Object-Oriented Deep Learning and Generative Adversarial Networks

Qianli Liao, Zhuoqiao Hong

Abstract

Deep learning is a field that has been mainly driven by connectionist models like neural networks, characterized by layered processing of distributed, sub-symbolic and statistical features. However, human high-level thoughts appear to be highly symbolic, focusing on objects and relations. To bridge the gap between perception and symbols, a series of models on "Object Oriented Deep Learning" was proposed [9, 8, 7]. In this project we further explore this class of models. We implement a generative version of OODL that can generate images instead of performing object recognition, in a similar way to Generative Adversarial Networks (GANs). In comparison to conventional "feature-oriented" deep learning, OODL naturally handles properties of objects by incorporating them as fields. It offers exact equivariance [8] to translation, rotation and scaling. When implementing it as a generative model, one should be able to precisely control such geometric properties of the generated objects.

1 Introduction

Deep learning is characterized by its typical layered processing of distributed, sub-symbolic and statistical features. It has achieved outstanding performance on many sensory processing tasks like computer vision, image recognition and speech recognition. Most of these tasks can be solved by highly statistical pattern recognition approaches.

Nevertheless, many tasks that involve human high-level cognition require a different solution from simple statistical pattern recognition. For example, when processing a region of the visual field, one cannot simply treat it like a statistical blob of features (as in current deep learning, which we call "feature-oriented" processing). High-level cognition requires separating objects from background and each other, knowing feature/property bindings (e.g., property 1 (red) belongs to object A and property 2 (tilted 30 degrees) belongs to object B). These properties cannot be achieved by existing building blocks in the current deep learning arsenal. To address such limitations, Object-Oriented Deep Learning (OODL) was proposed [9, 8, 7]. Generally speaking, Object-Oriented Deep Learning is a novel computational paradigm of deep learning that adopts interpretable "objects/symbols" as a basic representational atom [9, 8, 7] instead of N-dimensional tensors.

In this project, we further explore and expand the class of models in OODL. One open problem in current machine learning is the development of disentangled representations, often discussed in the literature of Generative Adversarial Networks (GANs) [4] and Variational Autoencoders [6]. One goal of disentangled representation is to extract transformations (e.g., 2D/3D rotation) of objects and keep information of different transformations disentangled from each other. Many attempts have been made in learning such disentangled representations (To list a few, [12, 3])

Different from the above data-driven approaches to disentangled representations, OODL proposes an approach called "symbolic disentanglement". That is, common transformations like translation, rotation and scaling are encoded as intrinsic properties of an object. This approach has the advantage of being data-independent — it can generalize new data distributions without extensive training. Furthermore, such "symbolic disentanglement" can be combined with existing data-driven

Figure 1: A simple illustration of our approach (B), compared with traditional DCGAN (A).

disentanglement approaches, making the learning-based component more focused on object-specific transformations like deformation, shading and styles.

To that end, we implement in this project an Object-Oriented Generative Adversarial Network (OOGAN), as shown in Figure 1. Unlike a traditional GAN, this model should allow the user to precisely control the pose (position, rotation and size) of the generated object (and even its parts). This will be a good demonstration of the exact equivariance and disentanglement property [8] of the object-oriented network. It provides a better visualization of this property compared to the case of a discriminative task like image classification.

We used Matlab and custom NVIDIA CUDA code to implement this model.

2 Methods

2.1 GAN

Generative Adversarial Network (GAN) [4] is a deep learning framework for estimating generative models through an adversarial process. It simultaneously trains two models: a generative model G to capture the data distribution and a discriminative model D to estimate the probability that a sample came from the training data (Goodfellow et al, 2014). G and D play a two-player minimax game with value function $V(G,D)$:

$$\min_{\mathbf{D}} \max_{\mathbf{G}} V(\mathbf{D}, \mathbf{G}) = \mathbb{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})} [\log \mathbf{D}(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})} [\log (1 - \mathbf{D}(\mathbf{G}(\mathbf{z})))]$$

In other words, the game is between two players, a generator and a discriminator. The discriminator aims at distinguishing between samples from the generator and the true distribution. and the goal of generator is to trick the discriminator by generating samples that resemble as much as possible the ground-truth distribution.

One limitation of a convolutional GAN is that it cannot explicitly represent and disentangle common transformations of the object, such as translation, rotation and scaling. Although it is possible for a convolutional GAN to learn such transformations from data [10], there is no guarantee for such learned representations to generalize to new data distributions.

For training Deep Convolutional Generative Adversarial Network (DCGAN) [11], we used an open-source implementation from <https://github.com/sunghbae/dcgan-matconvnet>. The training meta parameters are all the same as the default settings in this package.

Figure 2: Traditional DCGAN v.s. our OOGAN on MNIST. (B) and (C) are extracted from different epochs. Current implementation of OOGAN has somewhat less diversity in results, which could be due to undiscovered bugs in the code of model or training procedure. In principle, OOGAN should perform at least as well as DCGAN since OOGAN can be shown to capture DCGAN as a special case [9]

2.2 GAN with Object-oriented Generator

In our Object-oriented GAN (OOGAN), we replace the generator with the object-oriented network proposed in [8] with some modifications. In [8], the object-oriented network is a discriminator, which processes the input in a bottom-up fashion. To convert it to a generator, one needs to perform something similar to a "deconvolution" (in the sense that more low-level objects/features are generated from a few high-level objects). To achieve this, we shrink the binding radius in the binding layer so that finer spatial configuration is obtained while passing through more voting and binding layers. Finally, when the objects generated spatially span the desired image size, we can convert all objects into an image by discretizing them into a pixel grid. As a side note, we also found that it improves training when we limit the relative orientation difference from one object and the objects voted by it to $[-45, 45]$ degrees.

The training meta parameters are the same as DCGAN. We used a smaller minibatch (100) due to large memory requirement of our OOGAN implementation. The object-oriented network is the same as in [8] (excluding above mentioned modifications).

3 Results

3.1 DCGAN on MNIST

As a baseline, we trained a Deep Convolutional Generative Adversarial Network (DCGAN) and tested it on MNIST, as shown in Figure 2.

3.2 Object-oriented GAN

A preliminary result of OOGAN on MNIST is shown in Figure 2. We show that OOGAN can learn to generate reasonably looking handwritten digits. Current implementation is a few times slower than convolutional GAN, but we plan to speed it up with better optimized NVIDIA CUDA code in future work.

3.3 Control Object Poses at Hidden Layers of OOGAN

We show two properties of OOGAN: (a) when specifying a pose on the top layer of the generator, it can generate an object with exactly that global pose, as shown in Figure 3. (b) when changing the poses of some objects in the intermediate layers of the generator, we can control the poses of the parts of the generated object, as shown in Figure 4.

Figure 3: Control object pose by modifying the pose information in the first layer of the object-oriented generator in OOGAN. The top row is the generated object. The bottom row is the pose of the object (fed together with noise/seed into the generator). We can control both scale and orientation of the generated object.

Figure 4: Control object pose by modifying the pose information in the second layer of the object-oriented generator in OOGAN. The top row is the generated object. The bottom row shows the pose of the parts of the object. The pose of parts are originally generated by the first layer and then we add controlled noise to them. With this approach, we can induce local deformation of the generated image in a controlled way.

3.4 Movies of Deforming Objects as We Perturb Poses

With the approach induced in the above section, as we add small changes to object poses at different hidden layers, we render the generated images into small clips of movies shown in Figure 5 (all clips can be found in <https://bit.ly/2SDhHoV>). These clips provide a good visualization of the transformation disentanglement property of OODL.

4 Related Work

One piece of the related work is the Capsule Networks[5] by Hinton. They operate on a dense grid to detect objects at fixed locations, while OODL does not use any grid structure. In addition, the generalized pose in Capsule Networks is not the typical pose that we use in OODL, but rather a combination of pose and appearance (deformation, color, lighting, etc). Another line of work is a class of models related to "neural physics engines" where some form of object representation is used as their inputs [1][13][2]. They are different from OODL in two main ways: 1) They build high-level symbols from neural infrastructure, instead of symbolizing its most basic building blocks and 2) they only aim at high-level processing (e.g., physics simulation, reinforcement learning, etc.), instead of replacing traditional Convolutional Neural Networks.

Figure 5: We rendered many movies of deforming objects in <https://bit.ly/2SDhHoV>. Interestingly, when we allow the objects in the intermediate layer to perform a "Brownian motion" (<https://bit.ly/2rtePJ2>), the whole object "dissolves" since the part-whole relationship is no longer maintained at some point as all parts move independently.

5 Conclusions

In this project, we present Object-oriented Generative Adversarial Netowkrs (OOGAN), a novel deep learning model that can precisely control the pose of the generated object (and even its parts). It provides a new perspective on the open problem of disentangled representations.

The object-oriented deep learning framework provides a novel solution to the issues of interpretability and AI safety. Unlike most of the recent work, which trying to draw some interpretation from the distributed code of learned deep neural networks, this framework eliminates as much as possible non-symbolic components from neural networks. It also potentially leads to better efficiency, sparsity, flexibility, and new options of hardware accelerations.

References

- [1] Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. In *Advances in neural information processing systems*, pages 4502–4510, 2016.
- [2] Michael B Chang, Tomer Ullman, Antonio Torralba, and Joshua B Tenenbaum. A compositional object-based approach to learning physical dynamics. *arXiv preprint arXiv:1612.00341*, 2016.
- [3] Xi Chen, Yan Duan, Rein Houthoofd, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in neural information processing systems*, pages 2172–2180, 2016.
- [4] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, Warde-Farley David, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. Technical report, Departement d’informatique et de recherche operationnelle Universite de Montreal, 2014.
- [5] Geoffrey E Hinton, Alex Krizhevsky, and Sida D Wang. Transforming auto-encoders. In *International Conference on Artificial Neural Networks*, pages 44–51. Springer, 2011.
- [6] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- [7] Qianli Liao and Tomaso Poggio. 3d object-oriented learning: An end-to-end transformation-disentangled 3d representation. Technical report, 2017.
- [8] Qianli Liao and Tomaso Poggio. Exact equivariance, disentanglement and invariance of transformations. Technical report, 2017.
- [9] Qianli Liao and Tomaso Poggio. Object-oriented deep learning. Technical report, Center for Brains, Minds and Machines (CBMM), 2017.
- [10] Chen-Hsuan Lin, Ersin Yumer, Oliver Wang, Eli Shechtman, and Simon Lucey. St-gan: Spatial transformer generative adversarial networks for image compositing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9455–9464, 2018.

- [11] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.
- [12] Luan Tran, Xi Yin, and Xiaoming Liu. Disentangled representation learning gan for pose-invariant face recognition. 2017.
- [13] Nicholas Watters, Andrea Tacchetti, Theophane Weber, Razvan Pascanu, Peter Battaglia, and Daniel Zoran. Visual interaction networks. *arXiv preprint arXiv:1706.01433*, 2017.