Recursively Trained Autoencoders

William Ray Daniel Hannon
wray@umass.edu drhannon@umass.edu
University of Massachusetts Amherst

November 5, 2020

1 Introduction

Our Project attempts to improve upon the quality of images created by current autoencoders. From work done in GANS, we can see that it is possible to create crisp detailed images entirely from deep learning models. The problem we are addressing is that the current autoencoder produced images are blurry and lose the detail of the original image. Our approach to solving this problem is to train autoencoders recursively. The loss would be calculated at each intermediate layer to recover better the data lost in the encoding process.

2 Problem Statement

The problem that was seeking to address is that the images produced by a trained autoencoder come out blurry and of a lower quality than images produced by other GAN-based methods. The drop in quality is caused by the autoencoder losing information from the original image at every layer. In the learning process, the loss function incentivizes the model to create outputs strictly similar to the input and not realistic-looking images. The dataset we will be using is the UMass labeled faces in the wild database. We have chosen this dataset because faces lend themselves to autoencoders because they have very similar structures while having distinct features. We expect the generated faces in the dataset to be crisper and have more fine detail than the faces generated by a stacked autoencoder. To evaluate the images' quality, we will compare the facial image reconstructed at specific points in the encoded latent space between a stacked model and the recursive model. Faces will be compared and evaluated on how clear and detailed the image is and how much it resembles a face.

3 Technical Approach

We attempt to solve this problem and produce these clearer images by changing a stacked autoencoder's structure during training and training it in a more 'recursive' way. Instead of training the autoencoder as one model with one input (the image) and one output (the reconstruction), we train several models that share the same layers. Each layer in the decoder is an output layer that attempts to learn what the equivalent layer in the encoder output. For example, in Figure graph, we would train three models. The first model would have layers: [Image, Hidden 1, Reconstruction]. The weights in 'Image' are frozen, and the 'Reconstruction' layer attempts to recreate what the 'Image' layer output, given the output of 'Hidden 1'. Similarly, the second model would have layers: [Image, Hidden 1, Hidden 2, Hidden 5]. The weights in 'Image' and 'Hidden 1' are frozen, and 'Hidden 5' attempts to recreate what 'Hidden 1' output given the output of 'Hidden 2'. The third model follows the same pattern of freezing weights in the encoder above the decoder layer, while the decoder attempts to recreate the second-to-last output of the encoder, given the last output of the encoder. We hypothesize that this training will recover more of the fine details lost in the encoding process for two reasons. First, we believe that with each decoder layer attempting to re-

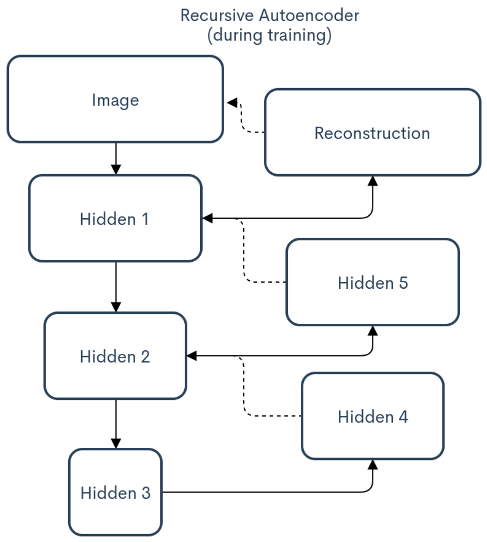

Figure 1: Recursive Autoencoder during training. Boxes represent input and output tensors. Solid arrows represent convolutional layers. Dotted arrows represent mean squared error comparisons. For example The Reconstruction tensor is being compared to the Image tensor and there is a loss proportional to squared difference between those two tensors.

produce its corresponding layer's input, the decoder layers at shallower levels will be more incentivized to learn detailed feature reconstruction for things like eyes, ears, or hair, rather than high-level features like shape or color. The second reason we expect this different training system to improve the clarity of produced images is that the encoders at each layer will learn encodings that benefit the corresponding decoder layer's feature decoding task. We think that in current autoencoders, the encoding layers each learn how to do the best job at fully encoding an image so that it can be reconstructed from its completely encoded representation. We expect the encoding layers to encode information in a way that preserves finer detail at the expense of larger characteristics. When this novel autoencoder reconstructs people's images, the reconstruction may look less like the original person. However, the reconstruction may look crisper and more like an image of a person that could exist. Every autoencoder loses information; this approach attempts to trade higher-level information for lower-level information to achieve higher quality images. With these two reasons working together, we believe our system will produce much clearer detailed images than stacked autoencoders.

4 Preliminary Results

Up to this point, we have trained a stacked autoencoder and a basic version of the recursive autoencoder with the architecture shown in Figure architecture. We have trained both models with the same batches of faces from the database. We have not fined tuned the parameters or the models' exact architecture, but they are both capable of reconstructing faces. In Figure faces we have a diagram showing the original images put into each model and the model's attempt to reconstruct it. The stacked model's output is blurry, but the face is visible in the reconstruction. Our Recursive model produces sharper images than the stacked model's images, but they still lack the original image's fine detail and appear to have discolored patches.

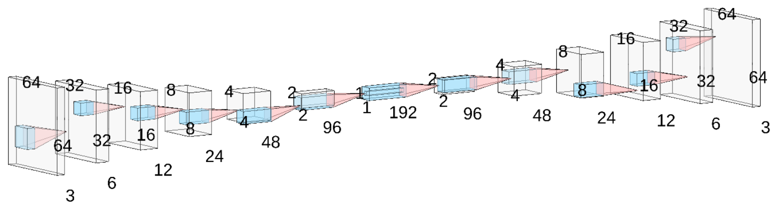

Figure 2: Architecture of the full autoencoder model. This figure does not show the intermediate models trained to produce layer weights for the recursive autoencoder.

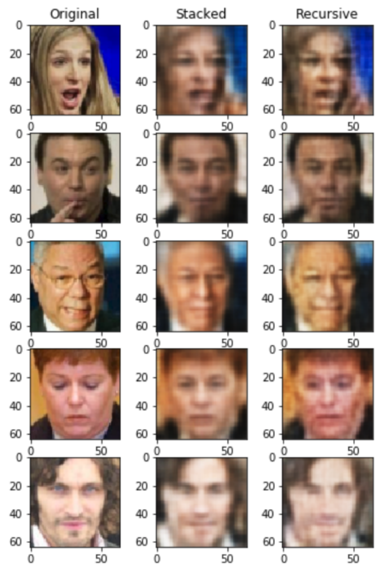

Figure 3: Faces reconstructed using the stacked and recursive autoencoders. The autoencoders were trained on the same data and have the same architecture but the recursive autoencoder was trained in the novel way.