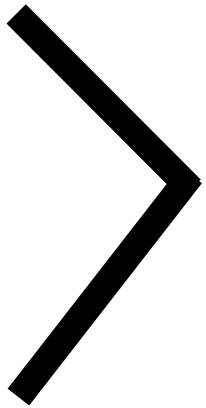
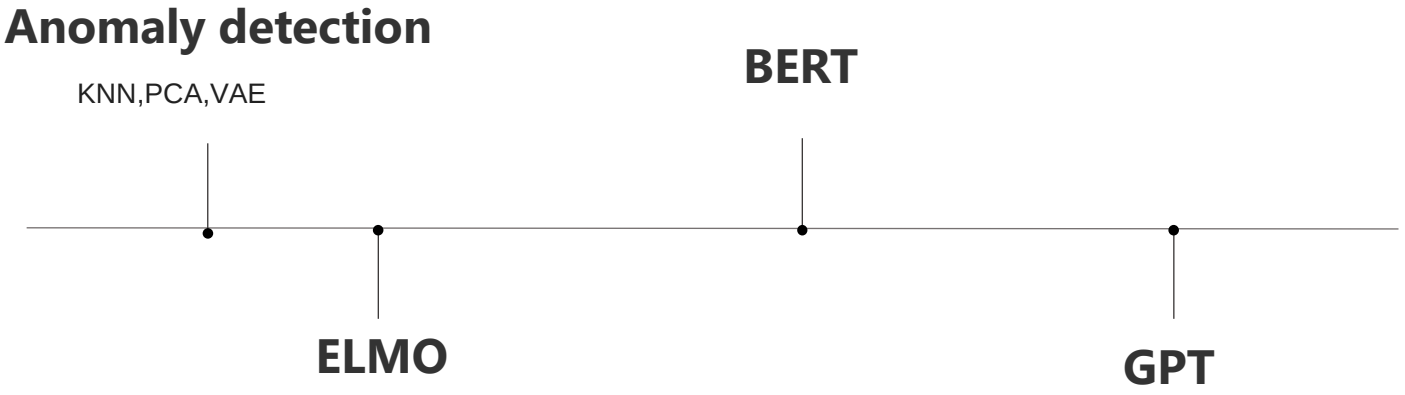
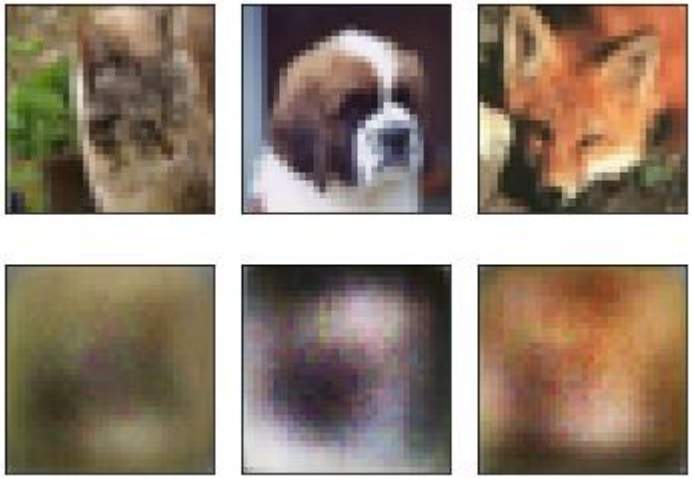
每周总结




Anomaly detection
KNN,PCA,VAE
BERT
ELMO
GPT

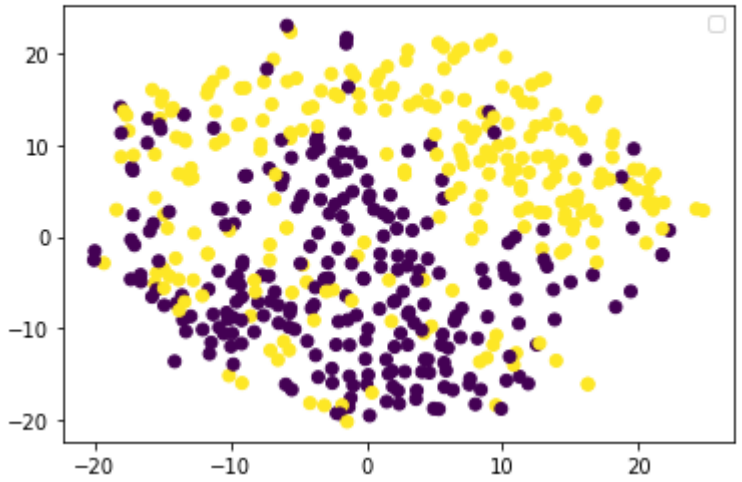
```
self.encoder = nn.Sequential(
        nn.Conv2d(3, 64, 3, stride=1, padding=1),
        nn.ReLU(True),
        nn.MaxPool2d(2),
        nn.Conv2d(64, 128, 3, stride=1, padding=1),
        nn.ReLU(True),
        nn.MaxPool2d(2),
        nn.Conv2d(128, 256, 3, stride=1, padding=1),
        nn.ReLU(True),
        nn.MaxPool2d(2)
)

self.decoder = nn.Sequential(
        nn.ConvTranspose2d(256, 128, 5, stride=1),
        nn.ReLU(True),
        nn.ConvTranspose2d(128, 64, 9, stride=1),
        nn.ReLU(True),
        nn.ConvTranspose2d(64, 3, 17, stride=1),
        nn.Tanh()
)
```

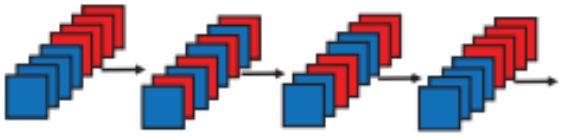


The clustering accuracy is: 0.722

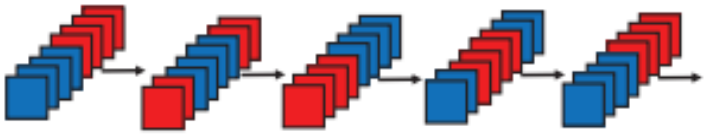
The clustering result:



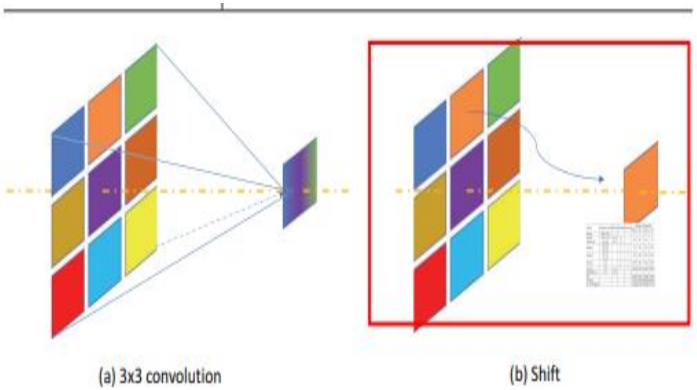
Synetgy: Algorithm-hardware Co-design for ConvNet Accelerators on Embedded FPGAs



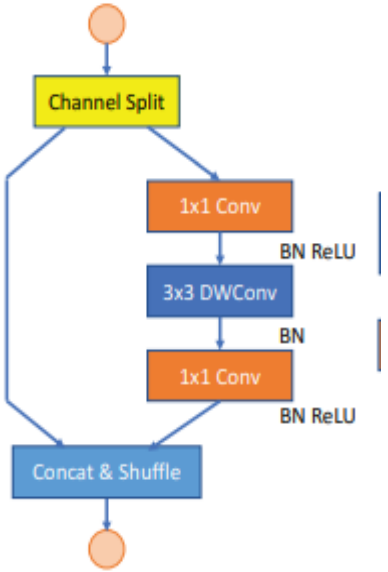
(a) Transpose based channel shuffle

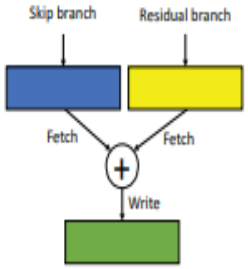


(b) Our channel shuffle

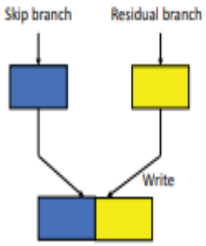


(a) 3x3 convolution (b) Shift





(a) Additive skip connection



(b) Concatenative skip connection

Layer	Output size	KSize	Stride	Repeat	Output channels			
					0.5×	1×	1.5×	2×
Image	224×224				3	3	3	3
Conv1	112×112	3×3	2	1	24	24	24	24
MaxPool	56×56	3×3	2					
Stage2	28×28		2	1	48	116	176	244
	28×28		1	3				
Stage3	14×14		2	1	96	232	352	488
	14×14		1	7				
Stage4	7×7		2	1	192	464	704	976
	7×7		1	3				
Conv5	7×7	1×1	1	1	1024	1024	1024	2048
GlobalPool	1×1	7×7						
FC					1000	1000	1000	1000
FLOPs					41M	146M	299M	591M
# of Weights					1.4M	2.3M	3.5M	7.4M

Table 5: Overall architecture of ShuffleNet v2, for four different levels of complexities.

1. 224X224->192X192
2. 将conv5的输出为4x4
3. Globalpool为HxW为一组

1. Conv5：
   input size=（192，4，4）
   output size=（512，4，4）

```
void conv_last(float input[1][192][4][4],
    float weight[512][192][1][1],
    float bias[512],
    float output[1][512][4][4]){

    for(int co = 0;co<512;co++){
        for(int h = 0;h<4;h++){
            for(int w = 0;w<4;w++){
                float sum = 0;
                for(int ci = 0;ci<192;ci++){
                    sum += weight[co][ci][0][0]*input[0][ci][h][w];
                }
                float result = sum + bias[co];
                output[0][co][h][w] = (result > 0)? result : 0.0f;
            }
        }
    }
}
```

2. Global pool：
   input size=（512，4，4）
   output size=(512,) to fc

```
void avgpool(float input[1][512][4][4],
    float output[512]){

    for(int co = 0;co<512;co++){
        float sum = 0;
        for(int h = 0;h<4;h++){
            for(int w=0;w<4;w++){
                sum += input[0][co][h][w];
            }
        }
        output[co] = sum/16;
    }
}
```

3. 如何处理padding

```
sum += weight[co][ci][m][n] * (( h+m-1 >= 0 && w+n-1 >= 0 && h+m-1 < 32 && w+n-1 < 32) ?input[0][ci][h+m-1][w+n-1]:0);
```