Assignment 2
COL775: Deep Learning. Semester II, 2023-2024.
Due Date: May 8, 2024

April 5, 2024

The dataset for this assignment can be found from `/scratch/cse/dual/cs5190448/A2_dataset` on HPC.

PART 1

1 Object-Centric Learning With Slot Attention

Neural models are considered as black box methods due to their poor interoperability. Recently it has been found that having object centric representation in latent space not only improves the interpretability but also improves the performance of models. Slot attention is one of the method to learn object centric representations from images. In this assigment you will implement slot attention from scratch.

1.1 Slot Attention

Slot attention was proposed in the paper [1]. And is defined as follows: "The Slot Attention module maps from a set of N input feature vectors to a set of K output vectors that are refer to as slots. Each vector in this output set can, for example, describe an object or an entity in the input. Slot Attention uses an iterative attention mechanism to map from its inputs to the slots. Slots are initialized at random and thereafter refined at each iteration t = 1 . . . T to bind to a particular part (or grouping) of the input features." Read the original slot attention paper [1] for more information. A pseducode for algorithm is shown in figure 1.

- More formally given image $x \in \mathbf{R}^{H \times W \times 3}$, a CNN encoder $\mathcal{E}$ gives feature maps $\mathcal{E}(x) = z \in \mathbf{R}^{\sqrt{N} \times \sqrt{N} \times D_{inputs}}$.

Algorithm 1 Slot Attention module. The input is a set of N vectors of dimension $D_{\texttt{inputs}}$ which is mapped to a set of K slots of dimension $D_{\texttt{slots}}$. We initialize the slots by sampling their initial values as independent samples from a Gaussian distribution with shared, learnable parameters $\mu \in \mathbb{R}^{D_{\texttt{slots}}}$ and $\sigma \in \mathbb{R}^{D_{\texttt{slots}}}$. In our experiments we set the number of iterations to $T = 3$.

Input: $\texttt{inputs} \in \mathbb{R}^{N \times D_{\texttt{inputs}}}$, $\texttt{slots} \sim \mathcal{N}(\mu,\ \mathrm{diag}(\sigma)) \in \mathbb{R}^{K \times D_{\texttt{slots}}}$
Layer params: $k,\ q,\ v$: linear projections for attention; `GRU`; `MLP`; `LayerNorm` (x3)
  $\texttt{inputs} = \texttt{LayerNorm}\,(\texttt{inputs})$
  for $t = 0 \ldots T$
    $\texttt{slots\_prev} = \texttt{slots}$
    $\texttt{slots} = \texttt{LayerNorm}\,(\texttt{slots})$
    $\texttt{attn} = \texttt{Softmax}\,(\frac{1}{\sqrt{D}} k(\texttt{inputs}) \cdot q(\texttt{slots})^T,\ \texttt{axis='slots'})$ # norm. over slots
    $\texttt{updates} = \texttt{WeightedMean}\,(\texttt{weights=attn} + \epsilon,\ \texttt{values=}v(\texttt{inputs}))$ # aggregate
    $\texttt{slots} = \texttt{GRU}\,(\texttt{state=slots\_prev},\ \texttt{inputs=updates})$ # GRU update (per slot)
    $\texttt{slots} \mathrel{+}= \texttt{MLP}\,(\texttt{LayerNorm}\,(\texttt{slots}))$ # optional residual MLP (per slot)
  return `slots`

Figure 1: Pseudo Code for slot attention

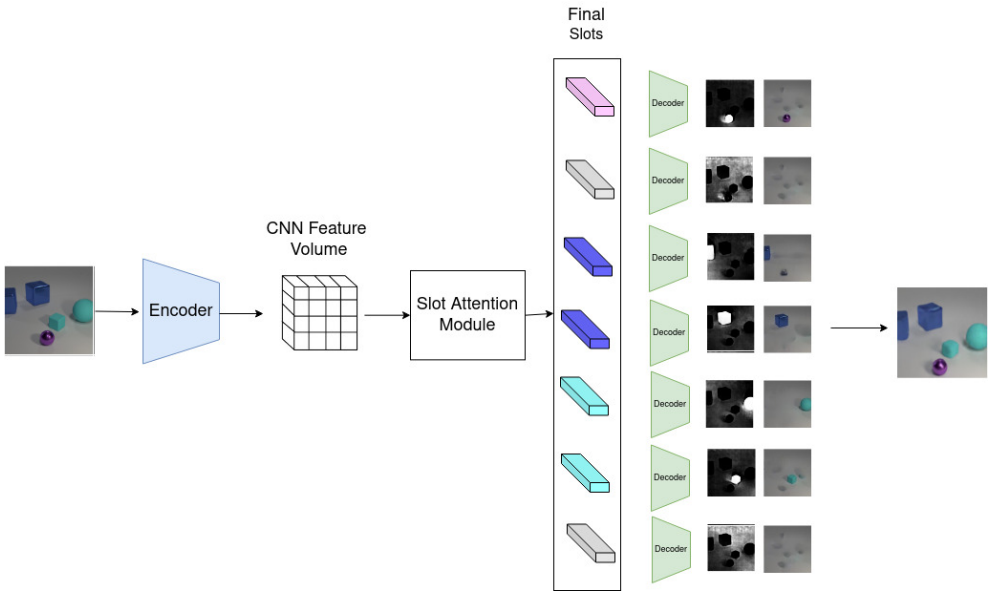

Figure 2: High level diagram

- A 2D positional encoding $P \in \mathbf{R}^{\sqrt{N} \times \sqrt{N} \times D_{inputs}}$ is added to z,

$$z' = z + P$$

- z' is flattened spatially to get N input vectors. These are passed through `MLP` to get final $\texttt{inputs} \in \mathbf{R}^{N \times D_{inputs}}$

$$\texttt{inputs} = \texttt{MLP}(\texttt{Flatten}(z'))$$

- Then K slot vectors, $\texttt{slots} \in \mathbf{R}^{K \times D_{slots}}$ are initialized by sampling them from normal distibution $\mathcal{N}(\mu, \text{diag}(\sigma))$, where both μ and σ are learnable.
- Then `slots` attend to `inputs` itertively for T times and updated `slots` at the end of each iteration. For detailed steps refer to pseudo code 1.
- Each of K vectors from `slots` are decoded with **Spatial Broadcast Decoder** [3], $\mathcal{D}$ to produce 4 channeled image. Denote $\texttt{output}_i = \mathcal{D}(\texttt{slots}[i])$, note that $\texttt{output}_i \in \mathbf{R}^{H \times W \times 4}$
- To get the final image, first K masks are generated one for each slot by taking softmax for each pixel of first channel, across slots.

$$\texttt{masks}_i = \texttt{Softmax}(\{\texttt{output}_j[:, :, 0]\}_{j=1}^{K})_i$$

- The final reconstructed image is obtained as

$$\hat{x} = \sum_{i=1}^{K} \texttt{masks}_i \cdot \texttt{content}_i$$

  where $\texttt{content}_i = \texttt{output}_i[:, :, 1:]$

These are spatially flattened to get N input vectors $\texttt{inputs} \in \mathbf{R}^{N \times D_{inputs}}$.

1.2 Experiments

You're provided with a sub-sampled CLEVRTex dataset which comprises of synthetic images of objects with different attributes, locations and properties. Train the Slot-Attention based network as shown in 2 on this data and visualize the generated masks and the reconstructed images. **We recommend using similar hyper-parameters to [1] [4]. However you're free to experiment with other values as well.** Use number of slots $K = 11$ for this experiment.

- Report the Adjusted Rand Index (ARI) score between the ground-truth and predicted object segmentation masks on the val split.
- **Compositional Generation**: Create a slot library using the training data, apply K-means clustering with K being the number of slots in the trained model. Sample new slots from each of the cluster to generate an image with compositional attributes. Generate N_{val} such images, where N_{val} is number of validation images and report the clean-fid [2] metric using the validation images as ground truth.

PART 2

Coming Soon

References

[1] Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention, 2020.

[2] Gaurav Parmar, Richard Zhang, and Jun-Yan Zhu. On aliased resizing and surprising subtleties in gan evaluation. In *CVPR*, 2022.

[3] Nicholas Watters, Loic Matthey, Christopher P. Burgess, and Alexander Lerchner. Spatial broadcast decoder: A simple architecture for learning disentangled representations in vaes, 2019.

[4] Ziyi Wu, Jingyu Hu, Wuyue Lu, Igor Gilitschenski, and Animesh Garg. Slotdiffusion: Object-centric generative modeling with diffusion models, 2023.