# Pytorch Implementation of AINet

# 1.Quick Learning

For the convenience, you could go to **./Model/Img\_few\_shot\_prj.py** directlyfor quick learning of our method. I prepared a toy example in the main function there. By setting fs==’AINet’ in the class of ThreeD\_Support\_Net, you use AINet as few-shot head for the prediction.

Besides, you could also go to **./Model/Head/AINet.py** to check the class “AINet”, to learn the detail of AINet. I also prepare a toy example in the main function, for your convenience to learn the detail of our proposed method.

# 2. Data Preparation

All the query images are from [1], and support images are generated by us from 3D mesh data.

## ModelNet40:

For the query images, you could go to <https://github.com/rehg-lab/lowshot-shapebias> to download ModelNet40-LS, the query images path is ./ModelNet40-LS /renders

For the support projections generated by us, you could download it here: <https://drive.google.com/file/d/1U6VLcY-kEhQhkWI3fkA-ObJOlC88Ytfa/view?usp=share_link>

Unzip it, and put the Projection\_mesh\_darkbg folder into ./ModelNet40-LS

ModelNet40-LS is the data folder

## ShapeNet:

For the query images, you could go to <https://github.com/rehg-lab/lowshot-shapebias> to download ShapeNet55-LS, the query images path is ./ ShapeNet55-LS/renders

For the support projections generated by us, you could download it here: <https://drive.google.com/file/d/14LChsEJX4hJYx-lo9nlpMg45pfoo_zUD/view?usp=share_link>

Unzip it, and put the Projection\_mesh\_darkbg folder into ./ ShapeNet55-LS

ShapeNet55-LS is the data folder

## Toys4K:

For the query images, you could go to <https://github.com/rehg-lab/lowshot-shapebias> to download TOYS4K, the query images path is ./ TOYS4K/renders

For the support projections generated by us, you could download it here:

<https://drive.google.com/file/d/15FACStHmzJ8Q-DhBk4GSAtiaoUfvWCwM/view?usp=share_link>

unzip it, and put the Projection\_mesh\_darkbg folder into ./ TOYS4K

TOYS4K is the data folder

# 3. Pretraining

You could directly download the well-pretrained backbone(ResNet) from this link: <https://drive.google.com/file/d/1JsCcquiOFdGzOpNtQ5FQdY1pQZN7IrYy/view?usp=share_link>

Unzip it, and place “Exp” folder into ./Pretrain

Or you could pretrain it by yourself by running python ./Pretrain/main\_pretrain.py –exp\_name $your experiment name$ --dataset $Dataset used for pretraining$ --fold $which fold used for testing$ --project\_path $The path you save this project$ --data\_path $The data path you downloaded the data to in section 2$

# 4. Few-shot Classification by 3DG2D method

Running the following script

python main.py --exp\_name $Experiment name given by you$ --pretrain\_path $The pretrained backbone’s path obtained in section 3$ --dataset $the dataset used for training and testing$ --project\_path $the path you save the project$ --data\_path $The data folder path obtained in section 2$

After running the code, the experiment log file and tensorboard file will be saved into ./Exp/$Your selected dataset name$/ $Experiment name defined by you$

After each training epoch, the accuracy is evaluated on testing set, and the testing accuracy is recorded into the experiment log file.

The meaning of each argument in main.py is shown below:

**--exp\_name**: Experiment name defined by you. The experiment folder with this name will be created in Exp Folder

**--dataset**: which dataset you choose to use

**--epochs**: number of training epoch

**--decay\_ep**: parameter needed for learning rate scheduler

**--gamma:** parameter needed for learning rate scheduler

**--lr:** Learning rate

**--train:** The mode is for training if set it as true

**--seed:** Random seed

**--device:** determining if run the file with gpu or cpu, default value is ‘cuda’

**--lr\_sch:** Using learning rate scheduler if set as True

**--prj\_num:** projection number generated from each 3D mesh sample

**--pretrain:** Set true if using pretrained backbone

**--pretrain\_path:** The pretrained backbone’s path.

**--point\_support:** set true if using 3DG2D method for the training, otherwise, using traditional few-shot classification method for the training.

**--alpha:** alpha value mentioned in the paper to adjust the weight of Angle Inference Loss.

**--n\_way:** number of classes

**--k\_shot:** shot number of each class

**--query:** query image number of each class

**--fold:** The index of fold used for the testing. Here we perform n fold cross validation, which is as explained in the paper.

**--backbond:** ResNet here

**-- fs\_head:** few-shot head used for the experiment. All few-shot head code are in the ./Model/Head.

**-- project\_path**: The path you save the whole project’s code to

**--** **data\_path:** the path of the dataset folder. For example, if you select ModelNet40-LS as experiment, you should fill the path of ModelNet40-LS folder here.

# Reference

[1] Stojanov, Stefan, Anh Thai, and James M. Rehg. "Using shape to categorize: Low-shot learning with an explicit shape bias." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021.