6.869 Final Project - PVAE

Fareed Sheriff Sameer Pai

Abstract

VAEs, or variational autoencoders, are autoencoders that explicitly learn the distribution of the input image space rather than assuming no prior information about the distribution. This allows it to classify similar samples close to each other in the latent space's distribution. VAEs classically assume the latent space is normally distributed, though many distribution priors work, and they encode this assumption through a K-L divergence term in the loss function. While VAEs learn the distribution of the latent space and naturally make each dimension in the latent space as disjoint from the others as possible, they do not group together similar features — the image space feature represented by one unit of the representation layer does not necessarily have high correlation with the feature represented by a neighboring unit of the representation layer. This makes it difficult to interpret VAEs since the representation layer is not structured in a way that is easy for humans to parse.

We aim to make a more interpretable VAE by partitioning the representation layer into disjoint sets of units. Partitioning the representation layer into disjoint sets of interconnected units yields a prior that features of the input space to this new VAE, which we call a partition VAE or PVAE, are grouped together by correlation — for example, if our image space were the space of all ping ping game images (a somewhat complex image space we use to test our architecture) then we would hope the partitions in the representation layer each learned some large feature of the image like the characteristics of the ping pong table or the characteristics and position of the players or the ball. We also add to the PVAE a cost-saving measure: subresolution. Because we do not have access to GPU training environments for long periods of time and Google Colab Pro costs money, we attempt to decrease the complexity of the PVAE by outputting an image with dimensions scaled down from the input image by a constant factor, thus forcing the model to output a smaller version of the image. We then increase the resolution to calculate loss and train by interpolating through neighboring pixels. We train a tuned PVAE on MNIST and Sports10 to test its effectiveness.

1. Introduction

Variational autoencoders [CITE] are a type of neural network that attempts to learn the distribution of the input space while imposing the prior that the input space is normally distributed. The neural network (a multi-layer perceptron) is autoencoder-like in that the hidden layer learns the input space as each input is passed through the network. The prior that the hidden layer, the representation layer, is normally distributed is maintained by introducing a K-L divergence term in the loss function, which itself includes some measure of difference between the input and output images (mean squared error or binary cross-entropy, for example, depending on what is known about the values over which the input ranges). VAEs represent the latent space as a distribution by learning the mean and log variance of the latent space, which allows us to sample from the latent space to produce viable entries in the input space. While effective at encoding the input space in the usually more compact latent space, VAEs are classically difficult to interpret. Specifically, the latent space has no predefined structure beyond the prior that it is normally distributed.

Game representation [CITE] is a long-standing field of interest that focuses on finding efficient representations of games. The importance of this problem is visible in both the real world and the machine learning world: efficient game representations can increase the efficiency and potentially accuracy of intelligent game agents, and compact game representations are often easier for humans to understand than more dense representations. Furthermore, efficient game representations can be used in conjunction with common game representations to simplify common representations without losing too much information, which is a good trade-off for both humans and machines.

Various architectures have been designed to increase the interpretability of the VAE, discussed in more detail in the next section. The PVAE seeks to increase interpretability by grouping together similar features (features of the input space with high correlation). The end result is a partitioned representation layer each of whose partitions contains a set of correlated units. The representation layer is partitioned into groups of neurons that are each connected to a neural network whose output is of the same size as the size of each group of neurons. This puts a large portion of the PVAE

into disjoint sets of neurons that together form the representation, which pushes the representation to be partitioned such that individual partitions of the representations represent features of the input space that correlate with each other and features that do not correlate with each other that much are generally not represented by neurons in different partitions. The partition feature of the PVAE is especially useful for game representations because based on the number of partitions, we can isolate individual important components of an image representation of a game. In ping pong, for example, a given partition could encode the characteristics of the ping pong table.

2. Related Work

Significant research has been done both on game representations and making VAEs more interpretable. Some examples of more interpretable VAEs are oi-VAE (output interpretable VAE) [CITE] and PI-VAE (physics-informed VAE) [CITE]. oi-VAEs attempt to disentangle latent variables by penalizing for multiple variables sharing data. This attempts to keep units in the representation layer as disjoint as possible in terms of what they represent. In general terms, PI-VAE attempts to make the method of training interpretable by modeling the VAE through physics processes, making the model a subset of a field of ML known as physics-based ML. While both seek to make the VAE more interpretable in different ways, it should be noted that oi-VAE, which is closest to our definition of interpretability (making the representation layer more interpretable), acts specifically on latent variables by decreasing correlation between variables. In contrast, PVAE encourages information-sharing between variables that already have high correlation and prevents interaction between latent variables within different partitions and therefore lower correlation. PVAE therefore aims to be more interpretable over partitions rather than latent variables themselves.

Various articles have been published on general game representation, including a paper on creating representations of video games that are edition- and graphics-invariant [1], an article on contrastive learning between games based on genre that aimed to learn representations of games by genre rather than specifically by game [2]. Furthermore, play2vec, a sports representation robust to noise, is a sports representation model that attempts to compare sports plays by comparing the similarity of their representations [3]. While each of these models learn representations that are invariant over some category, they do not explicitly create a game representation optimized for individual games. Because they are supposed to learn invariant representations, efficient representation is not the main goal of these papers but more a secondary goal if at all. We apply PVAE to images of ping pong matches to evaluate our primary goal of creating a compact and accurate but interpretable representation of the ping pong game image space, contrasting in purpose with the models mentioned above.

3. Methods

We describe how we define and test our PVAE in this section. PVAE is implemented in Python using the PyTorch library. We test PVAE on MNIST and Sports10 with a variety of parameters to analyze how well the representation encodes samples. Finally, we examine sample representations and examine the effects of modifying the representation partition-by-partition.

A PVAE is a VAE with three parts: an encoder, a partition, and a decoder. The encoder consists of a set of convolution layers as defined by the user followed by a Flatten layer and a Linear layer to share a small bit of information between units and to allow partitions in the representation to assign information to partitions based on the information's dimensionality. The partition layer consists of ANNs that preserve partition size, thus learning features of the representation, two sets for each partition to yield the mean and log variance. We sample from the mean and log variance layers and pass this through the decoder. Finally, the decoder consists of Linear layers and alternating conv and deconv layers to rebuild an output image from the representation. The loss function is a weighted sum of reconstruction loss and K-L divergence, the former to ensure the output resembles the input and the latter to maintain the prior that the representation is normally distributed. We weight the reconstruction loss (MSE loss) multiple magnitudes more than the K-L divergence when testing PVAE because the normal prior is far stronger than the reconstruction prior. We test representations of size k on MNIST and Sports10 by comparing the losses of different partitions of k with h.

Because we do not have free access to GPUs, we take measures to decrease the complexity of training our PVAE. We do this by making the PVAE yield an output image smaller than the input image, then upscale the output to the resolution of the input. In this way, the representation does not need to be as exact and the decoder is less complex. Finally, we slightly perturb the train data by inserting low levels of random noise into training samples. Examples of different components of a PVAE are displayed below.

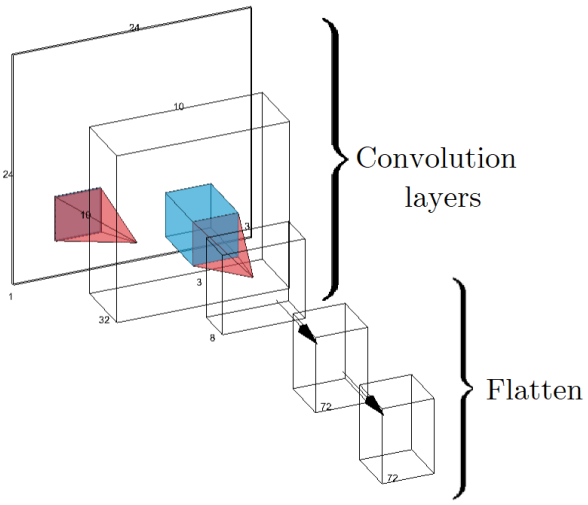
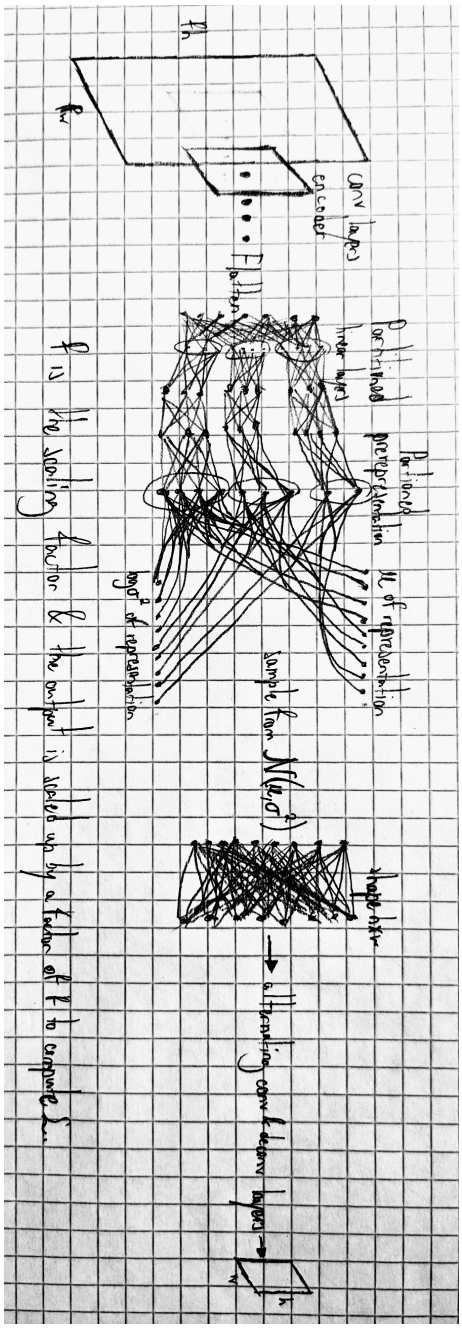


Figure 1. Sample PVAE encoder for MNIST



4. Code, Progress, & Data

We have completely written the code for the Partition-VAE, and we have been testing the model against sample datasets to see how well it performs. In testing our model against MNIST, we have found that after only around five to ten epochs the VAE learns a representation that can reliably be decoded into an image that looks very similar to the input. Furthermore, we have not seen any evidence of overfitting as the validation loss is consistently below the training error, and the loss has steadily been decreasing, implying

that the model still has a lot of learning to do. Alongside our testing, we have made some practical adjustments to our code to ensure smooth training. For example, we have adjusted our loss function by weighting the reconstruction loss portion greater than the K-L divergence because the K-L divergence term was originally affecting the loss so much that all of the outputs of the model looked largely the same regardless of input. By decreasing the weight of the K-L divergence relative to the reconstruction loss, we decrease the strength of the prior that the representation must be normal, allowing the output to be more diverse and closer to the input.

We have also downloaded and viewed the dataset we plan on using in our final project. Our primary dataset will be the *Sports10* dataset introduced by [2]. This is a dataset consisting of over 100K images of video games in ten different sports genres, including table tennis, volleyball, and soccer. This dataset further classifies each image into three categories: retro, modern, and photorealistic. We will use the table tennis subset of this dataset in our project, learning a representation from it and using the representation as described in the problem statement.

5. Individual Contributions — Fareed

Fareed Sheriff, one of the team members who created the PVAE, came up with the architecture of the PVAE, wrote the code for the PVAE, and wrote the abstract, introduction, related works, and methods sections. Sameer Pai, the other team member, wrote the code & progress section of the paper, revised the paper, and tested and analyzed the PVAE's performance on MNIST and Sports10

References

[1] Jin Ha Lee Rachel Ivy Clarke Jacob Jett, Simone Sacchi. A conceptual model for video games and interactive media, May 2016.

[2] Chintan Trivedi, Antonios Liapis, and Georgios N. Yannakakis. Contrastive learning of generalized game representations, 2021.

[3] Zheng Wang, Cheng Long, Gao Cong, and Ce Ju. Effective and efficient sports play retrieval with deep representation learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 499–509, New York, NY, USA, 2019. Association for Computing Machinery.