

CIFAR 100 ATNN Project

Popescu Andrei-George

January 16, 2024

The objective studied within this image classification project was to test the behavior of an architecture that uses several convolutional neural networks for object recognition with the aim of dividing them into coarse classes, and then reaching the fine grained ones. The aim was to reach some results that would show if the use of such an architecture, which risks using much more memory, brings advantages in terms of the time required for training and the accuracy obtained.

1 Introduction

In this competition, we had to classify images into 100 classes. The dataset provided for this problem consists of 50,000 images for training and 10,000 for testing. In addition to the 100 classes of these 32x32 pixel color images, we are also offered 20 rough classes in which these images are divided. This information served as a starting point for the architecture we are going to discuss.

The solution designed to solve this problem uses several convolutional neural network models (which can differ from each other depending on the preferences related to the memory required for the architecture). A model is trained to divide the images into coarse categories, in the case of the structure used for the competition, 3 classes were used, and 3 other models to predict the exact label for each element in the coarse categories.

In the competition, the score obtained was 0.47305. In order to make time more efficient, models whose training time should not be too long and a reduced number of epochs were used. From the point of view of the parameters used to train this architecture, we must start with the type of networks used. In order to train the model that divides into the 3 coarse classes, a ResNet-34 network was used that covered 35 epochs in 20 minutes, reaching an accuracy of 83%, and for the 3 models that did the detailed classification of pictures, a PreResNet type network was used that covered 40 epochs in 37 minutes. From the point of view of the memory occupied by this architecture, 90.5 Mb were needed to store the models.












2 State of The Art

The main article that served as a source of inspiration was Deep Residual Learning for Image Recognition [1]. From this article I took over the structure of the ResNet networks that I implemented with the aim of applying them in the architecture. I also relied on the statistics presented in this article to choose early what types of ResNet networks to test.

Another article that served as a source of inspiration was HD-CNN: Hierarchical Deep Convolutional Neural Network for Image Classification [2]. Within it is presented a complex hierarchical structure that uses several models and combining the results obtained from them to obtain better accuracy at the cost of a larger memory space occupied by the used models.

There are many models that have achieved impressive scores over the years without using extra data for training. The architecture that we implemented, tested by training four models of different types over a short period of time (less than an hour for all models together), it managed to surpass the results obtained by CNN39 and rank below AlexNet.

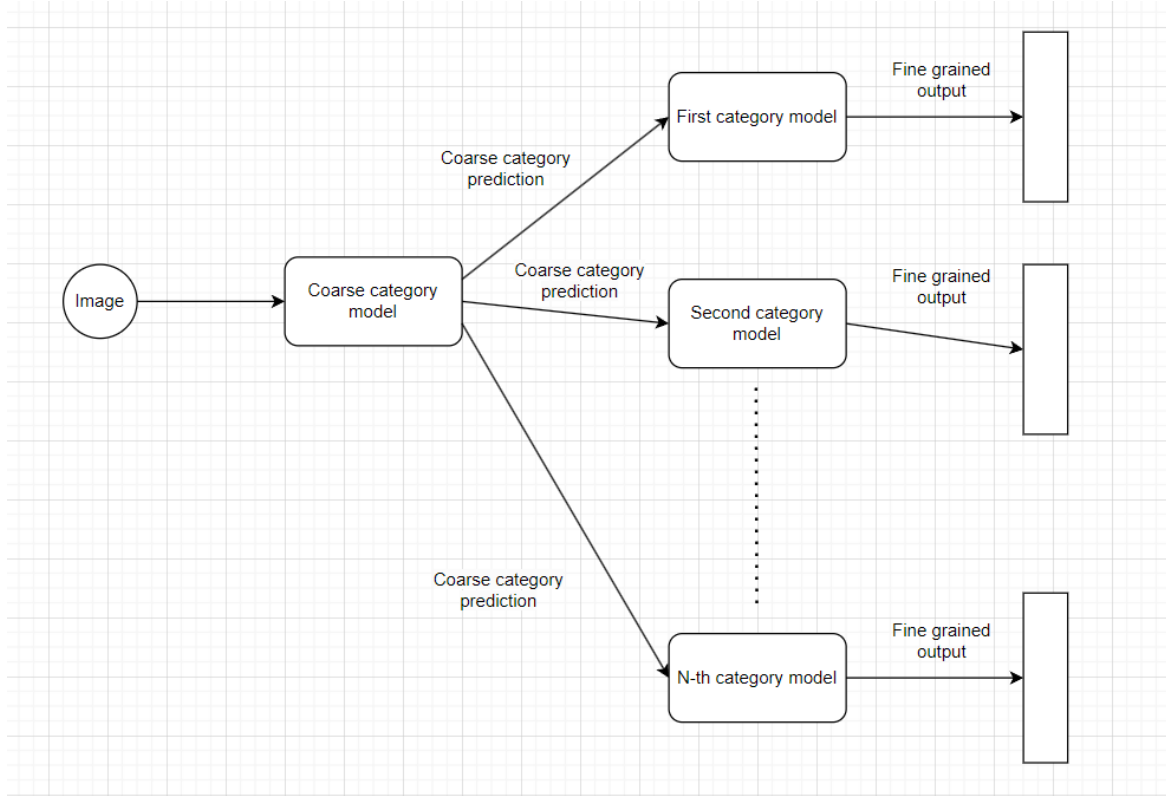
Figure 1: Rank in relation with State of the art models

29	RReLU	59.8	×	Empirical Evaluation of Rectified Activations in Convolutional Network	 	2015
30	Stochastic Pooling	57.5	×	Stochastic Pooling for Regularization of Deep Convolutional Neural Networks	 	2013
31	Sign-symmetry	48.75	×	How Important is Weight Symmetry in Backpropagation?	 	2015
32	AlexNet (DFA)	48.03	×	Learning the Connections in Direct Feedback Alignment		2020
33	CNN39	42.64	×	Sharpness-Aware Minimization for Efficiently Improving Generalization	 	2020
34	CNN36	36.07	×	Sharpness-Aware Minimization for Efficiently Improving Generalization	 	2020

3 Detailed architecture

The idea behind the proposed architecture is that by dividing a data set with many classes, which were fundamentally different from each other, into several intermediate classes, which represent a rough division of them according to certain specific features, could bring an extra accuracy in training. More specifically, each model used within these coarse classes can focus more on learning detailed features if all the data in that class are as similar as possible to each other (for example, a model that has to learn to classify only mammals, unlike one that can specifically classify both mammals and buildings and fruits).

Figure 2: Architecture



The advantage of such a structure is the fact that different types of convolutional neural network can be used depending on the performance studied for recognizing certain features that are necessary in the class in

which the training is done, but also depending on the time and memory space occupied by this model. In order to study the efficiency of using model combinations, as previously discussed, we worked with three types of convolutional neural networks: PreResNet, ResNet-18 and ResNet-34.

Having the bases necessary for the use of the discussed architecture, we can move on to how it was built in the concrete cases approached by me. We trained a ResNet-18 model to divide the data sets (training and testing) into three gross categories, more precisely, one category addressed to animals (humans, animals, fish), one related to small objects (fruits, vegetables, flowers) and one that includes large objects (furniture, bridges, buildings).

Table 1: Coarse classes distribution

First category	Second category	Third category
apple, bottle, bowl, can, clock, cup, keyboard, lamp, orange, orchid, mushroom, pear, plate, poppy, rose, sunflower sweet pepper, telephone, tulip, worm	aquarium fish, baby, bear, beaver bee, beetle, boy, butterfly, camel, caterpillar, cattle, chimpanzee, cockroach, crab, crocodile, dinosaur, dolphin, elephant, flatfish, fox girl, hamster, kangaroo, leopard, lion, lizard, lobster, man mouse, otter, porcupine, possum rabbit, raccoon, ray, seal shark, shrew, skunk, snail, snake, spider, squirrel, tiger trout, turtle, whale, wolf, woman	bed, bicycle, bridge, bus, castle, chair cloud, couch, forest house, lawn mower, maple tree motorcycle, mountain, oak tree palm tree, pickup truck, pine tree, plain, road, rocket sea, skyscraper, streetcar table, tank, television tractor, train, wardrobe, willow tree

After the training of the first model, the part of dividing the data into the 3 coarse categories according to the network takes place. After this, the training of the models that will do fine grained classification begins. Since the division of data into these large categories is not perfect (it depends on the accuracy obtained in the first model), there will be data that end up in the wrong category and have no way to be classified correctly. These were used in training to insert noise into the data. Each input, which did not have a label according to the category to which it was assigned, receives a random label among those suitable for the class in which it arrived. In this way, overfitting is avoided, which was reached very quickly (approximately 30 epochs, depending on the model) if these data that ended up in the wrong category were simply ignored during training.

3.1 Benchmark Performance

From the point of view of the results obtained using different combinations of models, they can be seen in the table below.

Coarse Model	First Category Model	Second Category Model	Third Category Model	Top 1 Accuracy	Top 3 Accuracy	Epochs	Training Time	Memory Usage (MB)
ResNet-18	ResNet-34	ResNet-34	ResNet-34	42.87%	64.36%	120	37 min	293
ResNet-18	ResNet-18	ResNet-18	ResNet-18	42.82%	63.23%	120	25 min	175
ResNet-18	PreResNet	PreResNet	PreResNet	46.92%	69.98%	120	39 min	51

The differences obtained in terms of accuracy can be observed, as well as those related to the training time and space required for storage, first the modification of the models used in the structure.

3.2 Ablation Study

From the point of view of improving the obtained results, in addition to using different types of convolutional neural network, I tried to change several hyperparameters and the structure of certain components in the models (such as changing the activation functions). However, one of the most important changes I could make for testing the architecture was changing its structure. From this I restructured the coarse classes, using 4 instead of 3, thus reaching a total of 5 models to be trained. In order to compare the results with the previous structure, I used the same combinations of models for training and the same number of epochs.

Table 3: Second coarse classes distribution

First category	Second category	Third category	Fourth category
apple, bee, bottle, bowl, butterfly, can, caterpillar, clock, cockroach, cup, keyboard, lamp, orange, orchid, mushroom, pear, plate, poppy, rose, snail, snake, spider, sunflower, sweet pepper, telephone, tulip, worm	baby, bear, beaver beetle, boy, camel, cattle, chimpanzee, dinosaur, elephant, fox, girl, hamster, kangaroo, leopard, lion, lizard, man mouse, porcupine, possum rabbit, raccoon, shrew, skunk, squirrel, tiger wolf, woman	bed, bicycle, bridge, bus, castle, chair cloud, couch, forest house, lawn mower, maple tree, motorcycle, palm tree, pickup truck, mountain, oak tree, pine tree, plain, road, rocket, sea, skyscraper, tractor, train, wardrobe, streetcar, table, tank, television, willow tree	aquarium fish, crab, crocodile, dolphin, flatfish, lobster, otter, ray, seal, shark, trout, turtle, whale

The purpose of this new division into the 4 classes was to better separate the elements that are similar, especially the vitae that we had put in the same class. For this reason, I have put the aquatic animals in a new category and the terrestrial ones of small dimensions such as reptiles and bugs in the category of small objects.

Coarse Model	First Category Model	Second Category Model	Third Category Model	Fourth Category Model	Top 1 Accuracy	Top 3 Accuracy	Epochs	Training Time	Memory Usage (MB)
ResNet-18	ResNet-34	ResNet-34	ResNet-34	ResNet-34	41.85%	62.91%	120	39 min	376
ResNet-18	ResNet-18	ResNet-18	ResNet-18	ResNet-18	43.04%	66.87%	120	26 min	218
ResNet-18	PreResNet	PreResNet	PreResNet	PreResNet	44.48%	68.46%	120	40 min	54

As can be seen from the resulting data, although in certain combinations of models a better accuracy is reached and the training time does not differ considerably, the additional memory space required for one more model is not justified considering that in general the results obtained were worse than those obtained by using only 3 coarse classes. The results presented in the tables above were obtained by training on an Nvidia RTX 3080 video card.

For the case of training on the CPU, we were able to use the model with 4 coarse classes, each of them using a ResNet-18 type model to obtain a shorter training time.

Coarse Model	First Category Model	Second Category Model	Third Category Model	Fourth Category Model	Top 1 Accuracy	Top 3 Accuracy	Epochs	Training Time	Memory Usage (MB)
ResNet-18	ResNet-18	ResNet-18	ResNet-18	ResNet-18	42.09%	66.33%	85	90 min	218

4 Conclusion

To conclude, the purpose of this experiment was to test the results obtained by a multi-model architecture as well as the ways in which it can be expanded to experiment on fine-grained classification using an increasing number of models. The results obtained over short periods of time were followed, with the aim of creating structures that sacrifice memory (depending on the types of nonvolutional neural network chosen) in order to gain time and obtain decent results in a small number of training epochs.

References

- [1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [2] Zhicheng Yan, Hao Zhang, Robinson Piramuthu, Vignesh Jagadeesh, Dennis DeCoste, Wei Di, and Yizhou Yu. Hd-cnn: hierarchical deep convolutional neural networks for large scale visual recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 2740–2748, 2015.