Landmark Recognition using HD-CNN

Devanshi Patel
(504945601)
Dept. of Computer Science
University of California, Los Angeles
devanshipatel@cs.ucla.edu

Satya Vasanth Reddy
(405029459)
Dept. of Computer Science
University of California, Los Angeles
satyavasanth@ucla.edu

Shreya Sunkam Ramaprasad
(204946268)
Dept. of Computer Science
University of California, Los Angeles
shreya.sr@cs.ucla.edu

***Abstract*—In image classification problems, most of the existing classifiers assume a flat architecture without giving merit to predicted labels that are similar to actual labels. These classifiers don't take into account the visual similarity and differences between images. In this project, we implement a Hierarchical Deep Convolutional Neural architecture to classify the landmarks using a hierarchical model approach. We provide a detailed description of the implementation and analysis of the results and performance of the model. We also briefly discuss the challenges faced and the causes.**

Keywords—deep learning, CNN, classification, image recognition, vision

I. INTRODUCTION

Many a times, it happens that we might forget name of the monument, we once visited. In such cases, a technology to classify the images can be of great help. It can facilitate prediction of landmark labels directly from the raw image pixels. An additional advantage is that people will be able to organize their photos in a better and organized manner. In future, this can also be applied to travel guide recommendation applications as well as for geolocation visualization.

In this report, we try to solve the Landmark Recognition Challenge by Google hosted on Kaggle. The idea is to build a model that recognizes the correct landmark in a dataset of challenging test images. A major challenge for this application is to obtain a large annotated dataset. It is extremely difficult to add annotations manually for all images. So, to build the model, we utilize the annotated data provided by them. This data has been constructed by clustering visually similar photos and matching them based on local features.

For image recognition and classification, deep CNN is the state-of-the-art approach for training the model. The reason for high popularity of CNN is because it takes advantage of local spatial coherence in the input images. Moreover, they get trained using fewer weights compared to other regular neural nets. However, the issue with normal deep CNNs is that they do not scale well with the increase in number of classes. This indicates that they directly cannot work for our dataset which consists of more than 15k classes. In our huge dataset, some categories are easier to classify than others. If we take an example from CIFAR dataset, it is easy to tell the difference between an apple and a car. But it is hard to distinguish a car from a bus. Here, we can say that car and bus belongs to the same higher level category of vehicles while apple belongs to the category of fruits. If we leverage this categorization, then we will be able to scale our model well with the increase in number of classes.

We implement hierarchical deep CNN which consists of a two level hierarchy. These hierarchy can either be learnt from the data or it can be predefined. Once the hierarchies have been established, we train our model to first classify the incoming images into a coarse category and then direct the flow to the corresponding fine category classifier. The HD-CNN model is modular and it can built on top of any existing building block CNN. Thus, the model consists of a coarse to fine classification followed by probabilistic integration of predictions from both classifiers.

There are some related works that makes use of category hierarchical structures [9]. One of the most common strategies for classification with a large number of classes using linear classifier to build a hierarchy or taxonomy of classifiers so that the number of classifiers evaluated given a testing image scales sub-linearly in the number of classes. There are approaches that predefine the hierarchy or learn the hierarchy. In the model implemented we learn the hierarchy

II. Background and Related Work

First, In this section we provide background knowledge required to understand the layers widely used in the implementation of the Hierarchical Deep CNN model. We also briefly talk about the existing work in this domain.

A regular neural network receives an input and transforms it through a series of hidden layers of neurons. Each neuron in hidden layer is fully connected to all neurons in the previous layer. Neurons within a layer function completely independently and do not share any connections. Regular Neural Nets don't scale well to full images: For example, an image of size, e.g. 200x200x3, would lead to neurons that have 200*200*3 = 120,000 weights in the fully connected layer.

On the other hand, convolutional neural networks take advantage of the fact that the input consists of images and constrain the architecture in a more sensible way. In particular, unlike a regular Neural Network, the layers of a

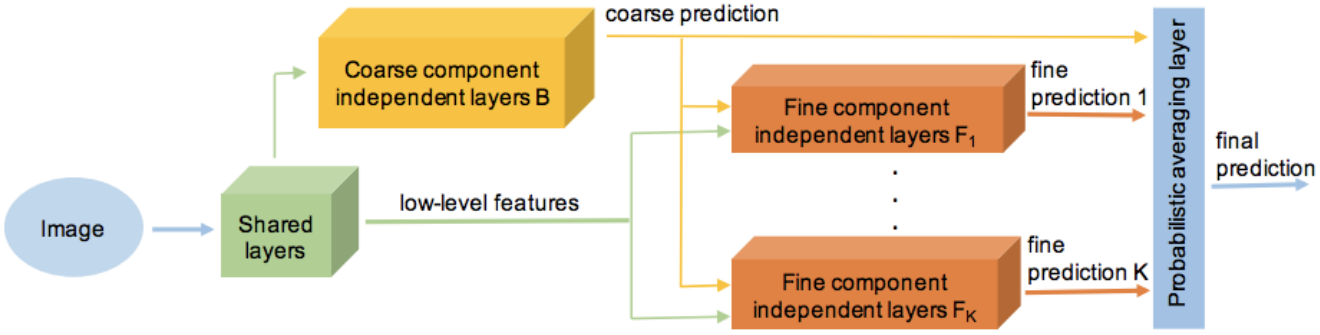


Fig.1 Architecture of Hierarchical Deep CNN

ConvNet have neurons arranged in 3 dimensions: width, height, depth. Every layer of a ConvNet transforms one volume of activations to another through a differentiable function. We give a brief overview of layers below:

Convolutional Layer: It consists of neurons that connect to subregions of the input images or the outputs of the layer before it. A convolutional layer learns the features localized by these regions while scanning through an image. Each of the layers computes a dot product between the weights and a small region they are connected to in the input volume. A set of weights that are applied to a region in the image is called a filter. The filter moves along the input image vertically and horizontally, repeating the same computation for each region. The step size with which it moves is called a stride.

ReLU Layer: This layer applies a nonlinear activation function such as the max(0,x) (thresholding at zero) to the output of the convolutional layer.

Max and Average-Pooling Layers: The max and average-pooling layers follow the convolutional layers for down-sampling, hence, reducing the number of connections to the following layers (usually a fully connected layer). They help in reducing overfitting. A max-pooling layer returns the maximum values of rectangular regions of its input. The average-pooling layer outputs the average values of rectangular regions of its input.

Dropout Layer: A dropout layer randomly sets the layer's input elements to zero with a given probability.

Fully Connected Layer: The convolutional (and down-sampling) layers are followed by one or more fully connected layers. This layer combines all of the features (local information) learned by the previous layers across the image to identify the larger patterns. For classification problems, the last fully connected layer combines the features to classify the images.

Output Layer: For classification problems, a softmax layer and then a classification layer must follow the final fully connected layer.

CNN models have been widely used for tasks related to computer vision and image parsing for a long time.

However, not a lot of research and experimentation has been done in regard to hierarchical architecture that utilizes an existing CNN as a building block. We did come across a paper that tried using category hierarchy but their main agenda was to transfer knowledge among classes for improving accuracy. Additionally, some of the works that we found, used predefined hierarchy, but in our case that is not possible. So, we use spectral clustering to learn the hierarchy.

III. HIERARCHICAL DEEP CNN (HD-CNN)

In large scale supervised visual recognition tasks with a large number of categories, the visual separability of object categories could be highly uneven. Some categories are much harder to distinguish than others. Hence instead of a flat N-way structure for the classifiers it would make sense to have the classifiers arranged in a hierarchical fashion. The HD-CNN model consists of a coarse classifier which learns the coarse categories of the image in the dataset. A set of fine category classifiers learn to classify the categories within each coarse category classifier.

HD-CNNs are modular and are built upon a building block CNN. It is more scalable than a CNN in the sense that the individual fine classifiers can be independently trained hence reducing the overall time during training. In addition, when a test image is input to the network, only certain portions of the network are conditionally executed. This gives us performance gains during the testing phase.

In the fig.1, we can see the architecture of the model. It mainly comprises of four parts:

Shared layers : Layers shared by both the fine and coarse category classifier

Coarse category classifiers : Single component B to handle coarse categories

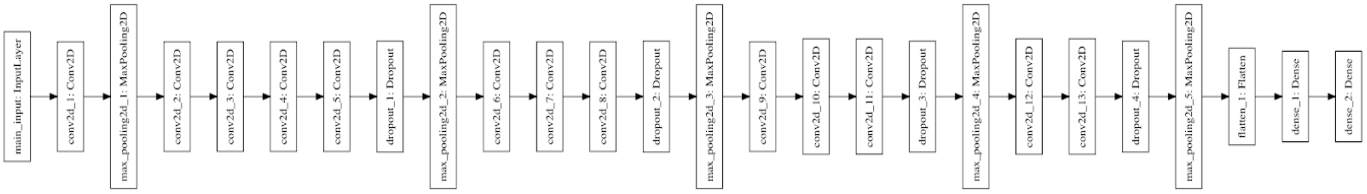


Fig.2 Different layers within shared layers component

Fine category classifiers: Multiple components(one for each coarse category) for fine classification

Probabilistic averaging layer : To obtain the final prediction from coarse and fine predictions

A. *Shared Layers*

As seen from the architecture, the shared layers receive raw image pixels as input and extract low level features such as edges and corners. Since the low level features are useful for both coarse and fine category classification, the shared layers which identify low level features are shared between coarse and fine classifiers. The fig. 2 shows the architecture of the shared layers.

The major advantage of having shared layers is that it helps in reducing the computational complexity of the model as well as its memory footprint. Additionally, it helps in reducing the number of HD-CNN parameters.

B. *Coarse category classifier*

The coarse category classifier predicts the coarse category for an image. In other words it reduces fine predictions into coarse using a mapping P : [1, C] → [1, K] where C is the number of fine categories and K is the number of coarse categories. The coarse classifier reuses configuration of the rear layers of building block CNN.

The predictions of the coarse category classifier are used as weights while combining the predictions of the fine category classifier in order to do probabilistic averaging. Also, these predictions are thresholded and only the fine category classifiers whose corresponding coarse category components are sufficiently high are executed.

C. *Fine category classifier*

The fine category classifier is responsible for making fine predictions and it helps classifying images that are similar in appearance. There is one fine classifier dedicated to each coarse category. Each such classifier is trained to classify a partial set of categories. The layer configurations are mostly copied from the building block CNN. The only difference is in the number of filters for the final layer which is equal to number of

categories within that classifier. While making predictions, the probabilities of other fine categories are implicitly set to zero.

D. Probabilistic averaging layer

The probabilistic averaging layer receives both the coarse and fine predictions and makes a final prediction based on a weighted average.

$$p(y_i = j|\mathbf{x}_i) = \frac{\sum_{k=1}^{K} B_{ik} p_k(y_i = j|\mathbf{x}_i)}{\sum_{k=1}^{K} B_{ik}} \qquad (1)$$

Here Bik is the probability of coarse category k for image xi predicted by the coarse category component B. pk(yi = j|xi) is the fine category prediction made by the fine category component Fk.

E. Spectral Clustering for learning Coarse categories

In certain datasets, for example the cifar dataset, the images are labelled with coarse categories as well as fine categories. However, it was not the case for our dataset and we learnt the hierarchy from the validation dataset which was a held-out set of images with balanced class distribution. The coarse categories are learnt by spectral clustering so that confusing fine categories can be grouped into a single coarse category.

In clustering of data, spectral clustering techniques make use of the spectrum (eigenvalues) of the similarity matrix of the image classification results to perform dimensionality reduction before clustering in fewer dimensions. Using the confusion matrix as input, we compute the distance matrix using the equation below:

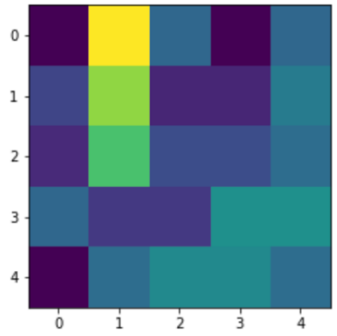
$$\mathbf{D} = \frac{1}{2}[(I - \mathbf{F}) + (I - \mathbf{F})^T] \qquad (2)$$

When spectral clustering is performed on D, we get K clusters which represent coarse categories. These clusters can be either disjoint or overlapping. However, it is highly preferred to have overlapping clusters. The problem with disjoint clusters is that they rely highly on correct prediction of coarse classifier. Based on selection of coarse classifier only, one of the fine categories under it will be selected. Thus, if an image is routed to incorrect coarse category, there is no way to recover correct class label as we explicitly set the probabilities of other fine classes to zero.

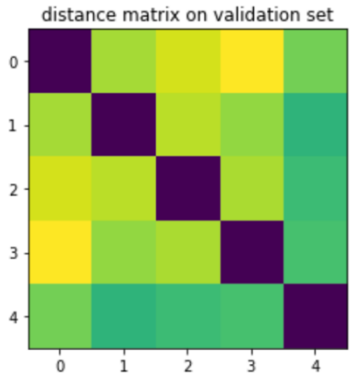
Having overlapping clusters reduces the dependence on coarse classifier. To achieve this, we add a fine category to more than one coarse category depending upon the likelihood of that label getting misclassified into that coarse category. This done with the help of a likelihood function that selects the images that are highly likely to get confused among some coarse categories.

```
[[0.          0.6         0.2         0.          0.2        ]
 [0.125       0.5         0.0625      0.0625      0.25       ]
 [0.07142857  0.42857143  0.14285714  0.14285714  0.21428571]
 [0.2         0.1         0.1         0.3         0.3        ]
 [0.          0.21428571  0.28571429  0.28571429  0.21428571]]
```

(a) Prediction probabilities learnt from a basic building block neural network



(b) Confusion matrix



(c) Distance Matrix

```
cluster 0 size 4
0,
1,
2,
4,

cluster 1 size 1
3,

[[0, 1, 2, 4], [3]]
```

(d) Cluster membership

Fig 3: Results of Spectral Clustering

Figure 3(a) shows the prediction probabilities of the held out dataset which are output from the building block CNN. Figure 3(b) represents the confusion matrix for the dataset obtained from the probabilities and Figure 3(c), the corresponding distance matrix. Figure 3(d) represents the cluster membership of each of the fine categories.

IV. Experiments and Results

The dataset of our project is a subset of an original dataset consisting of nearly 15000 classes.

The original dataset was constructed by clustering photos based on their visual similarity and geolocation. Certain features belonging to the training images were used to establish a matching between training images that belong to the same cluster. Since the dataset itself was constructed by clustering, the dataset is not very clean and it is possible that some images in the class are not pertaining to that label. When clustered on the dataset this way we get multiple clusters for each landmark which correspond to the different parts and different views of the landmark. Hence the dataset is characterized by high intra-class variance. The number of images per class is highly uneven: ranging from 10000 per class to a few 100s.

For our experimentation purposes we used a subset of the original dataset. We randomly sampled a subset of classes using a script to generate datasets of different sizes: 20, 50 and 100 classes and with different number of images for each class.

Results

For all of our experiments, we used Nvidia's GeForce 1080 GPU. We set up a docker environment with the Keras module with tensorflow-gpu backend and required libraries. The time taken for training ranges from 1 hour (20 classes) to 5-6 hours (100 classes).

TABLE 1: Summary of results

No. of classes	***Random Guess***	***Single Classifier***	***HD-CNN***	
			Coarse Classifier	***Fine Classifier***
20	*0.0493*	*0.29*	*0.31*	*0.62*
100	*0.009*	*0.42*	*0.20*	*0.46*

The accuracy of the HD-CNN relies highly on results on clustering and coarse classifier. We could achieve higher accuracy if we trained on more images per class. Due to computational constraints we had to limit the number of images per class to 1000.

There is a stark difference in the accuracy of the single classifier and HD-CNN. The single classifier is an N-way classifier and does not take into consideration the fact that certain categories are more easy to distinguish than others.

The accuracy of the coarse category classifier is critical to achieving a good overall accuracy. This is because only the fine categories corresponding to the coarse categories of the image are executed. The probability of the fine categories belong to all the other coarse categories are implicitly set to 0.

The Coarse classifier accuracy decreases with the increase in the number of classes. Hence the complexity of the coarse classifier is to be increased with the number of classes. The fine classifier generally has a good accuracy as it has to deal with a small number of classes.

We also evaluated using Global Average Precision (GAP) at k, where k=1. This metric is also known as micro Average Precision (microAP), For each query image, we predict one landmark label and a corresponding confidence score. The evaluation treats each prediction as an individual data point in a long list of predictions (sorted in descending order by confidence scores), and computes the Average Precision based on this list. For N predictions (label/confidence pairs) sorted in descending order by their confidence scores, then the Global Average Precision is computed.

V. CONCLUSION

The HD-CNN model trained for the purpose of Landmark recognition makes use of the hierarchical nature of the dataset. The hierarchy is learnt from the dataset and independent classifiers are trained for the categories in each coarse level hierarchy. In doing so, we are able to to address the issue of large number of classes as well as classes with different amounts of distinguishability. The key to achieving higher accuracy is to train on large number of images per class. The key challenges faced are that considerable amount of computational resources are required during the training phase.

References

[1] Yan, Zhicheng, et al. "HD-CNN: hierarchical deep convolutional neural networks for large scale visual recognition." Proceedings of the IEEE international conference on computer vision. 2015.

[2] Zhu, Lei, et al. "Landmark classification with hierarchical multi-modal exemplar feature." IEEE Transactions on Multimedia17.7 (2015): 981-993.

[3] Amato, Giuseppe, Fabrizio Falchi, and Claudio Gennaro. "Fast image classification for monument recognition." Journal on Computing and Cultural Heritage (JOCCH) 8.4 (2015): 18.

[4] Zheng, Yan-Tao, et al. "Tour the world: building a web-scale landmark recognition engine." Computer vision and pattern recognition, 2009. CVPR 2009. IEEE conference on. IEEE, 2009.

[5] Y LeCun, et al “Handwritten digit recognition with a back-propagation network”, Advances in neural information processing systems, 1990

[6] Von Luxburg, “A tutorial on spectral clustering” U. Stat Comput (2007) 17: 395. https://doi.org/10.1007/s11222-007-9033-z

[7] Christian Szegedy et. al, “Going Deeper with Convolutions” https://arxiv.org/abs/1409.4842 CVPR 2015.

[8] Karen Simonyan et al, “Very Deep Convolutional Networks for Large-Scale Image Recognition”, arXiv:1409.1556v6 , ICLR .

[9] A.-M. Tousch, S. Herbin, and J.-Y. Audibert. Semantic hierarchies for image annotation: A survey. Pattern Recognition,2012.

[10] J. Deng, N. Ding, Y. Jia, A. Frome, K. Murphy, S. Bengio, Y. Li, H. Neven, and H. Adam. Large-scale object classification using label relation graphs. In ECCV. 2014.

[11]] H. Bannour and C. Hudelot. Hierarchical image annotation using semantic hierarchies. In Proceedings of the 21st ACM International Conference on Information and Knowledge Management, 2012.