BS6207 Assignment 4

SONG YIFAN

Table 1 result:

(Steps=2000)

Autoencoder vs. without Autoencoder

UCC and UCC^{2+} models had autoencoder branch and they were optimized jointly over both autoencoder loss and UCC loss. $UCC_{\alpha=1}$ and $UCC_{\alpha=1}^{2+}$ don't have autoencoder branch and optimized only by UCC loss. So, I just change the weight of the loss of autoencoder as 0 in the model to make sure that don't include the autoencoder loss.

Different uccs

UCC and $UCC_{\alpha=1}$ models trained on bags with labels of ucc1 to ucc4. (ucc_start=1, ucc_end=4)

UCC^{2+} and $UCC_{\alpha=1}^{2+}$ models trained on bags with labels of ucc2 and ucc4 (ucc_start=2, ucc_end=4).

	ucc acc.	clustering acc.
UCC	0.960	0.951
UCC^{2+}	0.954	0.949
$UCC_{\alpha=1}$	0.970	0.950
$UCC_{\alpha=1}^{2+}$	0.947	0.904

From the result we can see that the ucc accuracy of UCC and UCC^{2+} seems higher than $UCC_{\alpha=1}$ and $UCC_{\alpha=1}^{2+}$, which means autoencoder helps to generate more accurate result for same steps. The score of ucc2 to ucc4 is less than ucc1 to ucc4. That because of the ucc prediction task becomes easier at the absence of pure sets and models.

Try different feature extractors

Different instances:

	Instance=16	Instance =32	Instance =64
Ucc acc	0.950	0.96	1.00

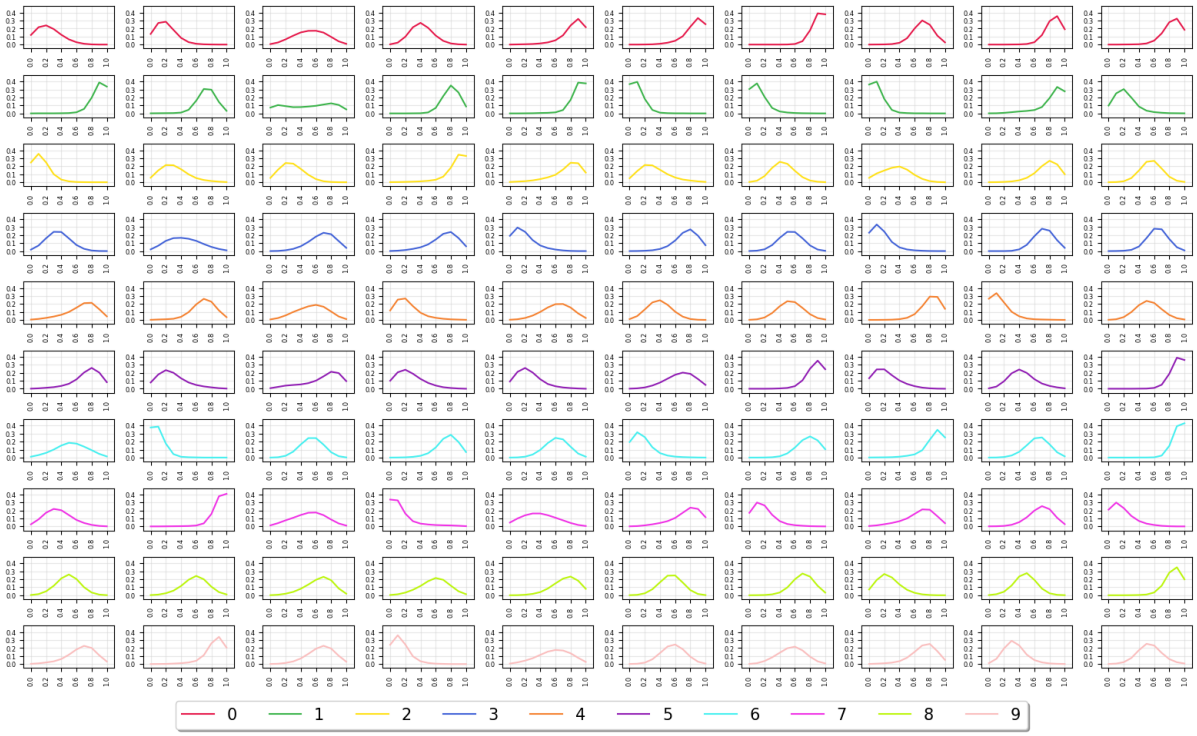
More instance means more complex the model is, so as expected, more instances, more running time, and more accurate. Instance means features in each bag, so more features will produce more accurate result.

Try different activation function (Relu to Tanh)

	ReLu	Tanh
Ucc_acc	0.960	0.610

ReLu is better than tanh. The biggest advantage of ReLu is the non-saturation of its gradient, which greatly speeds up the convergence of stochastic gradient descent compared to the tanh function.

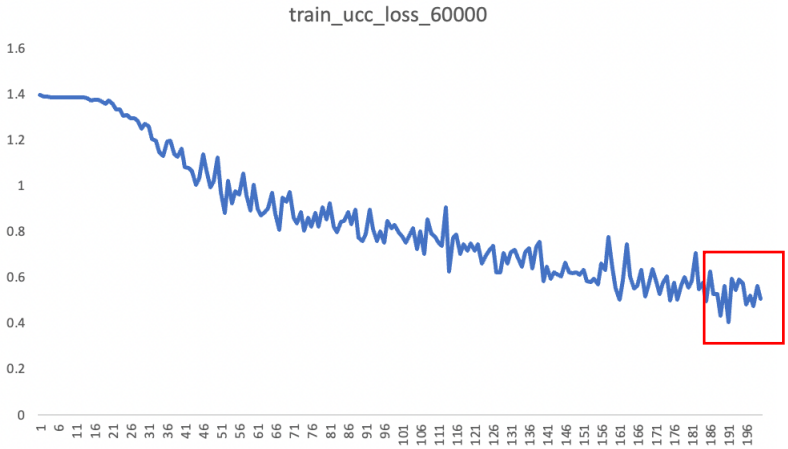
Distribution regression:



The kernel density estimation layer is used to construct the distribution of feature extracted by autoencoder. The figure shows above is the result of distribution regression. As we can see that the distribution of features in each class are not same. It means that KDE is a very efficient way to separate the distributions into different class.

Try different training number.

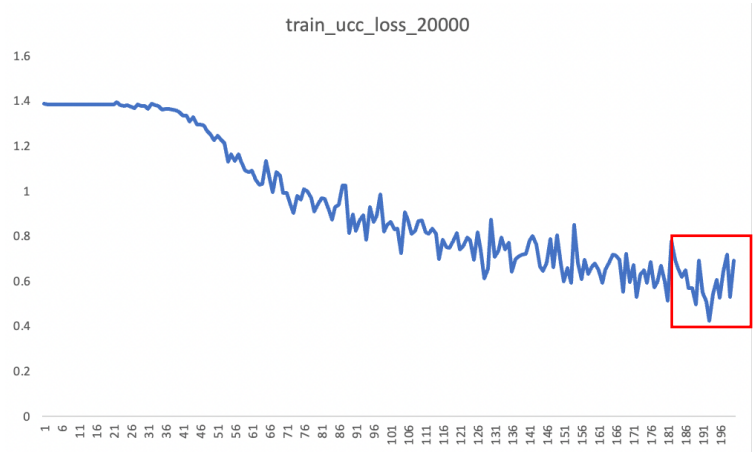
1. 60000 for training (default) ucc_auc=1.00



Seems like over fitting.

2. 20000 for training ucc_auc=0.950

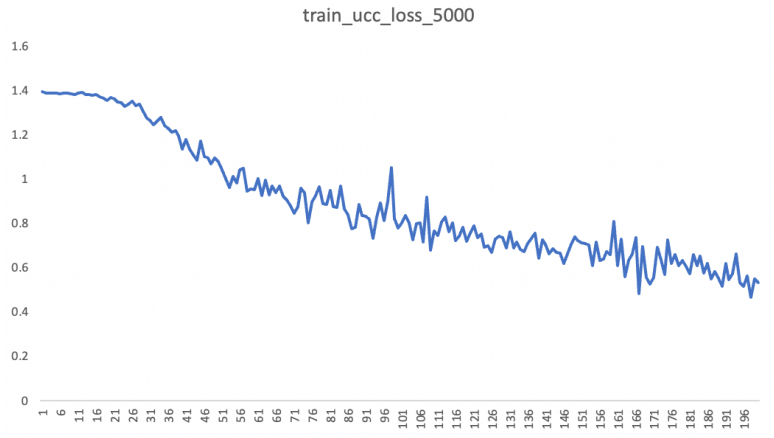
```
x_train shape: (16671, 28, 28, 1)
16671 train samples
3329 val samples
digit0:0, num_train:1662, num_val:332
digit1:1, num_train:1901, num_val:380
digit2:2, num_train:1608, num_val:321
digit3:3, num_train:1730, num_val:346
digit4:4, num_train:1621, num_val:324
digit5:5, num_train:1480, num_val:295
digit6:6, num_train:1643, num_val:328
digit7:7, num_train:1745, num_val:348
digit8:8, num_train:1602, num_val:320
digit9:9, num_train:1679, num_val:335
```



Seems like over fitting.

3. 5000 for training ucc_acc=0.900

```
x_train shape: (4171, 28, 28, 1)
4171 train samples
829 val samples
digit0:0, num_train:400, num_val:79
digit1:1, num_train:470, num_val:93
digit2:2, num_train:407, num_val:81
digit3:3, num_train:411, num_val:82
digit4:4, num_train:446, num_val:89
digit5:5, num_train:362, num_val:72
digit6:6, num_train:418, num_val:83
digit7:7, num_train:459, num_val:91
digit8:8, num_train:385, num_val:77
digit9:9, num_train:413, num_val:82
```



No overfitting util now, not sure for more steps.

4. 500 for training

Unable to get the result since small number of training data.

Hypothetically, if all hyperparameters were to be held constant, then more data means more steps along the gradient at the same learning rate, which could indeed overfit more easily. And observed the figure above, it's true that more training data means easier to overfit, but using more data makes the result more accurate.

In the proposed paper (Oner, Lee, & Sung, 2020), they implemented the model that can predict the number of unique classes in a package is learning patterns in the data that can be used to cluster the data into classes. I think the original model is accurate enough, however, it is time consuming because they default trained a million steps. To shorten the training time, I think some image augmentation can be applied. Using this method can both increase the accuracy and decrease the training time.

Reference

Oner, M. U., Lee, H., & Sung, K. W. (2020). WEAKLY SUPERVISED CLUSTERING BY EXPLOITING UNIQUE CLASS COUNT. *ICLR*.