here is a rich body of literature tackling the problem of domain adaptation. In this work, we focus on domain adaptation between numbers of SVHN and MNIST dataset. Our goal was to build a model that trains on SVHN and predicts MNIST.

Pictures in datasets are different colors and have a different number of channels. I walk through state of the art and found that many models that used CNN or GAN don't reach 80% of accuracy. Also, I walk through new articles and found that many of them need too many epoch (>1000). So I decide to use the existing implementation of [1]. It takes less than 300 epochs to reach at least 90% accuracy. The layers of the model used in this implementation are below.

Description	Shape
32 × 32 RGB image	32 × 32 × 3
Conv 3 × 3 × 128, pad 1, batch norm	32 × 32 × 128
Conv 3 × 3 × 128, pad 1, batch norm	32 × 32 × 128
Conv 3 × 3 × 128, pad 1, batch norm	32 × 32 × 128
Max-pool, 2x2	16 × 16 × 128
Dropout, 50%	16 × 16 × 128
Conv 3 × 3 × 256, pad 1, batch norm	16 × 16 × 256
Conv 3 × 3 × 256, pad 1, batch norm	16 × 16 × 256
Conv 3 × 3 × 256, pad 1, batch norm	16 × 16 × 256
Max-pool, 2x2	8 × 8 × 256
Dropout, 50%	8 × 8 × 256
Conv 3 × 3 × 512, pad 0, batch norm	6 × 6 × 512
Conv 1 × 1 × 256, batch norm	6 × 6 × 256
Conv 1 × 1 × 128, batch norm	6 × 6 × 128
Global pooling layer	1 × 1 × 128
Fully connected, 10 units, softmax	10

Figure 1. Layers of Model

The model builds upon the mean teacher semi-supervised learning model [2]. The structure of the mean teacher model [2] – is shown in Figure 2a. The student network is trained using gradient descent, while the weights of the teacher network are an exponential moving average of those of the student. During training each input sample xi is passed through both the student and teacher networks, generating predicted class probability vectors z_i (student) and $\widehat{z_i}$ (teacher). Different dropout, noise and image translation parameters are used for the student and teacher pathways.

In the paper [1] minimise the same loss as in [2]; they apply cross-entropy loss 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 (z_{T_i}) and teacher ($\widehat{z_{T_i}}$) networks with different augmentation, dropout and noise parameters.

Batch size can imply model in next manner: if a batch is too small your model will too much rely on certain examples and variance will be big. Otherwise, big number of elements in batch make the model unable to see hidden patterns due to big normalization and increase the bias as well. This is why batch size equal 64, for example, is a good choice.

Dropout is a good way of normalization and making the model not to rely on special features of certain examples and bathes but finding the needed level (probability of applying) may be hard.

Model parameters (not default): batch_size = 256, learning_rate = 0.001. Model parameters (default): loss ='var', double_softmax = False, confidence_thresh = 0.96837722,

rampup = 0, teacher_alpha = 0.99, fix_ema = False, unsup_weight = False, cls_bal_scale = False, cls_bal_scale_range = 0.0, cls_balance = 0.005, cls_balance_loss = 'bce', combine_batches = False.

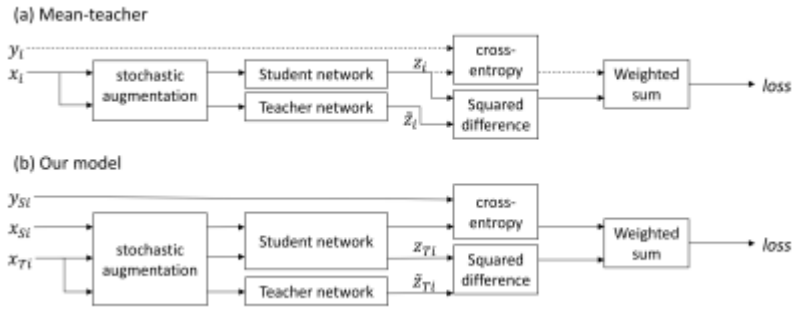

Figure 2: The network structures of the original mean teacher model and student model. Dashed lines in the mean teacher model indicate that ground truth labels – and therefore cross-entropy classification loss – are only available for labeled samples.

Accuracy on baseline is present in table #1. Accuracy of new model is present in table #2. Latent space before DA can see on Figure 3, after DA can see on Figure 4. Accuracy flow can be seen on Figure 5.

Table 1. Accuracy on each of the datasets on baseline

Dataset	Accuracy (%)
svhn-test	91.99
mnist-test	70.48

Table 2. Accuracy on each of the datasets on new model

Dataset	Accuracy (%)
svhn-test	94.526
mnist-test	82.29

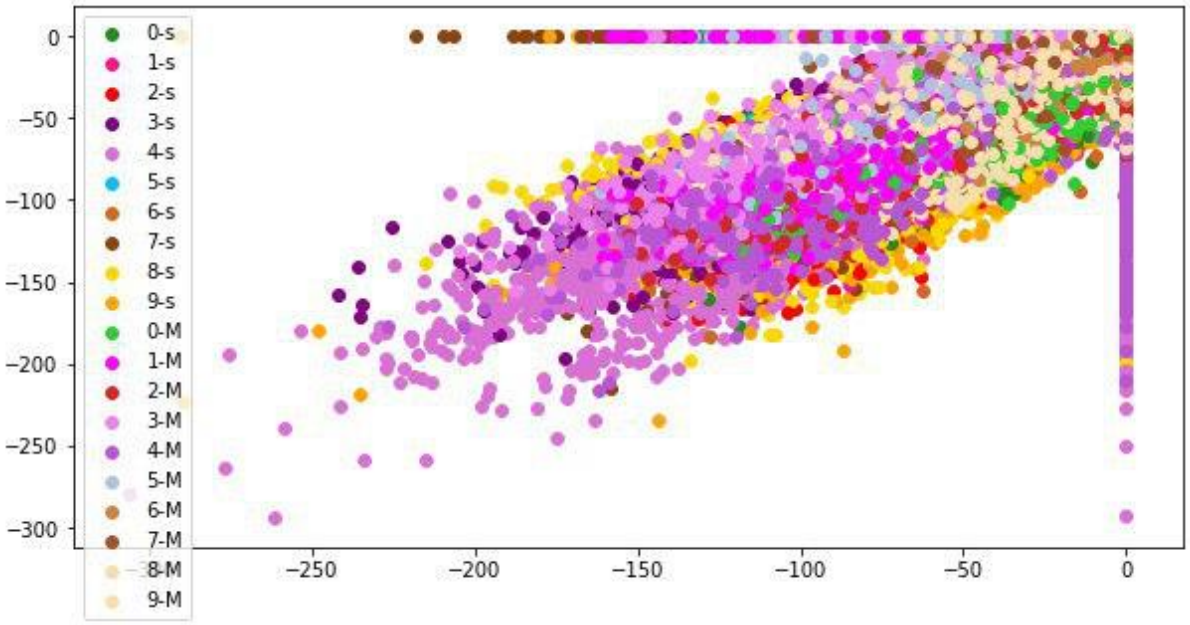

Figure 3. Latent space before DA

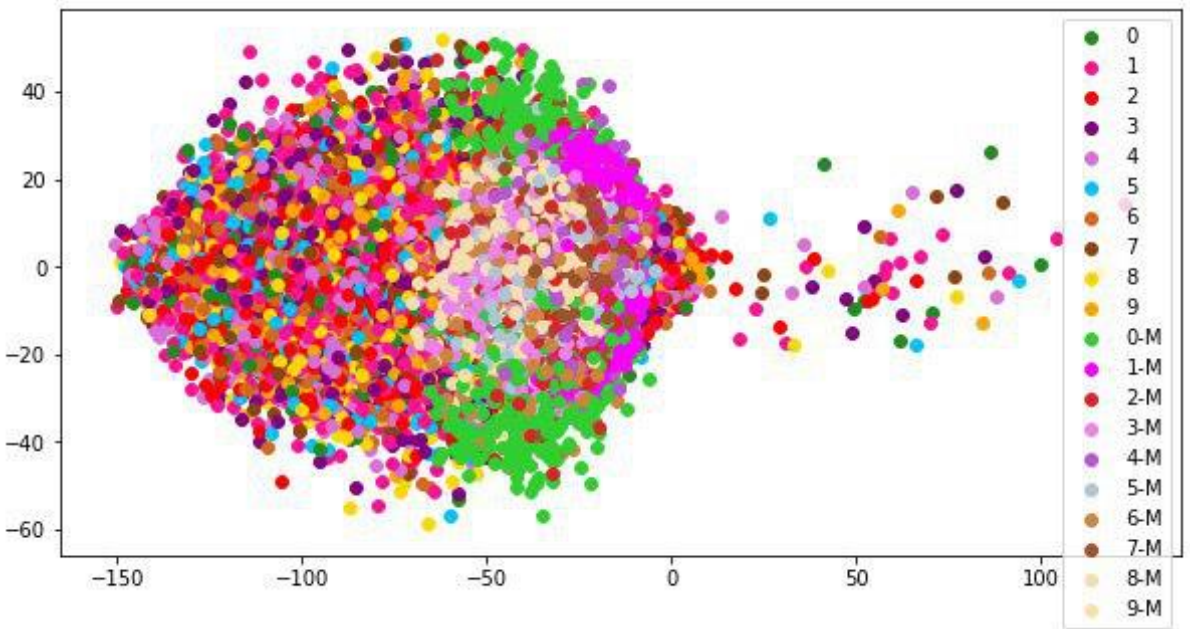

Figure 4. Latent space after DA

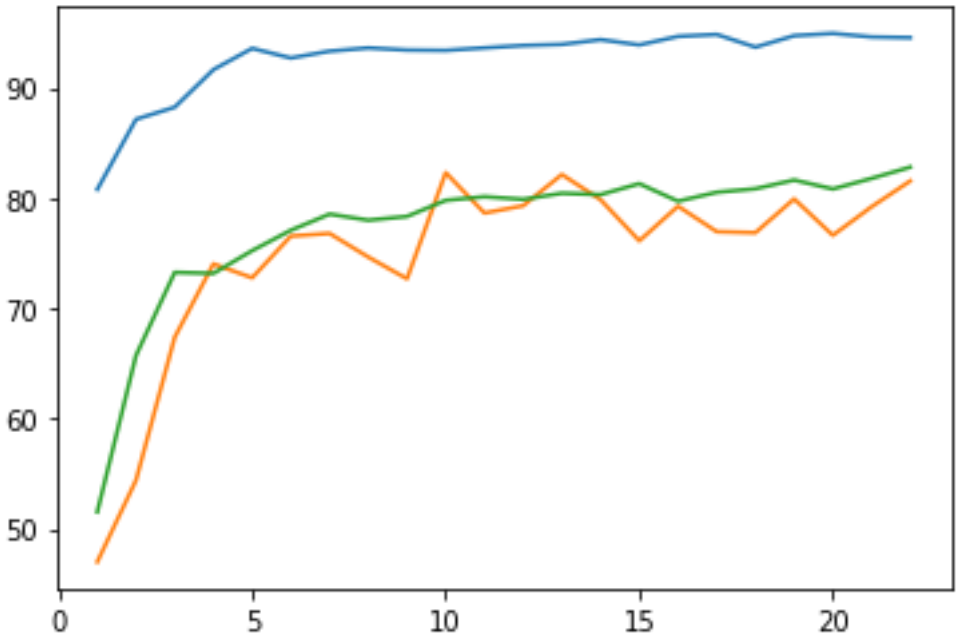

Figure 5. Accuracy flow.

Personal Thoughts

Domain adaptation is difficult task, you need a lot of time to build model from scratch. It could take a lot of time for training. In most cases on such task working team of several people. To find new Idea you need at least 3 month and research all (mostly new) articles in this area. Domain adaptation can used to help build simulations for self-driven cars, gestures [4], mimics of eyes [4], Political Document Analysis [3].

References

1. French, Geoffrey, Michal Mackiewicz, and Mark Fisher. "Self-ensembling for visual domain adaptation." *arXiv preprint arXiv:1706.05208* (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. Shrey Desai , Barea Sinno , Alex Rosenfeld , and Junyi Jessy Li. Adaptive Ensembling: Unsupervised Domain Adaptation for Political Document Analysis. doarXiv: 1910.12698v1 [cs.CL] 28 Oct 2019
4. Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Josh Susskind, Wenda Wang, Russ Webb. Learning from Simulated and Unsupervised Images through Adversarial Training. arXiv:1612.07828v2 [cs.CV] 19 Jul 2017
5. Repository [online] https://github.com/Britefury/self-ensemble-visual-domain-adapt
6. Colab notebook [online] https://colab.research.google.com/drive/12ohW0Dp_68JESdmwR9rgELGNFMLRoOvQ