# FishNet Implementation for Image Classification

E4040.2021Fall.FNET.report
Candong Chen cc4766, Tracy Wang jw4167, Jinyu Wang jw4168
*Columbia University*

## Abstract

*This is an reproduced version of the FishNet: A Versatile Backbone for Image, Region, and Pixel Level Prediction trained with smaller dataset. The main goal of this model is to design backbone structure that is for pixel-level or region-level predictions in image classification tasks. In our reproduced FishNet model, we did not accomplish the high performance on the original model. Main reason resulted in overfitting with lower accuracy.*

## 1. Introduction

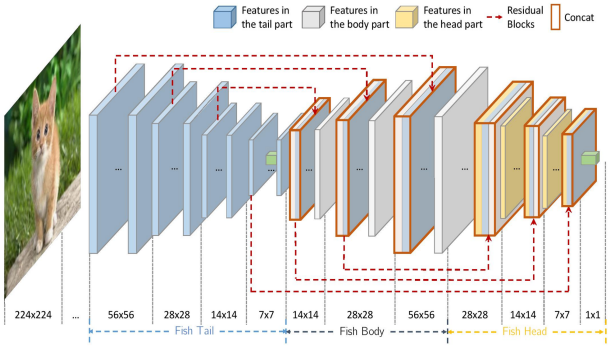
Current convolutional neural network structures are used for predicting objects on different levels. For example, for image classification, the goal is to summarise high-level semantic information of the whole image, so we usually implement convolution and then downsampling. On the other hand, tasks such as detection and segmentation require high-level semantic meaning with high spatial resolution, so we usually implement convolution and then upsampling [2], called the U-Net networks[3]. The architectures for tasks that require different resolutions are diverging. Thus, the authors of *FishNet: A Versatile Backbone for Image, Region, and Pixel Level Prediction* design a fish-like network that unifies the advantages of networks for pixel-level, region-level, and image-level tasks.

The authors claim that FishNet outperforms ResNet and DenseNet. Therefore, for this project, our goal is to reproduce the FishNet structure and evaluate how good this model actually is for image classification tasks.

One challenge we encountered was the size of the dataset that the original work was built on. Due to limitations in disk storage and device compatibility, we decided to use a simpler and smaller version of the dataset. Another challenge that we faced was that each epoch took longer and longer to train because of loss of memory when training such a complicated model with a relatively large dataset. To combat this problem, we saved trained models for each group of 5 epochs, reloaded the model, and kept training for the next group.

## 2. Summary of the Original Paper

### 2.1 Methodology of the Original Paper




The above architecture [2] summarizes the essence of the FishNet structure. The whole 'fish' is divided into three parts: tail, body, and head.

The fish tail consists of a CNN with downsampling, so as CNN goes deeper, the resolution decreases. The features in the tail go through regular convolutional connections but at the same time are transferred to the fish body.

The fish body concatenated both features from the tail and the previous stage of the body. Then they are upsampled and refined by Up-sampling & Refinement blocks (UR-block) in the fish body. Afterwards, these refined features are sent to the next stage of the body and transferred to the fish head.

Lastly, the fish head preserves and refines all the features from the fish body and the previous stage of the head. The process is done by the Downsampling & Refinement blocks (DR-block), and those refined features are sent to the next stage of the head.

In summary, each part of the fish not only goes through usual regular connections among itself, the features are also transferred to the next part. Therefore, we have different directions in our architecture chart.

The proposed FishNet structure has a few advantages. Upsampling blocks increases resolution and therefore is useful for pixel-level and region-level tasks. On the other hand, adding downsampling in the fish head will also help with image classification tasks. As a result, FishNet unifies the advantages of networks for pixel-level, region-level, and image level tasks. Additionally, it also allows direct gradient propagation from very deep layers to shallow layers because it can concatenate features of various depths to the final output. This is useful in image classification because the network

preserves features from different depths, building a diverse pool of features.

## 2.2 Key Results of the Original Paper

FishNet is created to unify the advantages of networks that solve pixel-level, region-level, and image level tasks, so FishNet is trained and tested on two types of dataset.

For image classification tasks, FishNet is evaluated on the ImageNet2012 classification dataset. Fishnet is first evaluated against a re-implemented ResNet. FishNet achieves a reduction in Top-1 error rate of around 0.7%. More importantly, the FishNet model is a more compact model with less number of parameters but ending up with high accuracy the ResNet. Afterwards, FishNet is also evaluated against DenseNet, and it achieves better classification accuracy even though it has less parameters. Furthermore, FishNet also has 30% less memory cost than DenseNet. FishNet has such accomplishments because it preserves features with more diversity and it handles back propagation in a deep network much better than DenseNet.

For object detection and instance segmentation, FishNet is evaluated on the MS COCO dataset. Using Feature Pyramid Networks with Fishnet, the model increases average precision by more than 1%, compared to using FPN with ResNet for object detection.

Overall, based on the key results, we can tell that FishNet improves performances of both image classification and object detection tasks.

# 3. Methodology (of the Students' Project)

## 3.1. Objectives and Technical Challenges

Our main objective is to reproduce FishNet structure and evaluate it on an image classification task to test if it indeed improves performance.

One challenge is that we could not reproduce the experiment on ImageNet directly because the dataset is too large for our computers. Another technical challenge is that if we run quite a few epochs at once, our training becomes slower and slower. Lastly, FishNet is a complicated neural network with a lot of hyperparameters, so it was difficult for us to tune our network to fit our dataset.

# 4. Implementation

Describe the organization of this section, then provide detailed discussion about your implementation.

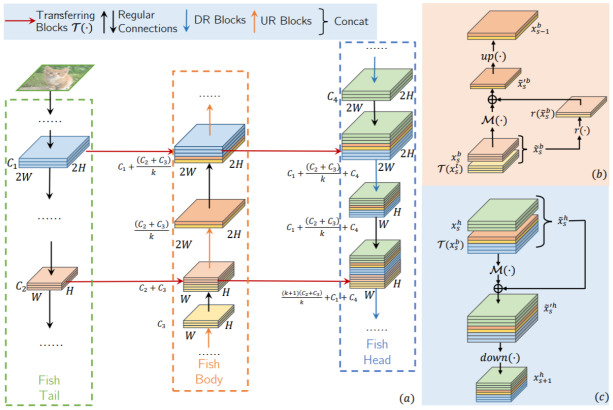
Our implementations are done in TensorFlow2.* and Keras. Each structure in our model belongs to a subclass of Keras Model (eg. Fish which contains head, body, tail; a bottleneck which blocks in Fish are based from). As discussed before, the backbone of FishNet consists of head, body, and tail. We added blocks within these sections to capture low-level information like pixel and region.

## 4.1 Data

The dataset used in the original paper is ImageNet 2012. Considering our computing resources and time cost, we choose Tiny ImageNet instead. The dataset contains 100,000 images of 200 classes (500 for each class) downsized to 64×64 colored images. Each class has 500 training images, 50 validation images, and 50 test images.

We first try to build tf.data.Dataset as the tensorflow official website shows to build our dataset for training. Unfortunately, the time cost of training is extremely expensive and tf throws OOM error (out of memory). Then we convert the original image files to TFRecord format by an official script imagenet_to_gcs from the Github repository of tensorflow to reduce memory cost and improve training efficiency. Note that the Tiny Imagenet's file structure is not the same as ImageNet 2012 so we write a bash script to make it look the same before the transformation.

## 4.1 Deep Learning Network




[2]

For training, we read and parsed TFRecords to Tensorflow Dataset of images in the resolution of 64*64 with batch size 256. To normalize pixels into the interval [0,1], we divide each pixel by 255. When compiling our model, we choose Adam as the training optimizer with the base learning rate set to 0.001. We train the network for 30 epochs in total. To save memory space and shorten training time, we save models and their corresponding weights every 5 epochs, and then we reload models and continue training in the next epoch. All the experiments

are evaluated by the validation images in the Tiny_ImageNet.

To improve overfitting, we also add ReduceLROnPlateau callback which reduces learning rate with fixed decay by monitoring validation loss during training.

After we build up the network structure, we find it always throws ValueError indicating that tf.function only supports singleton tf.Variables created on the first call. We try to call tf.config.run_functions_eagerly(True), which will make all invocations of tf.function run eagerly instead of running as a traced graph function. It solves this problem, to some extent. However, it brings another problem that the buffer memory won't be reset after each epoch so the training process becomes slower and slower. Since tf 2.0 doesn't have a reset method as tf 1.0 does, we turn to save our model each 5 epochs (one loop) and save the checkpoint of the model. Then we load the former model and overwrite the current model. After that, we train one loop again, and so forth.

### 4.2 Software Design

FishNet were built with params (eg. depth) in separate config files. These params indicate when to up/down sample layers, how many blocks are inserted, expanded dimension in the next layer, etc.

For a complete FishNet, we have 3 deep convoluted layers at front, fish tail, fish body, fish head, and output layers.

There are different kinds of layers and blocks we insert in fish to combine features from different layers. In most of the cases, blocks have the same input dimension and output dimension, it does not change layers before and after but to concatenate features from them. For example, UR-block connects and refine features from fish body and fish tail.

1. Pseudo code for each section of the implementation,
   Bottleneck: [BatchNorm, Conv2D, BatchNorm, Conv2D, BatchNorm, Conv2D, …]
   Residual blocks: [Bottleneck, …]
   Transfer blocks (is a special case residual block): [Bottleneck(in=out), …]
   Squeeze blocks: [BatchNorm, Activation(ReLU), AdaAvgPool2D, Conv2D, Activation(ReLU), Conv2D, Activation(Sigmoid)]
   Score layer: [BatchNorm, Activation(ReLU), Conv2D, BatchNorm, Activation(ReLU), Conv2D]
   Stage blocks: [Score layer, Residual blocks, Transfer blocks, Downsample layer, Upsample layer]
   Fish: [Stage blocks, …]
   Deep conv layer: [Padding, Conv2D, BatchNorm, Activation(ReLU)]
   FishNet: [Deep conv layer, Deep conv layer, Deep conv layer, MaxPool2D, Fish, Activation(Softmax)]
2. Links to code in your project github need to be included through the sections above.
3. The code in our project has been uploaded to Github and you can access it by this link.

## 5. Results

### 5.1 Project Results

In practice, we build 3 kinds of Fishnet with different complexity (order of magnitude of parameters) by giving different hyper-parameters which control the number of trainable parameters, and we use a ResNet as comparison.

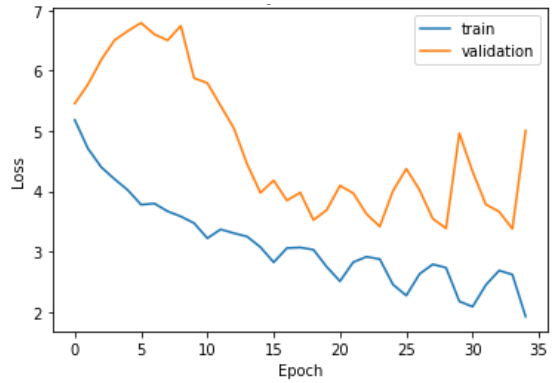
| | Params | Seconds per epoch |
|---|---|---|
| Simple FishNet | 667,584 | 64 |
| Medium FishNet | 1,034,440 | |
| Complex FishNet | 1,797,960 | 139 (average) |
| ResNet | 23,731,848 | 13 |

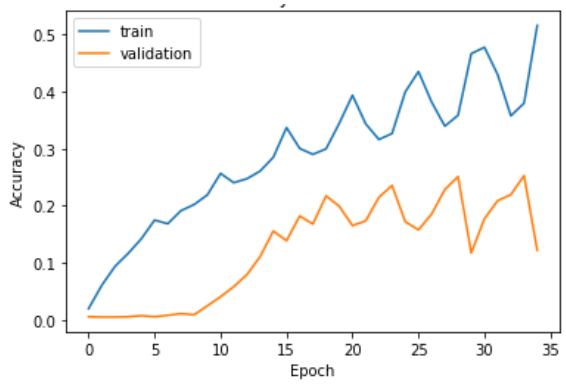
However, our result is not as good as what we have imagined. The training results of Simple FishNet and Medium FishNet are underfitting. The accuracy of Simple FishNet on training set is about 20%, while that of Medium FishNet on training set is about 30%. And their accuracy of validation set are both less than 10%. No matter how many epochs they are trained, these accuracy fluctuate little around steadily.

On the contrary, Complex FishNet and ResNet both have so great fitting ability that they overfit on the training data. The accuracy of Complex FishNet on training set is over 85% after enough training epochs. And the accuracy of ResNet on the training set even achieves 1. However, their accuracy on the validation set both are less than 20%. After tuning the number of trainable parameters carefully and trying different optimization methods, we improved the result of Complex FishNet as shown in the table below but it's performance is still very poor.

| | Training Accuracy | Validation Accuracy |
|---|---|---|
| Simple FishNet | 23.6% | 5.3% |
| Medium FishNet | 30.1% | 8.5.% |
| Complex FishNet | 50.4% | 18.8% |
| ResNet | 1 | 20.2% |

We can conclude that this FishNet general model has the capacity to solve this classification problem because we can get both underfitting and overfitting cases by just tuning the number of trainable parameters. However, we have to do more grid search to find the model with the optimal order of magnitude of parameters. Restricted by our finite computing resources and time cost, we have to end up here but our model's performance can be improved in the future if we have more time and more GPU resources.




The above two plots compare the loss and accuracy between the training set and the validation set for the complex fishnet. As discussed in the result section, training accuracy keeps rising, but validation accuracy has not made much progress since epoch 20. Clearly, our complex model is overfitting.

## 5.2 Comparison of the Results Between the Original Paper and Students' Project

We cannot find any accuracy or time cost of the original FishNet implementation in the classification problem in this paper but as the original paper said, the author trained FishNet-150 with 8 GPUs and batch size 256 and he got the Top-1 Error over 20%. The number of parameters is even over 25 million.

Like what we've mentioned before, we have to do more grid search to find the model with the optimal order of magnitude of parameters, but restricted by our finite computing resources and time cost, we have to end up here but our model's performance can be improved in the future if we have more time and more GPU resources.

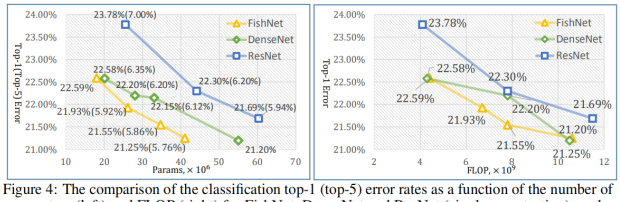



Figure 4: The comparison of the classification top-1 (top-5) error rates as a function of the number of parameters (left) and FLOP (right) for FishNet, DenseNet and ResNet (single-crop testing) on the validation set of ImageNet.

| Method | Params | Top-1 Error |
|---|---|---|
| ResNeXt-50 ($32 \times 4d$) | 25.0M | 22.2% |
| FishNeXt-150 (4d) | 26.2M | **21.5%** |

Table 1: ImageNet-1k val Top-1 error for the ResNeXt-based architectures. The 4d here for FishNeXt-150 (4d) indicates that the minimum number of channels for a single group is 4.

| Method | Params | Top-1 Error |
|---|---|---|
| Max-Pooling ($3 \times 3$, stride=2) | 26.4M | 22.51% |
| Max-Pooling ($2 \times 2$, stride=2) | 26.4M | **21.93%** |
| Avg-Pooling ($2 \times 2$, stride=2) | 26.4M | 22.86% |
| Convolution (stride=2) | 30.2M | 22.75% |

Table 2: ImageNet-1k val Top-1 error for different down-sampling methods based on FishNet-150.

## 5.3 Discussion of Insights Gained

We resulted in overfitting after some epochs of training. The overall results of our model compared to the original FishNet was not good, with much higher training accuracy comparing the validation accuracy. Although the overall results of our model is not great, we create models with config to obtain different complexity. Tuning model complexity didn't help in our case, it might be structural issue as we used the same structure for 224*224 images on training 64*64 images. With three deep convoluted layer at front, the image size were reduced to 16*16 whereas the original model still obtain an 56*56 matrix. Too many downsampling reduced the resolution and it made the model hard to translate important features (pixel-level, region-level).

## 6. Future Work

As we pointed out in the result discussion section, we concluded that FishNet has the capacity to classify the TinyImageNet dataset, but we just needed to do more grid search for hyperparameters. In the future, with adequate time and fast computing resources, we can train our model with different configurations to find the desired configuration to tune FishNet on TinyImageNet.

In our model, all activations in use are ReLU (excluding output activation such as softmax and sigmoid). It would be good to use other activations to increase model complexity.

Currently we are using Adam optimizer, in future work it is good to compare the performance of other optimizers such as RMSprop, SGD with momentum.

We encountered overfitting in training. Since we followed the original FishNet which uses blocks to obtain gradients, we never add dropout layers which is opposite of the objective. In future work, if possible we could change structures of the model to find the optimal parameters.

## 7. Conclusion

Provide the summary of this project. Repeat the statements made in the abstract, with more focus on the results. State if the objectives and goals are achieved. Emphasize and highlight the lessons learned. Point out the direction for future research and further improvements.

This project is to reproduce FishNet for ImageNet-1k. Our goal is to capture a sufficient amount of image information with our FishNet for TinyImageNet-200. However, although the original FishNet outreached 77.41% for top-1 error, our model received ??% for best performance.

## 6. Acknowledgement

Provide acknowledgements such as support, help, or assistance from online resources, TAs, colleagues.
Google Cloud Platform, Colab

## 7. References

[1] Our Github Link
[2] S. Sun, J. Pang, J. Shi, S. Yi, and W. Ouyang, “Fishnet: A versatile backbone for image, region, and pixel level prediction,” *arXiv.org*, 11-Jan-2019. [Online]. Available: https://arxiv.org/abs/1901.03495. [Accessed: 22-Dec-2021].
[3] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention, pages 234–241. Springer, 2015.

## 8. Appendix

### 8.1 Individual Student Contributions in Fractions

| | cc4766 | jw4168 | jw4167 |
|---|---|---|---|
| Last Name | Candong Chen | Jinyu Wang | Jiayi Wang |
| Fraction of (useful) total contribution | 1/3 | 1/3 | 1/3 |
| What I did 1 | FishNet | FishNet | FishNet |
| What I did 2 | data preprocess | data preprocess | data preprocess |
| What I did 3 | tuning model | tuning model | tuning model |

### 8.2 Support Material

As needed: additional diagrams, source code listing, circuit schematics, relevant datasheets, screenshots etc. Anything that may not be suitable for the main text, but would provide detailed information to be able to replicate the study.