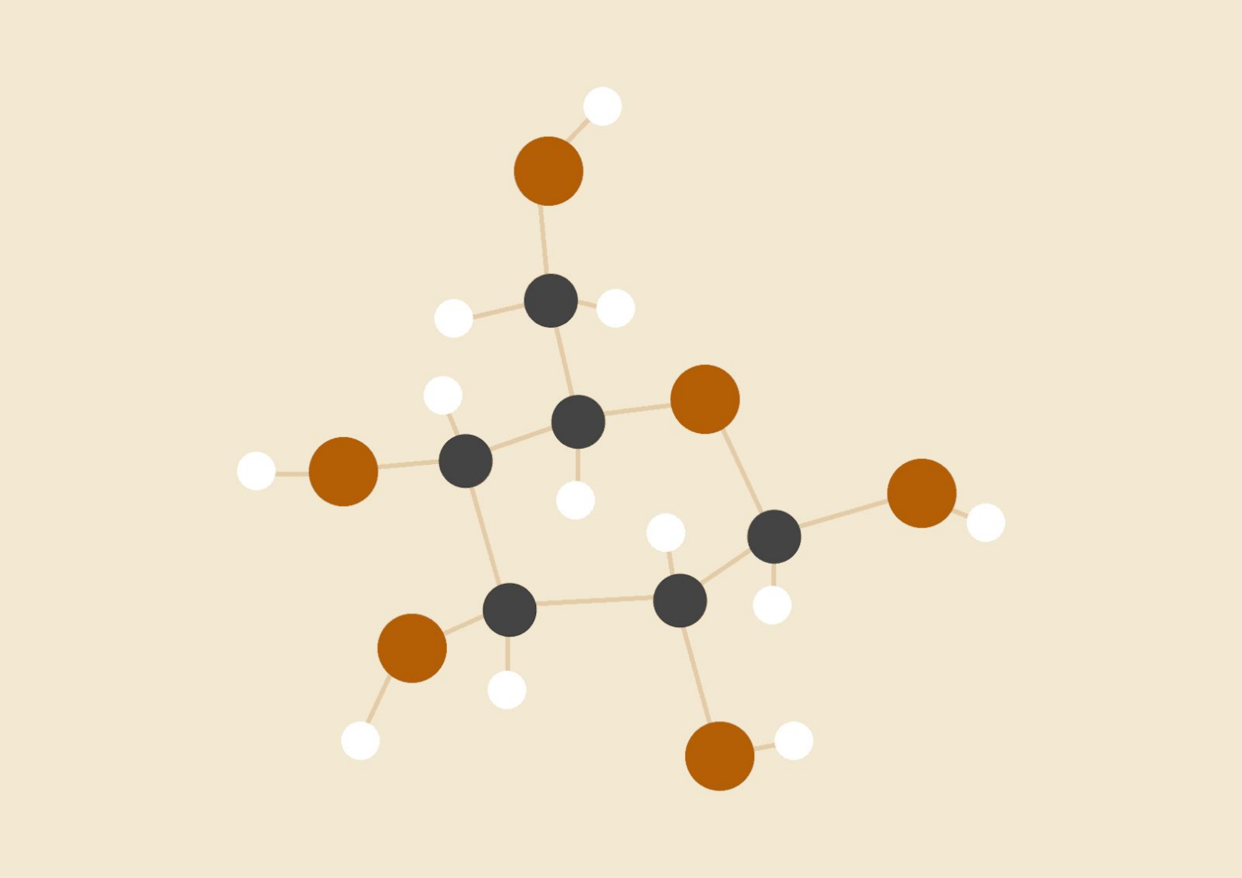
Final Submission

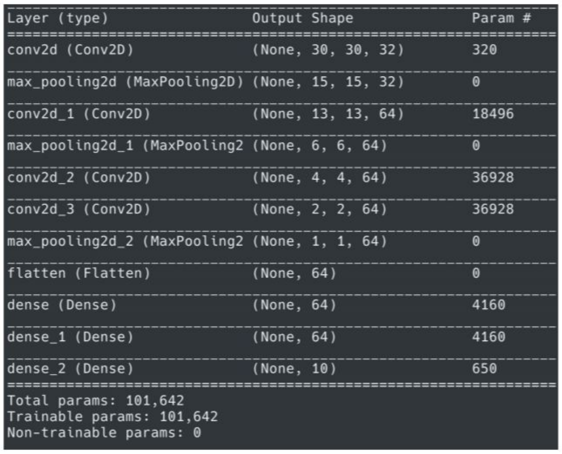
Machine Learning Project



Daria Vaskovskaya

Introduction

During previous submission a simple net was presented: model.summary() has given the following output:



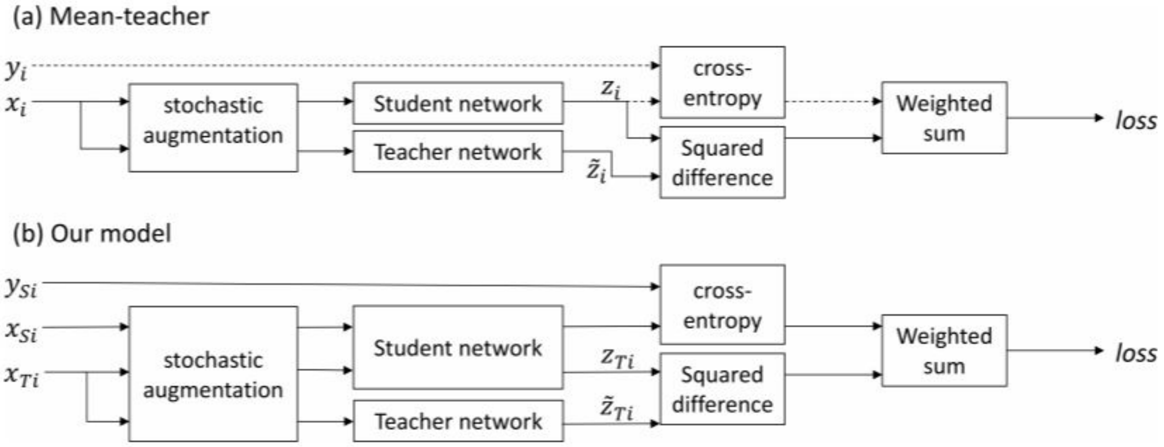
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 30, 30, 32)        320
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 13, 13, 64)        18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 6, 6, 64)          0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 4, 4, 64)          36928
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 2, 2, 64)          36928
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 1, 1, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 64)                0
_________________________________________________________________
dense (Dense)                (None, 64)                4160
_________________________________________________________________
dense_1 (Dense)              (None, 64)                4160
_________________________________________________________________
dense_2 (Dense)              (None, 10)                650
=================================================================
Total params: 101,642
Trainable params: 101,642
Non-trainable params: 0
```

There were 11 layers. In addition, only transformation to grayscale was used.

Now another model is used. Architecture is fully described in the paper "Self-ensembling for visual domain adaptation" by French, G, Mackiewicz, M, Fisher, M. [1]

Briefly, the technique is derived from the mean teacher variant [2] of temporal ensembling [3] with a number of modifications for challenging domain adaptation scenarios.

The achieved accuracy is close to that of a classifier trained in a supervised fashion.



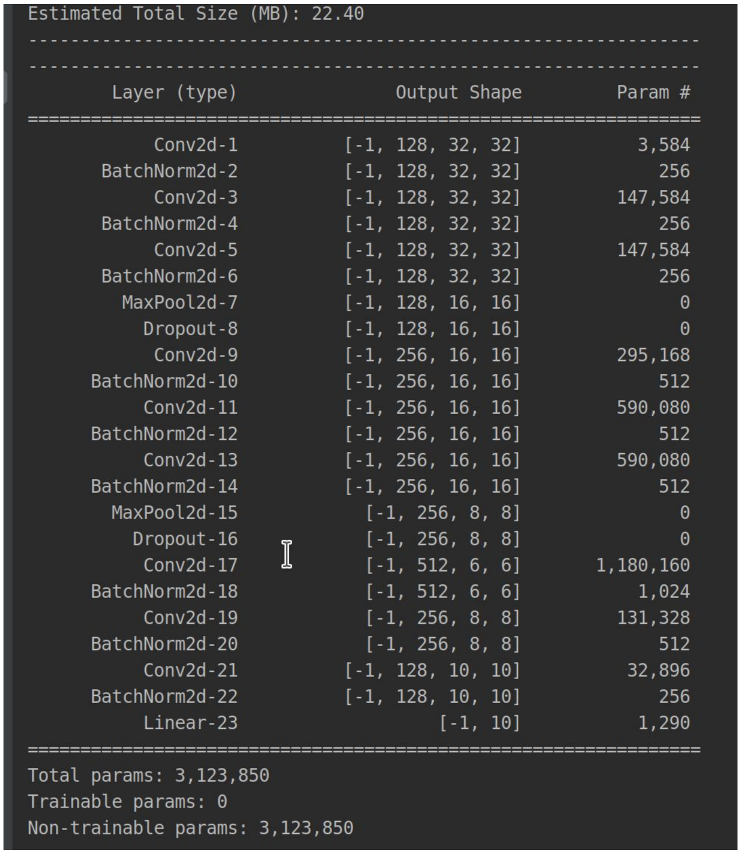
Loss is minimized in the same way as in [2], cross-entropy loss is applied to labeled source samples and unsupervised self-ensembling loss to target samples.
As in [2], self-ensembling loss is computed as the mean-squared difference between predictions produced by the student and teacher networks with different augmentation, dropout and noise parameters.

But the in contrast to the mean teacher model this one has separate source and target paths. Inspired by [4], mini-batches are processed from the source and target datasets separately (per iteration) so that batch normalization uses different normalization statistics for each domain during training.

IMPLEMENTATION

My implementation is adapted from the one from the paper [1] (http://github.com/Britefury/self-ensemble-visual-domain-adapt)

Code can be found: https://www.kaggle.com/bananaandbread/kernel33eff5cd9f
Layers used in a new model:

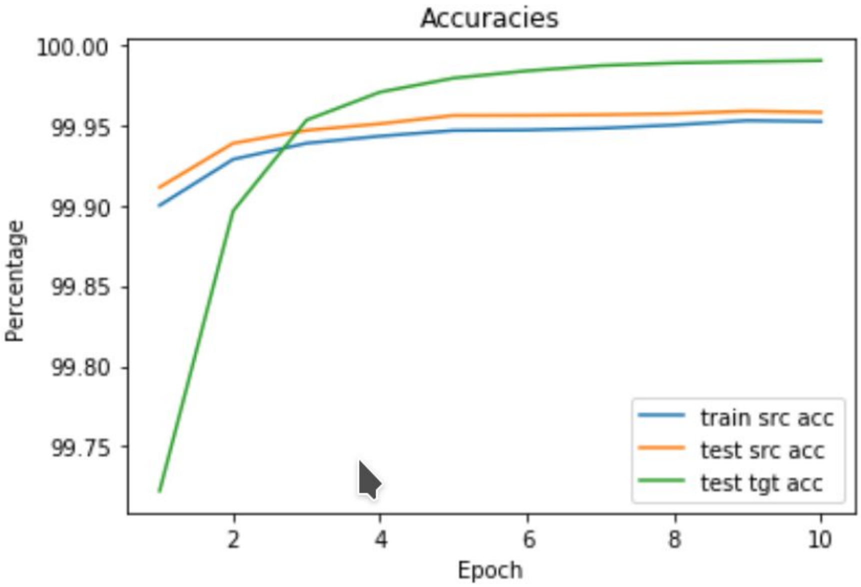


```
Estimated Total Size (MB): 22.40
----------------------------------------------------------------
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1         [-1, 128, 32, 32]           3,584
       BatchNorm2d-2         [-1, 128, 32, 32]             256
            Conv2d-3         [-1, 128, 32, 32]         147,584
       BatchNorm2d-4         [-1, 128, 32, 32]             256
            Conv2d-5         [-1, 128, 32, 32]         147,584
       BatchNorm2d-6         [-1, 128, 32, 32]             256
         MaxPool2d-7         [-1, 128, 16, 16]               0
           Dropout-8         [-1, 128, 16, 16]               0
            Conv2d-9         [-1, 256, 16, 16]         295,168
      BatchNorm2d-10         [-1, 256, 16, 16]             512
           Conv2d-11         [-1, 256, 16, 16]         590,080
      BatchNorm2d-12         [-1, 256, 16, 16]             512
           Conv2d-13         [-1, 256, 16, 16]         590,080
      BatchNorm2d-14         [-1, 256, 16, 16]             512
        MaxPool2d-15           [-1, 256, 8, 8]               0
          Dropout-16           [-1, 256, 8, 8]               0
           Conv2d-17           [-1, 512, 6, 6]       1,180,160
      BatchNorm2d-18           [-1, 512, 6, 6]           1,024
           Conv2d-19           [-1, 256, 8, 8]         131,328
      BatchNorm2d-20           [-1, 256, 8, 8]             512
           Conv2d-21         [-1, 128, 10, 10]          32,896
      BatchNorm2d-22         [-1, 128, 10, 10]             256
           Linear-23                  [-1, 10]           1,290
================================================================
Total params: 3,123,850
Trainable params: 0
Non-trainable params: 3,123,850
```

Important parameters:
teacher_alpha = 0.99 - Teacher EMA alpha (decay)
learning_rate = 0.001
src_intens_flip = True - src aug colour; enable intensity flip
src_intens_scale_range = "0.25:1.5" - src aug colour; intensity scale range `low:high`
src_intens_offset_range = "-0.5:0.5" - src aug colour; intensity offset range `low:high` (-0.5:0.5 for mnist-svhn)
tgt_intens_flip = True - tgt aug colour; enable intensity flip
tgt_offset_range = "-0.5:0.5" - tgt aug colour; intensity offset range `low:high` (-0.5:0.5 for mnist-svhn)
tgt_intens_scale_range = "0.25:1.5" - tgt aug colour; intensity scale range
tgt_intens_offset_range = "-0.5:0.5" - tgt aug colour; intensity offset range `low:high`
batch_size = 256
src_hflip = False - src aug xform: enable random horizontal flips
src_xlat_range = 2.0 - src aug xform: translation range
src_affine_std = 0.1 - src aug xform: random affine transform std-dev
src_intens_flip = False - src aug colour; enable intensity flip
tgt_hflip = False - tgt aug xform: enable random horizontal flips
tgt_xlat_range = 2.0 - tgt aug xform: translation range
tgt_affine_std = 0.1 - tgt aug xform: random affine transform std-dev

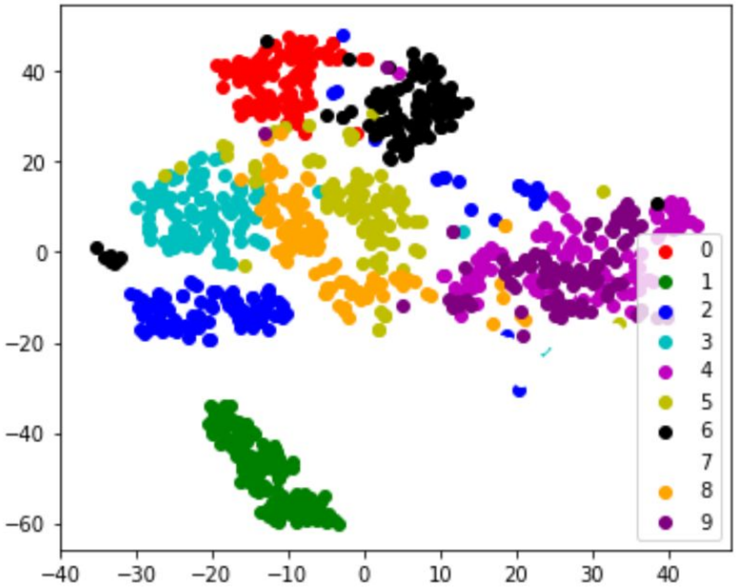
RESULTS

Final accuracies:
a) on svhn-train,
b) on svhn-test,
c) on mnist-test.

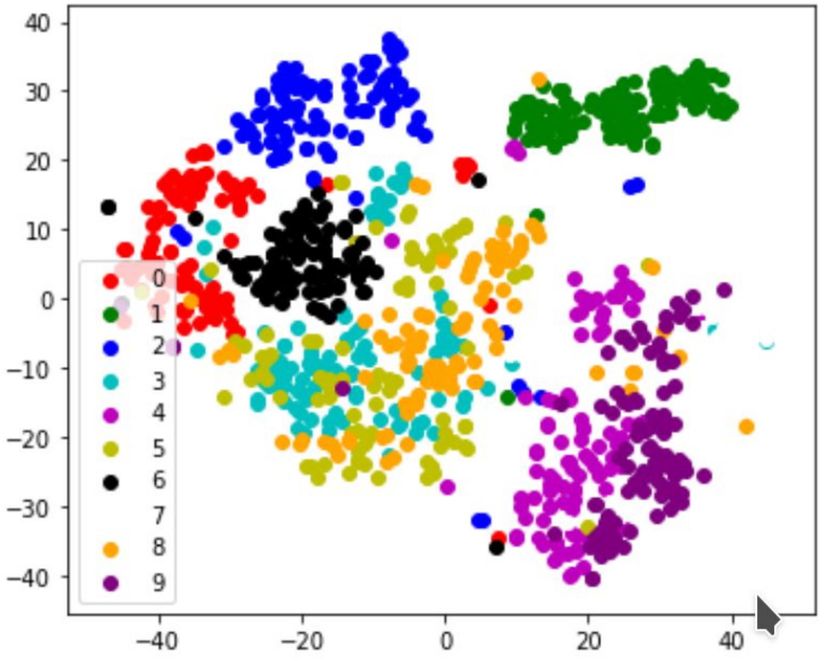


Src accuracy might be lower than tgt because of some inaccurate labels in svhn.
Plots of Latent space (next page):

Before training



After DA:



Domain adaptation is the essence of this model, by removing all the features corresponding to the domain adaptation, we get mean teacher architecture. It trains and reduces the gaps between two domains at the same time.

Remove some parts of the model and report the new accuracies:
If augmentation is removed model reaches 94% accuracy in 30 epochs

CONCLUSIONS

Because the achieved accuracy is close to that of a classifier trained in a supervised fashion, it is possible to say that the domain gap is solved. Also, because of that reason, it is hard to improve model and get better results.

REFERENCES

[1] Geoffrey French, Michal Mackiewicz, Mark Fisher. Self-ensembling for visual domain adaptation.2017

[2] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. 2017

[3] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In ICLR, 2017.

[4] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation. arXiv preprint arXiv:1603.04779, 2016.