EE 569 Discussion 14

Yao Zhu

04/16/2021

Contents

- Announcements
- Homework 6: Successive Subspace Learning
  - Will focus more on implementation details for P2

Announcements

HW6

- Issued: 04/14/2021
- Due: 04/30/21 11:59pm
- Start early!!

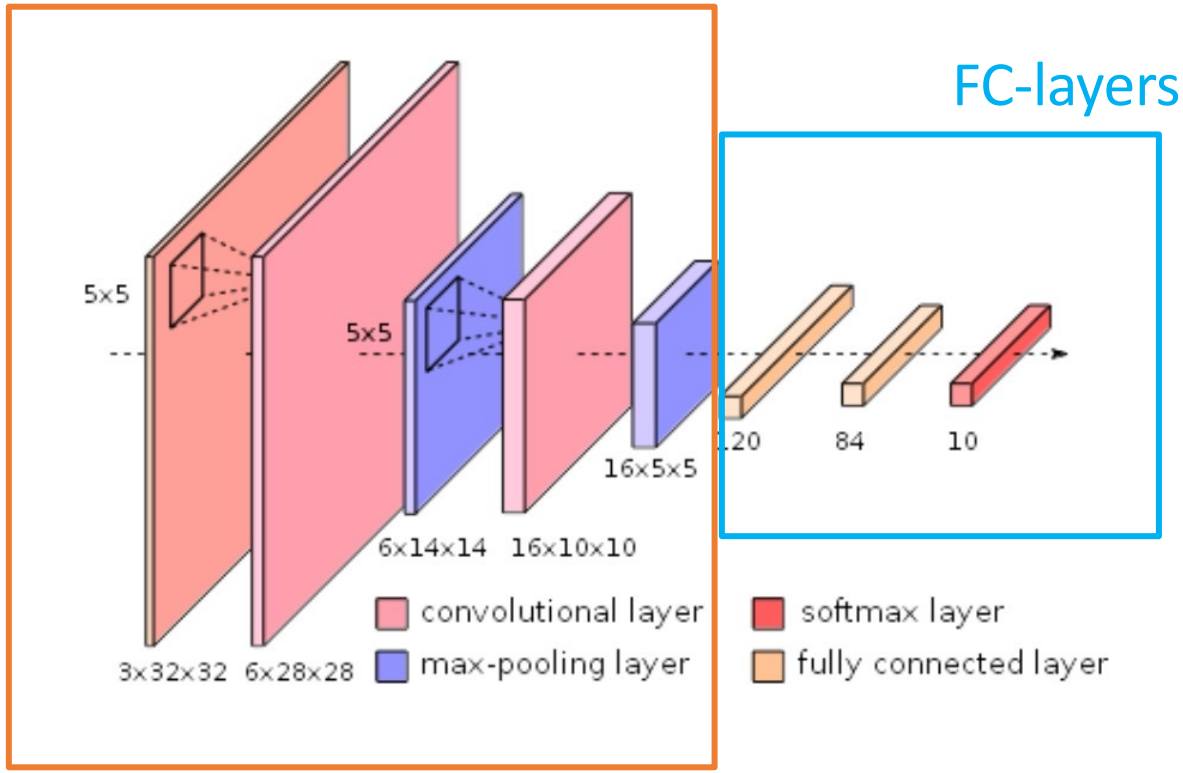
Summer Intern application

- Due 4/18/2021
- We welcome all applications, send us yours!

HW6 P1: origin of green learning

HW6 P1(a): FF-CNN and Saab Transform

Feedforward-designed Convolutional Neural Networks (FF-CNNs) [2]



FC-layers → Multi-stage LLSR

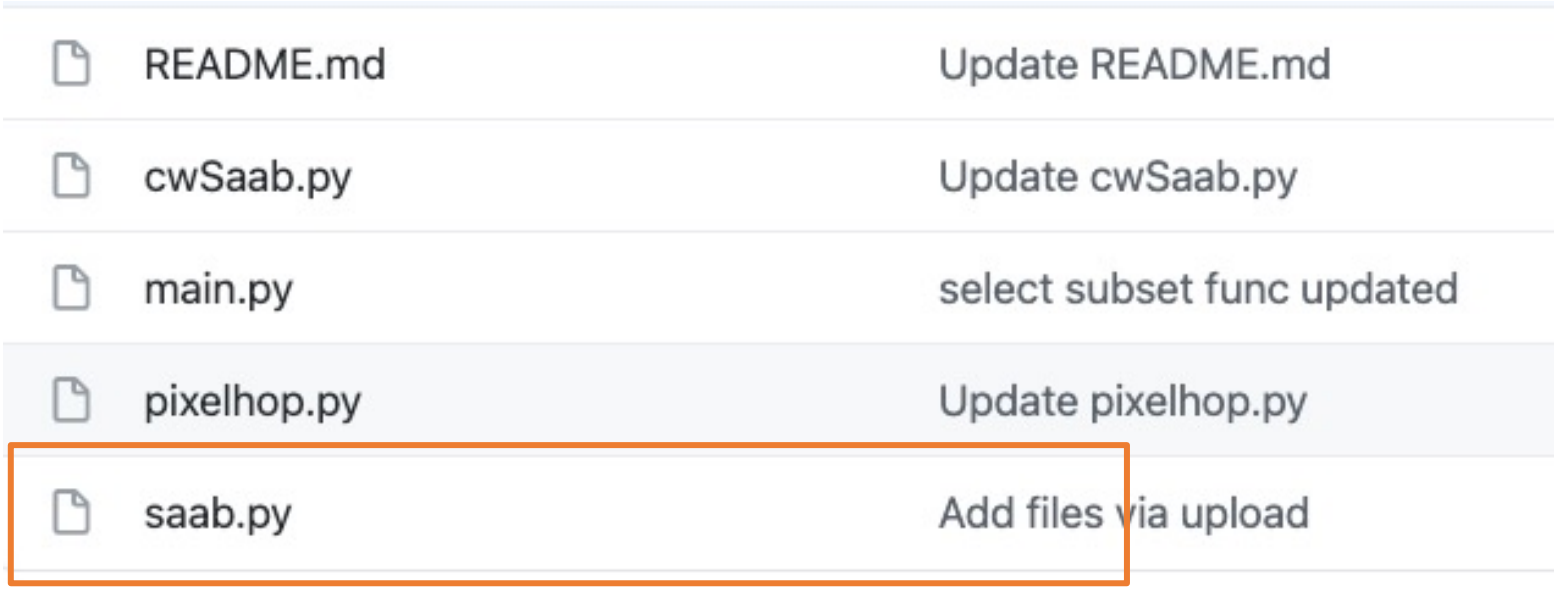
Conv layers →Multi-stage **Saab transform**

HW6 P1: origin of green learning

HW6 P1(a): FF-CNN and Saab Transform

Code for Saab Transform:

https://github.com/USC-MCL/Channelwise-Saab-Transform



HW6 P1(b): SSL Methods

1. PixelHop [3]

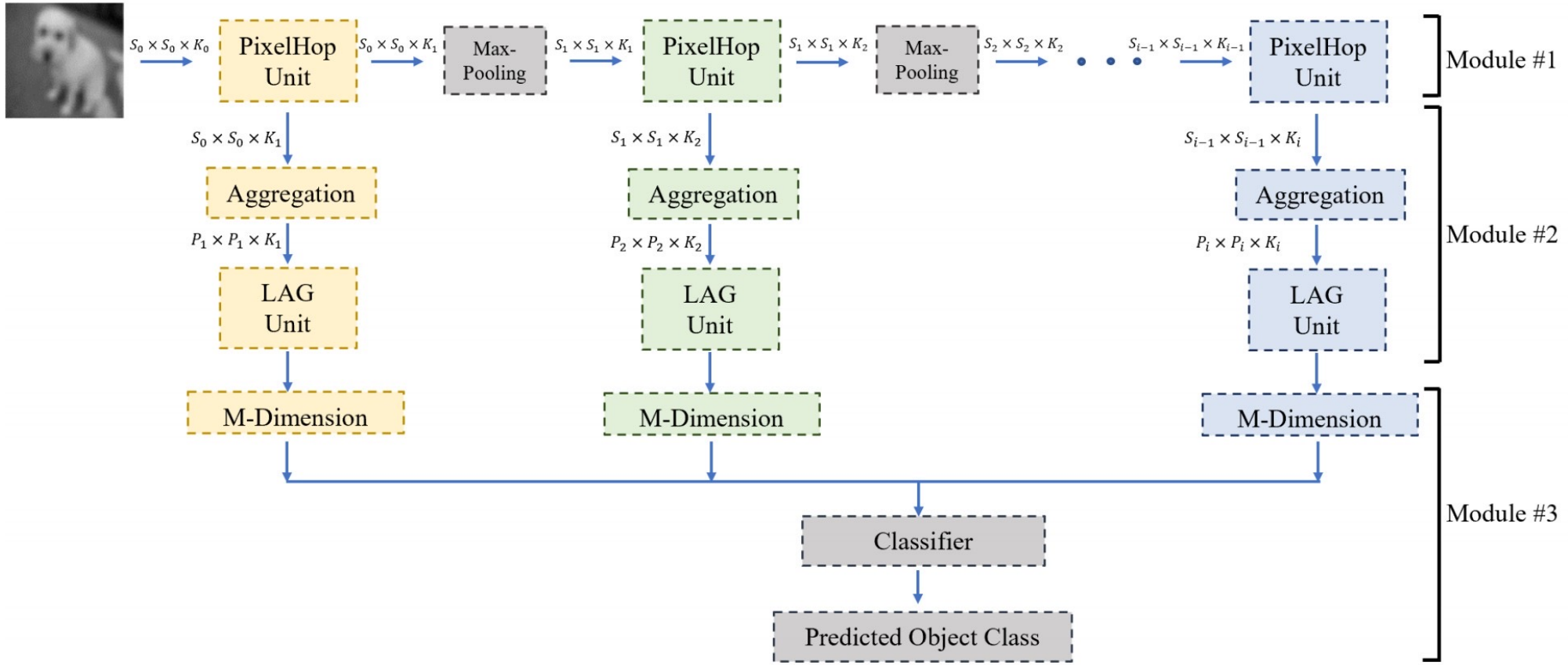


Figure 1: The block diagram of the PixelHop method.

HW6 P1(b): SSL Methods

2. PixelHop++ diagram in paper[4]

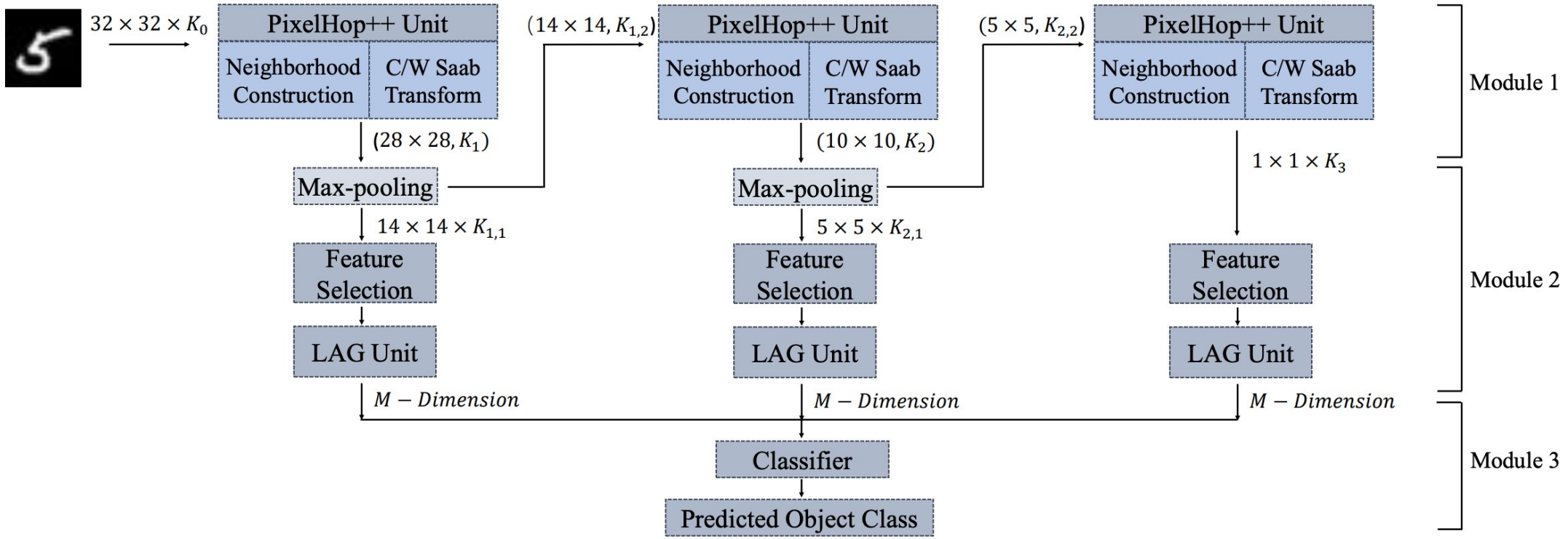


Fig. 1. The block diagram of the PixelHop++ method that contains three PixelHop++ Units in cascade.

HW6 P2: MNIST & Fashion-MNIST Classification using SSL

Things to be covered later:

- General guidance
- Explain the source code from high level
- Explain how to use the source code
- Implementation steps

HW6 P2: MNIST & Fashion-MNIST Classification using SSL

General guidance

- Source code for key modules are provided in the GitHub link
- Need to complete main.py file by yourself
- The link is not the official implementation for the PixelHop++ paper
- Your data should be channel-last,
  i.e. shape = *(N_image, Height, Width, **N_channel**)*

  instead of

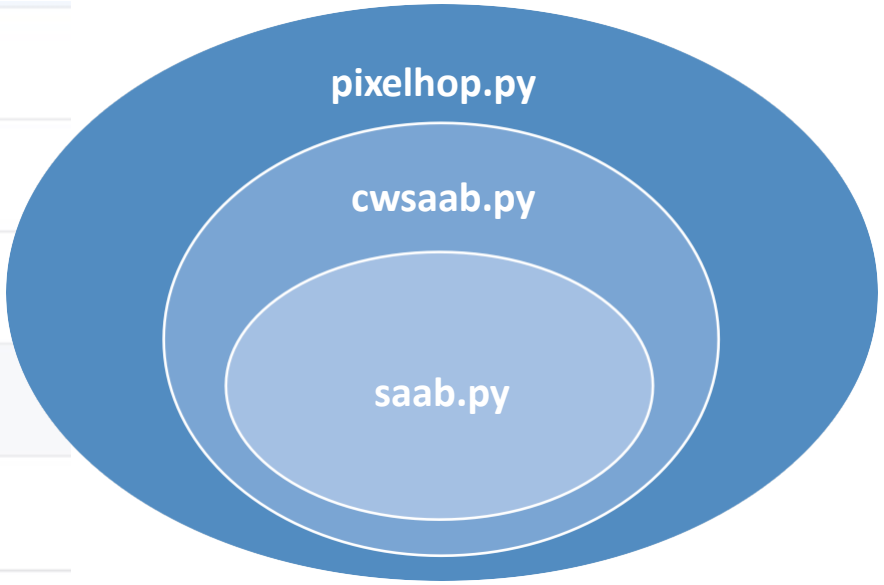
  *(N_image, **N_channel,** Height, Width)*

HW6 P2: MNIST & Fashion-MNIST Classification using SSL

More about the source code

README.md	Update README.md
cwSaab.py	Update cwSaab.py
main.py	select subset func updated
pixelhop.py	Update pixelhop.py
saab.py	Add files via upload

Feature-Selection Module



- Main.py directly uses pixelhop.py as feature extraction
- Module 2 and Module 3 should be constructed in main.py after feature extraction using pixelhop

HW6 P2: MNIST & Fashion-MNIST Classification using SSL

More about the source code

About *Neighborhood Construction (*write your own ***Shrink*** function*)*:

- Pooling

  Question: why pooling is ahead of neighborhood construction in Shrink function?
- Collect 5x5 (spatial) patches

Example:

2 Hop unit shrink argument settings

```
shrinkArgs = [{'func':Shrink, 'win':5, 'stride': 1, 'pad':2, 'pool': 1},
              {'func': Shrink, 'win':5, 'stride': 1, 'pad':0, 'pool': 2}]
```

'Pool' argument can be used as both pooling flag and pooling size

```
def Shrink(X, shrinkArg):
    #---- max pooling----
    pool = shrinkArg['pool']
    # TODO: fill in the rest of max pooling

    #---- neighborhood construction
    win = shrinkArg['win']
    stride = shrinkArg['stride']
    pad = shrinkArg['pad']
    ch = X.shape[-1]
    # TODO: fill in the rest of neighborhood construction

    if pad>0:
        X = np.pad(X,((0,0),(pad,pad),(pad,pad),(0,0)),'reflect')
    X = view_as_windows(X, (1,win,win,ch), (1,stride,stride,ch))
    return X.reshape(X.shape[0], X.shape[1], X.shape[2], -1)
```

HW6 P2: MNIST & Fashion-MNIST Classification using SSL

More about the source code

About Saab arguments:

- using Channel-wise structure or not

  PixelHop++: no channel-wise in Hop1, channel-wise for rest Hops

  PixelHop: no channel-wise for all Hops

- Need Bias or not (Hop1 no bias, rest Hops need bias)

Example of Saab arguments for 2 Hop PixelHop++:

```
SaabArgs = [{'num_AC_kernels':-1, 'needBias':False, 'cw': False},
            {'num_AC_kernels':-1, 'needBias':True, 'cw':True}]
```

HW6 P2: MNIST & Fashion-MNIST Classification using SSL

More about the source code

An example about how to import key modules and how to use it:

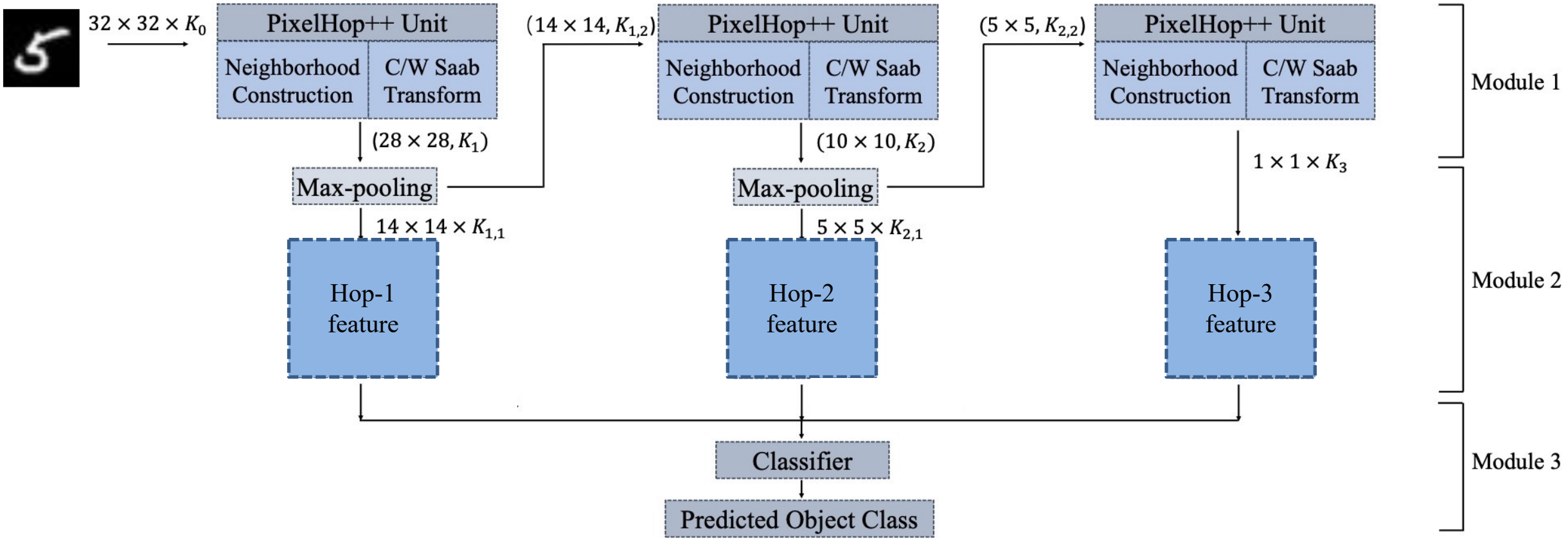
Class name in pixelhop.py

```
from pixelhop import Pixelhop
```

Filename under the same folder with main.py

```
p2 = Pixelhop(depth = 3,
              TH1 = 0.002,
              TH2 = 0.0005,
              SaabArgs = SaabArgs,
              shrinkArgs = shrinkArgs,
              concatArg = concatArg).fit(x_train)
```

HW6 P2: MNIST & Fashion-MNIST Classification using SSL



PixelHop++ diagram used in HW6

HW6 P2: MNIST & Fashion-MNIST Classification using SSL

Implementation Steps

1. Train Module 1:
    i. (Pre-process data): channel-last, int→float, rescale to 0-1, etc.
    ii. Randomly select 10K training images: need to be balanced (1000 per class) if your memory can hold 50k training images, ignore this step
    iii. Define your Neighborhood Construction function,*SaabArgs, ShrinkArgs, ConcatArg*
    iv. Use *pixelhop.py* to train (*pixelhop.fit*) your PixelHop model of a certain depth
    v. Save the PixelHop model

HW6 P2: MNIST & Fashion-MNIST Classification using SSL

Implementation Steps

2. Extract only Hop3 features from Module 1:
   - Use all the 50K training images, do *pixelhop.transform_singleHop* to extract features only from hop 3
   - Get 10K testing images Hop3 feature
   - May need batch processing to save memory. Try different programming methods on your own

HW6 P2: MNIST & Fashion-MNIST Classification using SSL

Implementation Steps

4. Module 3:
    i. For each image, the features from Hop3 is a long feature vector with shape 1*1*K3
    ii. Choose a classifier (e.g. XGBoost)
    iii. Train the classifier on the Hop3 feature vector of training set
    iv. Test on the testing set

HW6 P2: MNIST & Fashion-MNIST Classification using SSL

Parameters

Table 1 Choice of hyper-parameters of PixelHop++ model for MNIST dataset

Spatial Neighborhood size in all PixelHop++ units	5x5
Stride	1
Max-pooling	(2x2) -to- (1x1)
Energy threshold for intermediate nodes (*TH1*)	0.005
Energy threshold for discarded nodes (*TH2*)	0.001
Classifier	XGBoost
Number of estimators in classifier	100

TH1 and TH2 for **PixelHop++**
For **PixelHop**, only need to finetune TH2

HW6 P2: MNIST & Fashion-MNIST Classification using SSL

Dependencies and their recommended versions for main.py

- Sklearn 0.24.1
- Scikit-image 0.18.1
- xgboost 1.4.0
- Tensorflow.keras 2.4.0


Q
A