Tiny ImageNet Challenge

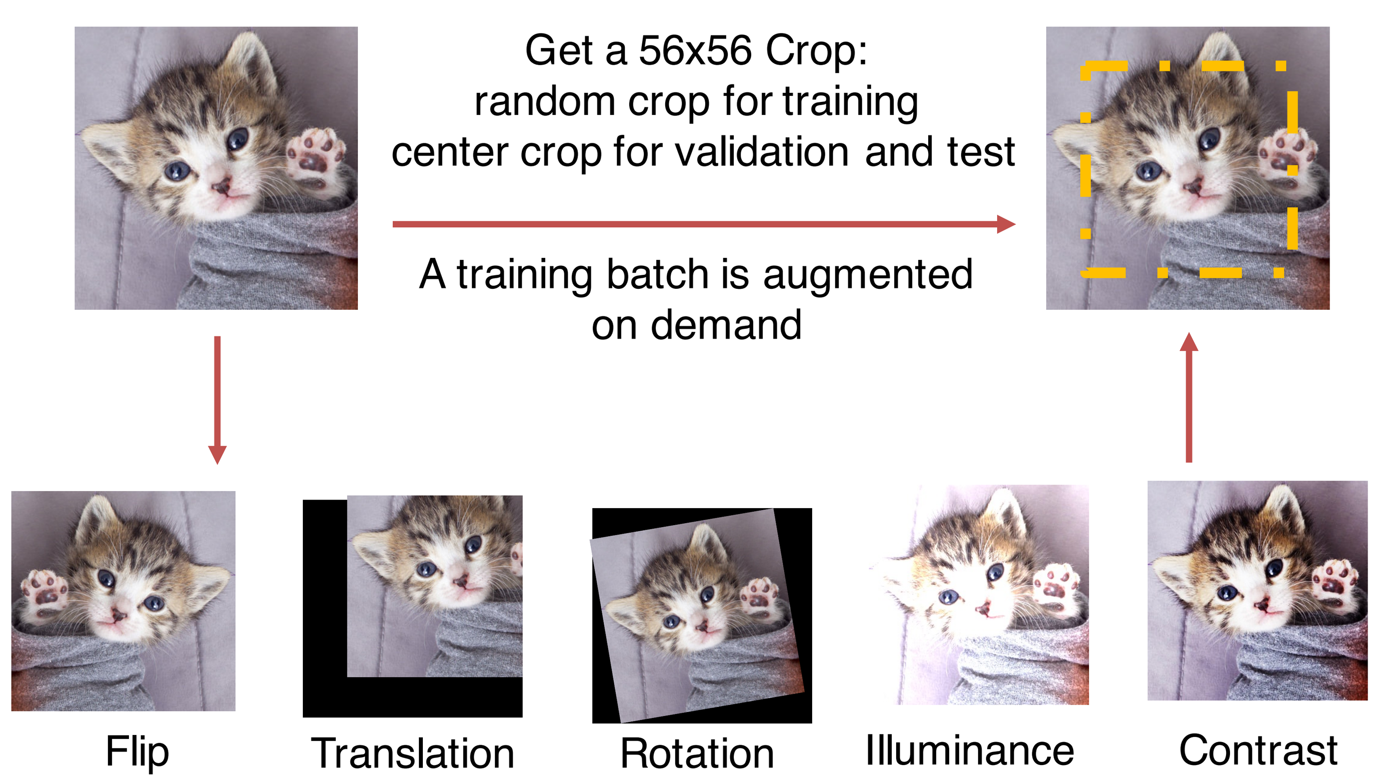
Qixiang Zhang, Jiayu Wu, Guoxi Xu
Stanford University

Problem

The goal of Tiny ImageNet Challenge is to do the image classification problem well. There are 200 image classes in total. We trained ResNet-like deep convolutional neural networks from scratch on the augmented dataset. The challenges of this project include identifying the optimal network structure and preventing overfitting. Our best model achieves 46.9% top-1 error on test data. Moreover, we tried object localization in our dataset using modified OverFeat-GoogLeNet algorithm.

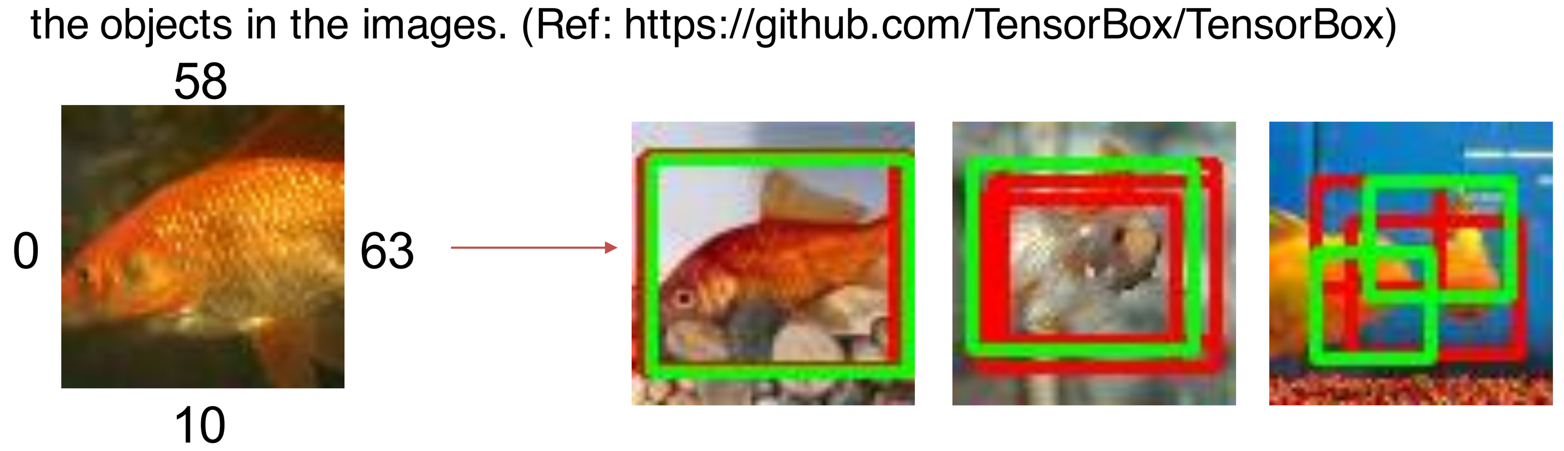
Data

Tiny ImageNet Challenge provides a training dataset of 100,000 images, a validation dataset of 10,000 images, and a test dataset of 10,000 images. All images are of size 64x64. In order to create more images for training, we use the following data augmentation pipeline:

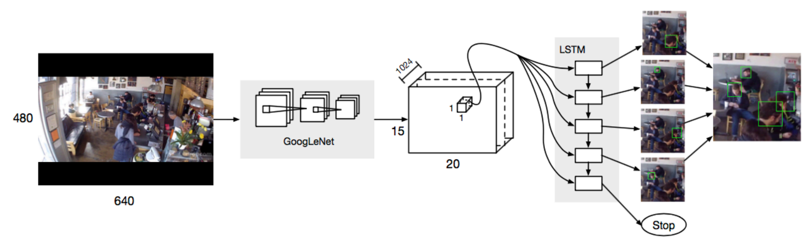


Localization

Based on Tensorbox code, we modified the OverFeat-GoogLeNet algorithm to localize the objects in the images. (Ref: https://github.com/TensorBox/TensorBox)



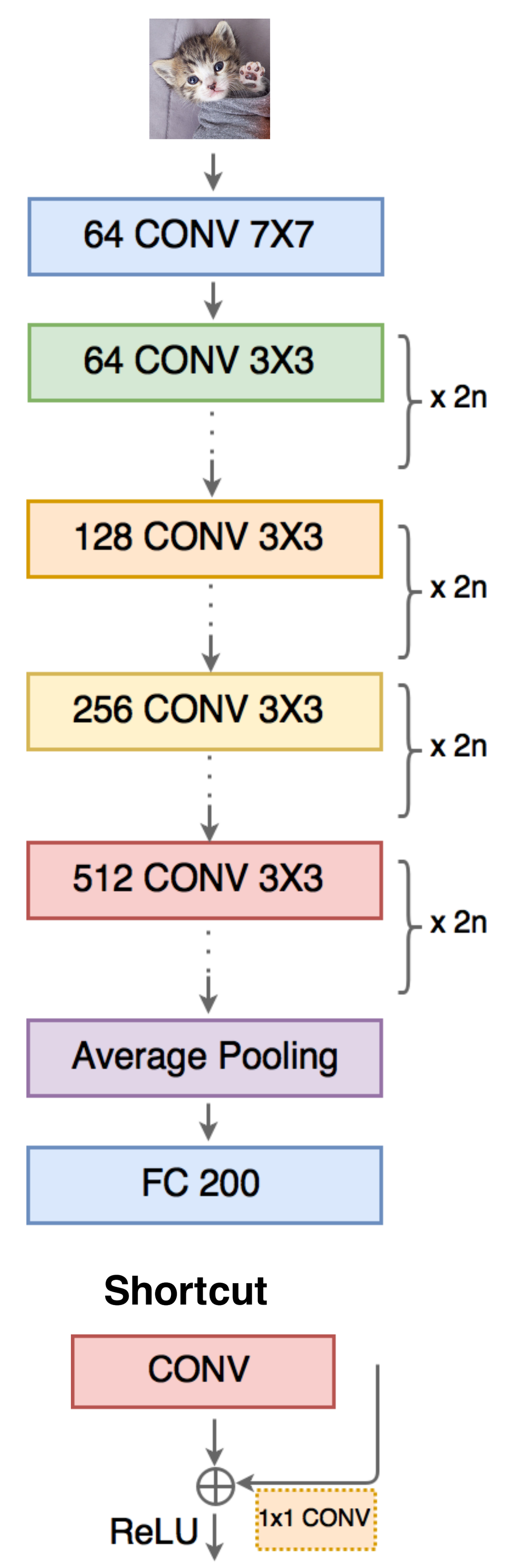
As the object localization only need to learn the feature for one kind of object, for one task we only feed one class of training images with existed boundaries of boxes in both x-axis and y-axis.(above left) The generated boxes are in above right.
The algorithm generate boxes with confidences between 0~1.



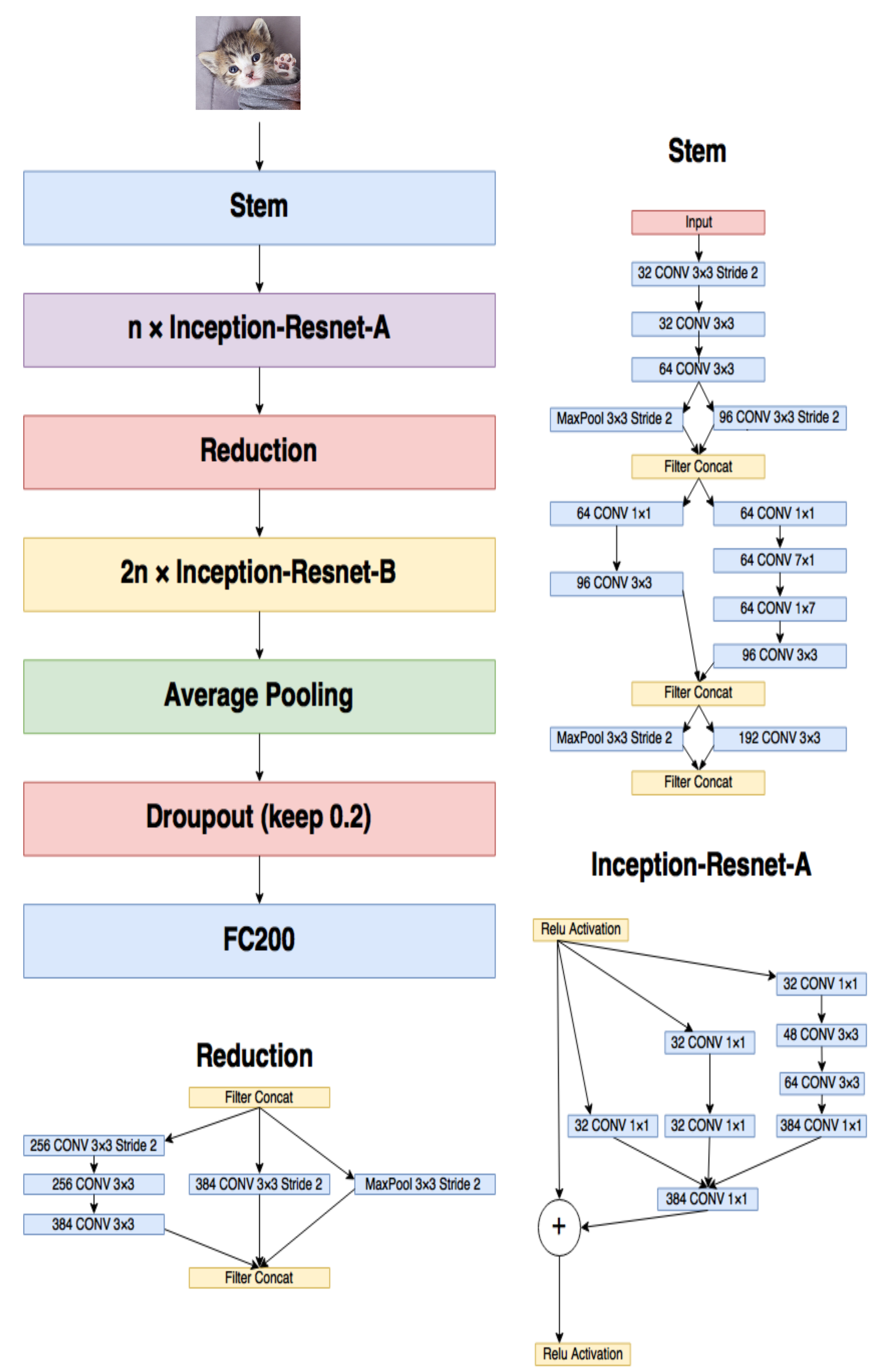
$$L(G,C,f) = \alpha \sum_{i=1}^{|G|} l_{pos}(\mathbf{b}^i_{pos}, \tilde{\mathbf{b}}^{f(i)}_{pos}) + \sum_{j=1}^{|C|} l_c(\tilde{\mathbf{b}}^j_c, y_j)$$

Models

ResNet



Inception-ResNet



❖ ResNet Details

- Spatial batch normalization is applied immediately after each convolutional layer;
- ReLU is used as the activation function;
- The first convolutional layer of each convolutional block performs a down-sampling by using a stride of 2;
- We use a stride of 1, instead of 2, at the first convolutional layer with 7x7 filters;
- The last average pooling layer averages over the entire feature map;

❖ Inception-ResNet Details

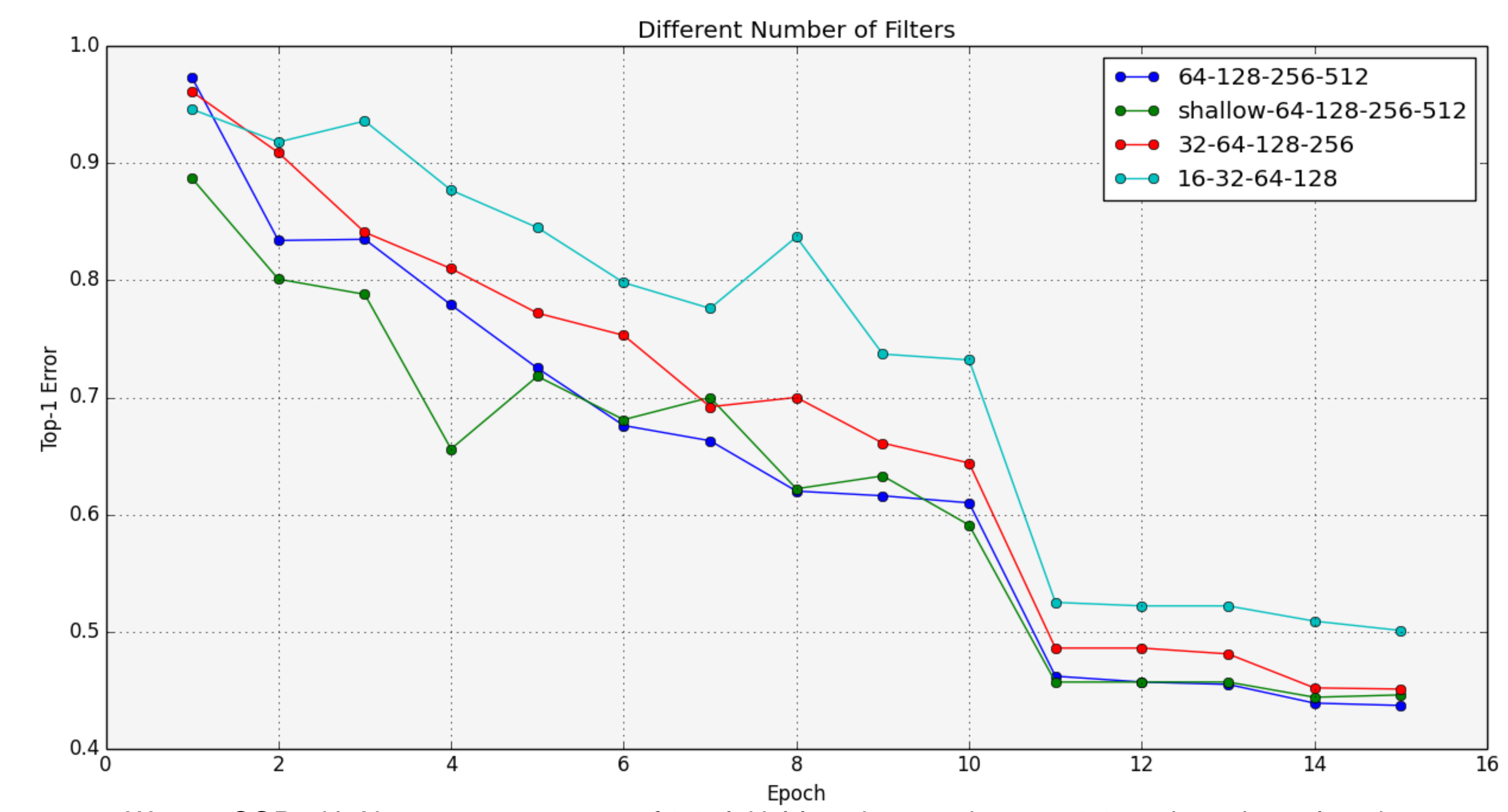
- Combine Resnet and Inception in each Inception-Resnet module.
- Down-sample spatial size in Stem module and Reduction module.
- Add batch normalization and ReLu activation to each convolutional layer.
- Add a strong dropout layer (keep 0.2) to prevent over-fitting.
- Apply a 1x7 -> 7x1 filter combination to replace a single 7x7 filter, significantly reducing number of parameters.

Results

- **Summary**

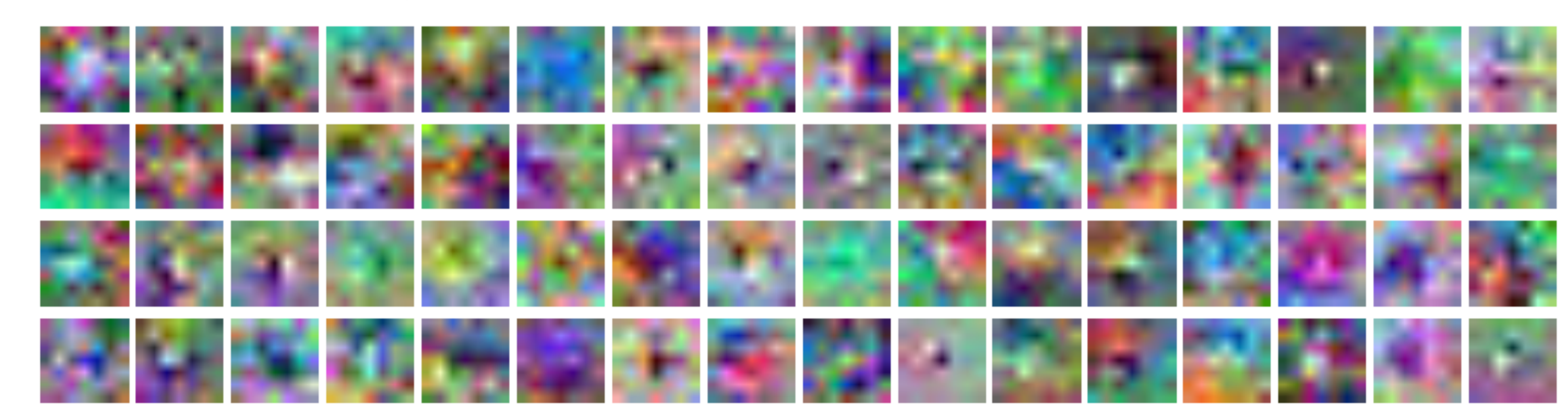
Model	# params	Top-1 Error (%)	Top-5 Error (%)
ResNet-18 [16-32-64-128]	0.73 M	50.7	24.9
ResNet-18 [32-64-128-256]	2.85 M	45.1	21.6
ResNet-18 [64-128-256-512]	11.28 M	43.5	20.3
ResNet-10 [64-128-256-512]	5.01 M	44.4	20.7
Inception-ResNet-50 [n=2]	8.3M	53.6	23.6

- **Validation Top1-Error vs. Epoch**



We use SGD with Nesterov momentum of 0.9. Initial learning rate is set to 0.1, and we decay learning rate to 0.01 after epoch 10, and to 0.001 after epoch 13.

- **First Conv Layer Visualization**



- **Analysis**
- The results of ResNet-18 [64-128-256-512] and its 10-layer shallow variant show that the benefits of going deeper for this task may be limited because the main problem is not model capacity but overfitting;
- Decaying learning rate is essential to reduce top-1 error. More specifically, the first decaying can reduce more than 15 percent error, and the second decaying can reduce around 2 percent, which indicates that global minimums may locate in those narrow valleys of the loss surface.