Stacked Capsule Autoencoders

Kobee Raveendran

January 26, 2020

1 Summary

In this paper, the authors propose a new, unsupervised method of reasoning about an object that takes advantage of the geometric organization of its components. This has shown to be powerful for classification in this paper, but I don't doubt that it can be extended effectively to other tasks. Instead of purely translational variance, modeling relationships between the object and its parts also provides viewpoint invariance, which can enable this method to be more robust to changes in viewpoint for many tasks. Their method, the stacked capsule autoencoder, is composed of two primary parts: the Part Capsule Autoencoder (PCAE) and Object Capsule Autoencoder (OCAE). The PCAE serves to segment inputs into their parts and respective poses, after which the parts are reconstructed. The OCAE then attempts to re-organize poses into the whole object, in the process learning the geometric structure of the object, as it has developed a relation between the object-viewer-relationship and the object-part-relationship.

2 Good points

Something I found incredibly impressive about this paper was the fact that the method is unsupervised, yet nearly outperforms supervised methods. Although I can't speak for current performance of cutting edge models on these tasks, in my experience training CNNs on MNIST, supervised models tended to float around 98 or 99% accuracy. This method can match or even exceed that, which is impressive given that it isn't given labels during training.

3 Weak points

Something I found lacking about the paper's method was the need for different template structures dependent on the dataset used, something that is uncommon in other methods, which makes me question the flexbility of this method. For instance, the authors change the template dimensions used when testing on MNIST, CIFAR10, and SVHN datasets, which confused me, as usually the input is adjusted to fit the model, not the other way around, unless I am misunderstanding something about the authors' method. If differing input traits are the cause for variances in structure, why are the inputs not simply adjusted to fit the model? This is commonly done with other things such as resizing input dimensions or converting images to grayscale, so is there a reason it cannot be done here?

4 Questions

Something that irked me when viewing this initially was how this method would perform on objects that lack modularity. For digits and more complex objects, parts can easily be discerned since there are many, but what about extremely simple objects that have little to no parts? Are parts artificially generated for the purposes of recombining later in the OCAE or can the method perform just fine even without significant deconstruction? How **basic** can a part be?

5 Ideas

Though I can't claim to fully understand the inner workings of the authors' method enough to propose ideas for the model directly, I am curious to see a tweaked version of this method used for semantic segmentation. Since this method can detect parts of an object and successfully associate it to the whole object, it makes me think such a form of representation learning would make models that use this method incredibly efficient for segmentation, especially given that the representations can be learned in an unsupervised manner.

6 Bonus - Performance of SCAE on MNIST dataset

I have trained a model using the authors method (from their original Github repository) on the MNIST dataset (non-coupled). I trained for 300,000 training steps on my NVIDIA GeForce RTX 2070, and the entire process took approximately 7 to 8 hours. Below are classification accuracies of the model when run through the evaluation script.

```
Collecting: 312/312
Collecting: 1875/1875
Linear classification accuracy:
        train: prior=0.9847, posterior=0.9866
        valid: prior=0.9864, posterior=0.9876
Bipartite matching classification accuracy:
        posterior_pres: train_acc=0.9837, valid_acc=0.9858
        prior_pres: train_acc=0.9825, valid_acc=0.9846
Savign TSNE plot at "stacked_capsule_autoencoders/checkpoints/mnist/tsne.png"
```

These results surprised me, because they actually approximately match the accuracy scores I'd achieved when training **supervised** CNNs on MNIST. Although this unsupervised method did take much longer to train than what I remember of my supervised method, it is still very impressive, and worth the extra training time when put into use in the wild.

Additionally, below is the TSNE plot generated during evaluation, to give an idea of the class distribution and classfication accuracy on the MNIST dataset. As can be seen, there are some mismatches, but the fact that the digit class clusters are so well-defined and easily separable from each other is a victory for SCAE.