototype system and the preliminary experiments on 26 different indoor locations show the high accuracy of the model and demonstrates the generality and robustness of the underlying approach. Future plans include extending the model to more diverse types of locations as well as improving the on-device performance. In addition we plan to merge our ensemble of models into one unified deep neural network by exploiting the shared part of the models.

References

[1] Mit scene cnns. http://places.csail.mit.edu.

[2] Quantization of deep neural models. https://petewarden.com/2016/05/03/how-to-quantize-neural-networks-with-tensorflow.

[3] Martin Azizyan, Ionut Constandache, and Romit Roy Choudhury. Surroundsense: mobile phone localization via ambience fingerprinting. In *Proceedings of the 15th annual international conference on Mobile computing and networking*, pages 261–272. ACM, 2009.

[4] CM Bishop. Bishop pattern recognition and machine learning, 2001.

[5] Yohan Chon, Yunjong Kim, and Hojung Cha. Autonomous place naming system using opportunistic crowdsensing and knowledge from crowdsourcing. In *Information Processing in Sensor Networks (IPSN), 2013 ACM/IEEE International Conference on*. IEEE, 2013.

[6] Yohan Chon, Nicholas D Lane, Fan Li, Hojung Cha, and Feng Zhao. Automatically characterizing places with opportunistic crowdsensing using smartphones. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*. ACM, 2012.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[8] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014.

[9] Justin Johnson, Andrej Karpathy, and Li Fei-Fei. Densecap: Fully convolutional localization networks for dense captioning. *arXiv preprint arXiv:1511.07571*, 2015.

[10] Donnie H Kim, Younghun Kim, Deborah Estrin, and Mani B Srivastava. Sensloc: sensing everyday places and paths using less energy. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, pages 43–56. ACM, 2010.

[11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 2012.

[12] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 2015.

[13] Li-Jia Li, Richard Socher, and Li Fei-Fei. Towards total scene understanding: Classification, annotation and segmentation in an automatic framework. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2036–2043. IEEE, 2009.

[14] Rufeng Meng, Sheng Shen, Romit Roy Choudhury, and Srihari Nelakuditi. Matching physical sites with web sites for semantic localization. In *The 2nd workshop on Workshop on Physical Analytics*. ACM, 2015.

[15] M. M. Moazzami, D. E. Phillips, R. Tan, and G. Xing. Orbit: A platform for smartphone-based data-intensive sensing applications. *IEEE Transactions on Mobile Computing*, PP(99):1–1, 2016.

[16] Mohammad-Mahdi Moazzami, Dennis E Phillips, Rui Tan, and Guoliang Xing. Orbit: a smartphone-based platform for data-intensive embedded sensing applications. In *Proceedings of the 14th International Conference on Information Processing in Sensor Networks*. ACM, 2015.

[17] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning*, 2010.

[18] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 2010.

[19] Ling Pei, Robert Guinness, Ruizhi Chen, Jingbin Liu, Heidi Kuusniemi, Yuwei Chen, Liang Chen, and Jyrki Kaistinen. Human behavior cognition using smartphone sensors. *Sensors*, 13(2):1402–1424, 2013.

[20] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 2009.

[21] Piotr Sapiezynski, Arkadiusz Stopczynski, Radu Gatej, and Sune Lehmann. Tracking human mobility using wifi signals. *PloS one*, 2015.

[22] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014.

[23] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[24] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[25] Véase también SCHUM. The evidential foundations of probabilistic reasoning.

[26] David Kofoed Wind, Piotr Sapiezynski, Magdalena Anna Furman, and Sune Lehmann. Inferring stop-locations from wifi. *PloS one*, 2016.

[27] Amir Roshan Zamir, Afshin Dehghan, and Mubarak Shah. Visual business recognition: a multimodal approach. In *ACM Multimedia*. Citeseer, 2013.

[28] Xu Zhang, Felix Xinnan Yu, Shih-Fu Chang, and Shengjin Wang. Deep transfer network: Unsupervised domain adaptation. *arXiv preprint arXiv:1503.00591*, 2015.

[29] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *Advances in neural information processing systems*, 2014.