Efficient Mask Inpainting via Convolutional and Generative Adversarial Network Approaches

Ryan M. Beckwith
Department of Computer Science
Tufts University
Medford, MA 02155
ryan.beckwith@tufts.edu

Adam G. Peters
Department of Computer Science
Tufts University
Medford, MA 02155
adam.peters@tufts.edu

Abstract

Recent advances in computational power and efficiency have allowed for the training of increasingly larger and more accurate neural networks. Despite these developments, model performance is still greatly restricted by network size, and thus computational resources. Given the increasing demand for complex models and the environmental damage associated with training these models, there is a great need for network optimization. In this paper, we explore the problem of object recognition and inpainting by attempting to remove face masks from images. To solve this problem, which we call mask inpainting, we utilize the "Face Mask Lite" dataset and examine various architectures and deep learning paradigms, including convolutional and generative adversarial networks. With the former, we develop a promising data-efficient encoder-decoder model that successfully removes masks and inpaints viable facial features.

1 Introduction

Computer vision is an active and fundamental area of research in the machine learning space that deals with interpreting images and videos. Due to the large parameter space in low and high resolution images, training models on these data is often computationally arduous. The popularization of convolutional neural networks (CNNs) with AlexNet has led to more efficient and effective solutions, but robust models still require many parameters, and gathering sufficient, labeled data is often infeasible. Despite these bottlenecks, problems that were once impossible are now solvable with sufficient data, time, and resources.

Object detection and inpainting are two active areas of research in the computer vision subdomain that, like many other problems, are constrained by the aforementioned bottlenecks. Object detection is the process of identifying the boundaries of different objects in an image, and inpainting is the process of filling in an image's missing information. In this paper, we combine these two problems by attempting to remove face masks from portrait images, a hybrid problem we call **mask inpainting**. Our solution indirectly identifies the location of masks in 256×256 images, and then fills in those masks with intuited facial structures. We explore five distinct model architectures in this paper, and ultimately propose a successful model (our encoder-decoder CNN) that requires a fraction of the resources and training previously used to solve this problem.

1.1 Previous work

Existing literature in the image inpainting space is principally dominated by convolutional neural network (CNN) and generative adversarial network (GAN) approaches. Several sources [3, 5, 11] reported CNN-based architectures for general image inpainting, while facial inpainting CNN models

Preprint. Under review.

[10] were less common. One particularly relevant study [2] reported an encoder-decoder CNN solution specifically for the mask inpainting problem, which also utilized the "Face Mask Lite" dataset [8]. Similar, yet substantially more complex encoder-decoder approaches were reported in [10] and [11], thus solidifying the robustness and practicality of the encoder-decoder approach within the general context of image inpainting.

GAN-based approaches were also highly prevalent in the reported literature, with several papers proposing GAN-based architectures specifically within the domain of facial inpainting [1, 6, 9, 12]. Several of these papers present novel techniques for optimizing GAN-based approaches, such as utilizing free-form user sketches as a backbone for generated structures [12], or employing multiple loss functions for training discriminator models [6]. Evidently, it would appear that state-of-the-art facial inpainting techniques lie firmly within the realm of GAN-based approaches, while more generalized image inpainting problems are often solved with purely convolutional models.

1.2 Training Data

We used various subsets of Kaggle's "Face Mask Lite Dataset" [8] in order to train our models. These data consist of 10,000 PNGs of faces, and 10,000 corresponding images with superimposed virtual masks. All superimposed masks were added manually by the database's creator.

Notably, the images in this dataset are relatively standardized, with the vast majority representing centered, close-up portraits of white or white-passing individuals with considerable background bokeh. As such, immediate concerns may be reasonably raised regarding the quality of models trained on this dataset, which may be particularly susceptible to overfitting.

1.3 Technical Restrictions

As students, we were limited by monetary resources, so all models in this paper were trained using Google Colaboratory's free plan. The cloud computing services offered by Colab are relatively powerful, but the site frequently times out and interrupts all running tasks, which rendered training models for longer than eight hours quite difficult. We considered using Amazon Web Services, but were deterred by the pricing. Instead, we settled on progressively saving our models before Colab timed out, and then reloading and continuing training on a new Colab instance. The vast majority of our models, though, trained in under 5 hours and did not require this iterative strategy.

2 Convolutional Neural Network (CNN) Approaches

Given the immediate relevancy and success of [2] specifically within the context of mask inpainting, as well as the successes of other convolutional approaches in facial inpainting [10] and general image inpainting [3, 5, 11], we decided to initially adopt a CNN-based approach. Following the approach of [2], we decided to use the Tensorflow Keras framework to construct model architectures, employ pre-built loss functions and optimizers, and plot accuracy/loss metrics on a per-epoch basis using the Callbacks API.

In the following subsections, we chronologically present three CNN architectures that represent our preliminary attempts to solve the mask inpainting problem. All three models were trained using the Adam stochastic gradient descent optimizer, which was chosen for its computational efficiency as per [4], and all three models utilized the mean squared error (MSE) loss function.

2.1 Pooling-free approach

Our initial approach was to build a relatively shallow and purely convolutional network, which we reasoned would serve as an acceptable benchmark for a CNN-based approach. In contrast to the encoder-decoder approaches of [2] and [10], which initially reduce dimensionality with pooling layers and subsequently restore original dimensionality with up-sampling layers, we originally decided to adopt a pooling-free approach that maintains high dimensionality throughout the network. The rationale for this decision was that an encoder-decoder approach would effectively perform a lossy compression on the external features of the image (see results of [2], for example), which ideally should remain unmodified.

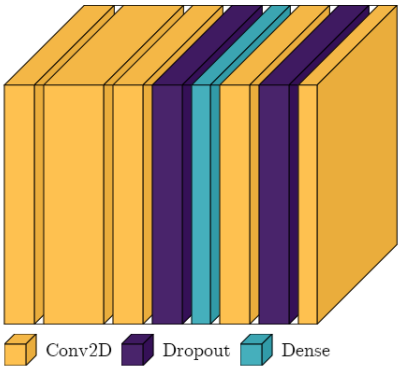

Figure 1: Pooling-free CNN architecture visualization.

Further, we reasoned that a pooling-free approach might learn an identity mapping between non-mask pixels in the input and output image, which would feasibly retain existing non-mask image features. In other words, we were optimistic that a pooling-free approach could successfully perform mask inpainting without adversely affecting existing facial features and background pixels.

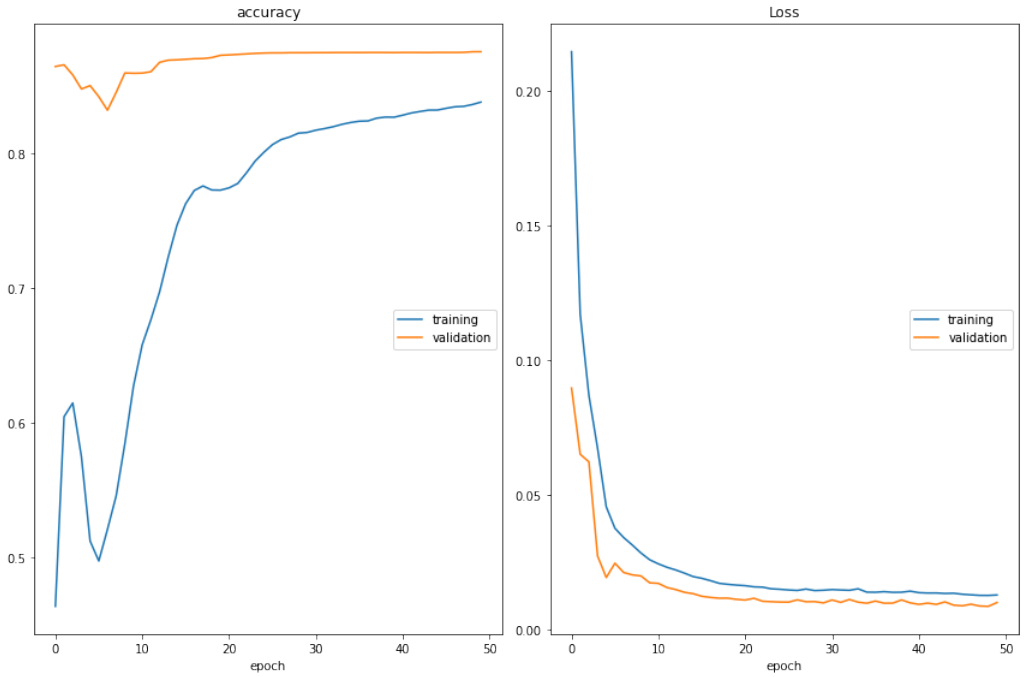

Figure 2: Pooling-free CNN accuracy and loss progression over 50 epochs.

Architecture: Our pooling-free network consists of five convolutional layers, two dropout layers, and one dense layer (see Figure 1). Input images were downscaled to dimensions of 256×256 pixels to reduce training time. Dropout layers were included after the third and fourth convolutional layers in an attempt to reduce overfitting, which we envisioned as a potential roadblock due to the visual and positional similarity of the masks in the training dataset. All convolutional layers employed the rectified linear activation unit (ReLU) function as the chosen activation function. The final convolutional layer serves to reduce the dimensionality of the fourth convolutional layer's output, which has a shape of $256 \times 256 \times 32$, back to the RGB layer space, $256 \times 256 \times 3$, which it achieves by only having three filters. In total, this model contains 72,486 parameters, all of which are trainable.

Training: The network was trained for 50 total epochs on a very small sampling of the "Face Mask Lite Dataset" [8]. The first 90 masked-unmasked image pairs in the dataset were selected as labelled training examples, and the 91^{st}-100^{th} image pairs were selected as validation examples. After the first 10 epochs, accuracy and loss graphs indicated consistent performance improvements, which scaled nonlinearly and converged upon reasonable values as the number of epochs increased (see Figure 2). The maximum training accuracy was found to be 0.839, and the minimum training loss was 0.012. In total, the entire network took approximately one hour to train.

Analysis: As shown in Figure 3, the pooling-free network was largely unsuccessful in generating facial structures, and instead appears to have uniformly discolored the masked input image while retaining most external details. In accordance with our predictions, this network does not appear to drastically compress the output image, likely as a result of the lack of pooling layers. However, the model clearly is not able to perform inpainting, as most of the mask's features, save for potentially the side-straps, were easily identifiable in predicted images. As such, while the model was able to attain reasonable accuracy and loss scores, it clearly was not providing solutions to the mask inpainting problem.

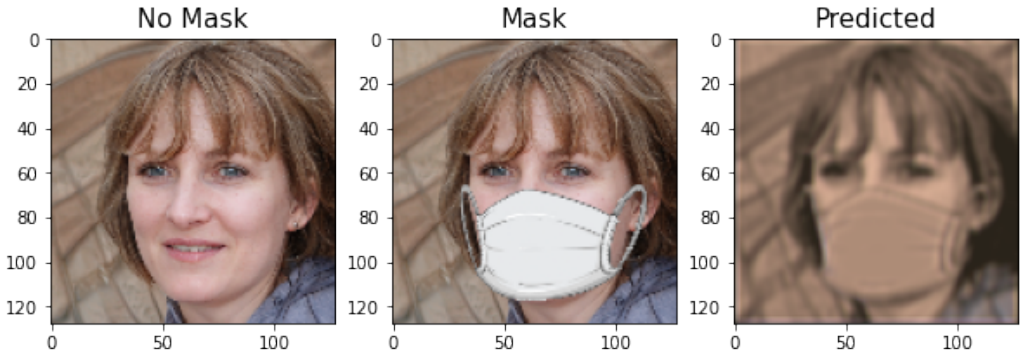

Figure 3: Pooling-free CNN predictions after final epoch.

2.2 Single-pool approach

Despite our initial concerns with output image downscaling as a result of pooling, we next decided to attempt an intermediate compromise between a pooling-free approach and a full-scale encoder-decoder approach. We reasoned that a single pooling layer might provide the necessary encoding for the network to begin to reliably recognize masks, while keeping the visual quality of the output image reasonably high. To restore the original input image dimensions, a single upsampling layer of equivalent size was added near the middle of the network.

Architecture: Our single-pool network consisted of nine convolutional layers, six dropout layers, two batch normalization layers, one dense layer, one pooling layer, and one upsampling layer (see Figure 4). Input images were downscaled even further than the pooling-free layer to dimensions of 128×128 pixels, as the increase in hidden layers substantially increased the training time of this network. All but three convolutional layers were followed immediately by dropout layers in an attempt to prevent overfitting. The introduction of batch normalization layers, inspired by [2], was intended to reduce internal covariate shift, a commonly-encountered problem for deep-CNN architectures. Again, a 3-filter convolutional final layer was utilized to restore proper dimensionality of the output image. In total, this model contains 279,910 parameters, 279,654 of which are trainable.

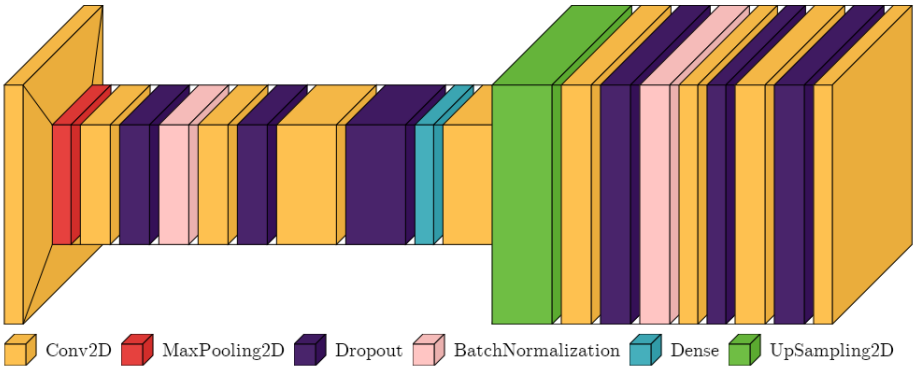

Figure 4: Single-pool CNN architecture visualization.

Training: The network was trained for 50 total epochs on a slightly larger sampling of [8]. The first 350 masked-unmasked image pairs in the dataset were selected as labelled training examples, and the 351st-400th image pairs were selected as validation examples. Throughout the entire training process, accuracy and loss graphs indicated consistent performance improvements, which again scaled nonlinearly and converged upon slightly improved values as compared to the pooling-free approach (see Figure 5). The maximum training accuracy was found to be 0.869, and the minimum training loss was 0.006. In total, the network took approximately three hours to train in its entirety.

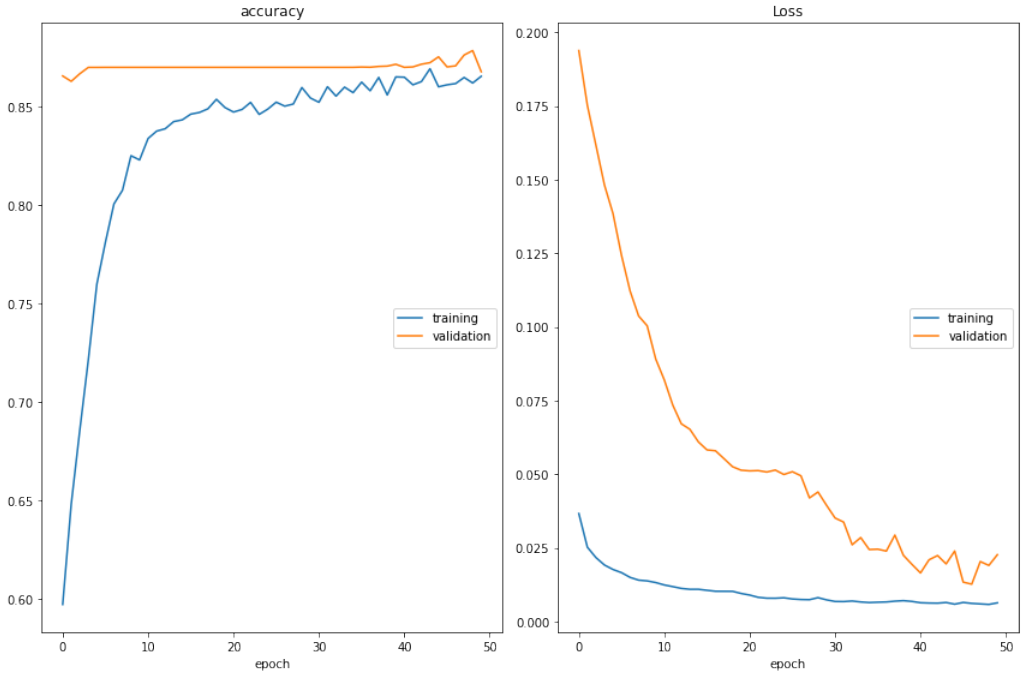

Figure 5: Single-pool CNN accuracy and loss progression over 50 epochs.

Analysis: Unfortunately, preliminary results for the single-pool CNN indicated what one might colloquially term "the worst of both worlds." The introduction of the pooling-upsampling paradigm appears to have substantially reduced the visual clarity of the output image, and successful inpainting of facial features was not observed (see Figure 6). Although the mask's original features were considerably less noticeable as compared to the pooling-free approach, the increased blurriness of the single-pool prediction was also rather noticeable. So, while the single-pool network may have been slightly more successful than the pooling-free network in removing masks, it was equally

unsuccessful at inpainting facial features and regrettably reduced the overall clarity of the output images.

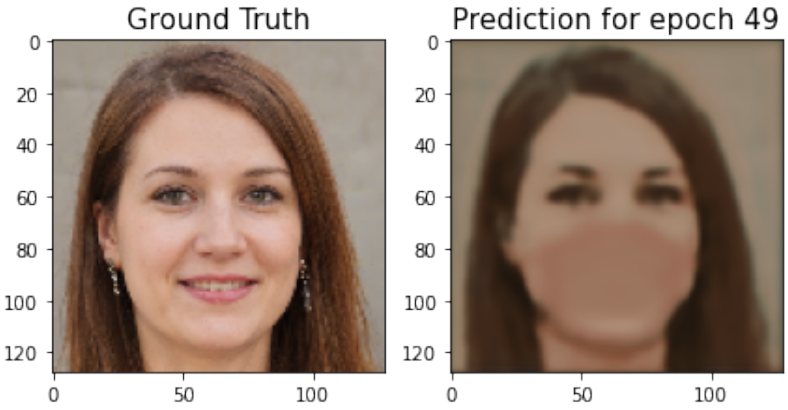

Figure 6: Single-pool CNN prediction after final epoch.

2.3 Encoder-decoder approach

Dismayed by the apparent failures of the pooling-free and single-pool approaches, but hopeful that the single-pool CNN's improved mask removal was indicative of improved mask recognition, we decided to attempt a full-scale encoder-decoder approach. Inspired primarily by [2], but also by [10] and [11], we constructed an encoder-decoder CNN with four pooling layers and four corresponding upsampling layers which produced surprisingly high-quality results.

Our initial designs for an encoder-decoder network, which were comparatively unsuccessful, included a network with only two pooling/upsampling pairs, a network with primarily asymmetric convolutional filter sizes and filter quantities, and a network with multiple miniature (single-pool) encoder-decoder subunits. Each of these encoder-decoder approaches failed to meaningfully generate facial features, and as such, we initially dismissed the encoder-decoder architecture entirely as a feasible data-efficient solution to the mask inpainting problem. We reasoned that successful results reported in the literature were principally reliant on large datasets and raw computational power, implying that an efficient approach would be comparatively impractical.

However, it was not until we attempted to construct a model with near-symmetrically equivalent encoder and decoder architectures that we began to see drastic improvements. By constructing a mirrored encoder-decoder architecture, we witnessed rudimentary facial reconstructions as early as epoch 20, and decided to upscale the initial images from 128×128 pixels to 256×256 pixels, despite the increase in training time.

Architecture: Our encoder-decoder network consisted of eleven convolutional layers, four dropout layers, four pooling layers, four upsampling layers, and four leaky ReLU layers (see Figure 7). The reduction in the number of dropout layers and the removal of the batch normalization layers (as compared to the single-pool architecture) represented a respective shift of focus away from concerns about overfitting and internal covariate shift in favor of successful facial structure generation. The introduction of leaky ReLU layers, inspired by [2], served to perform activation of key convolutional layers immediately after dropout layers, not before. Once more, a 3-filter convolutional final layer was utilized to restore proper dimensionality of the output image. In total, this model contains 555,459 parameters, all of which are trainable.

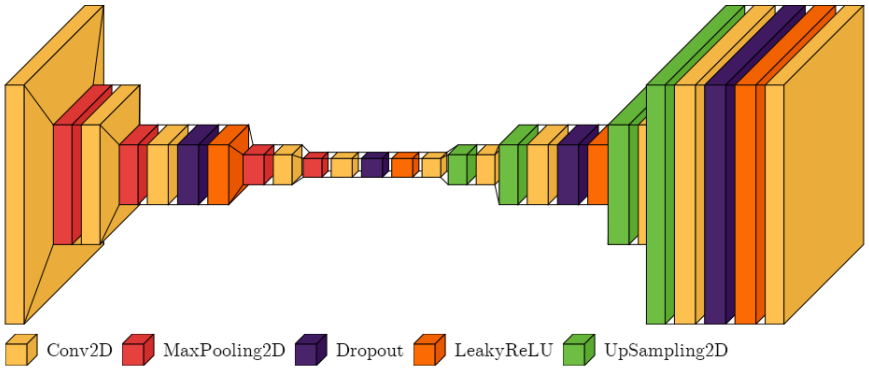

Figure 7: Encoder-decoder CNN architecture visualization.

Training: This CNN was trained for 100 total epochs on an even larger sampling of [8] than the single-pool network. The first 450 masked-unmasked image pairs in the dataset were selected as labelled training examples, and the 451st-500th image pairs were selected as validation examples. Save for the very first few epochs, accuracy and loss graphs indicated consistent performance improvements, which again scaled nonlinearly and converged upon slightly improved values as compared to both the pooling-free and single-pool approaches (see Figure 8). The maximum training accuracy was found to be 0.873, and the minimum training loss was 0.005. In total, the network took approximately five hours to train in its entirety.

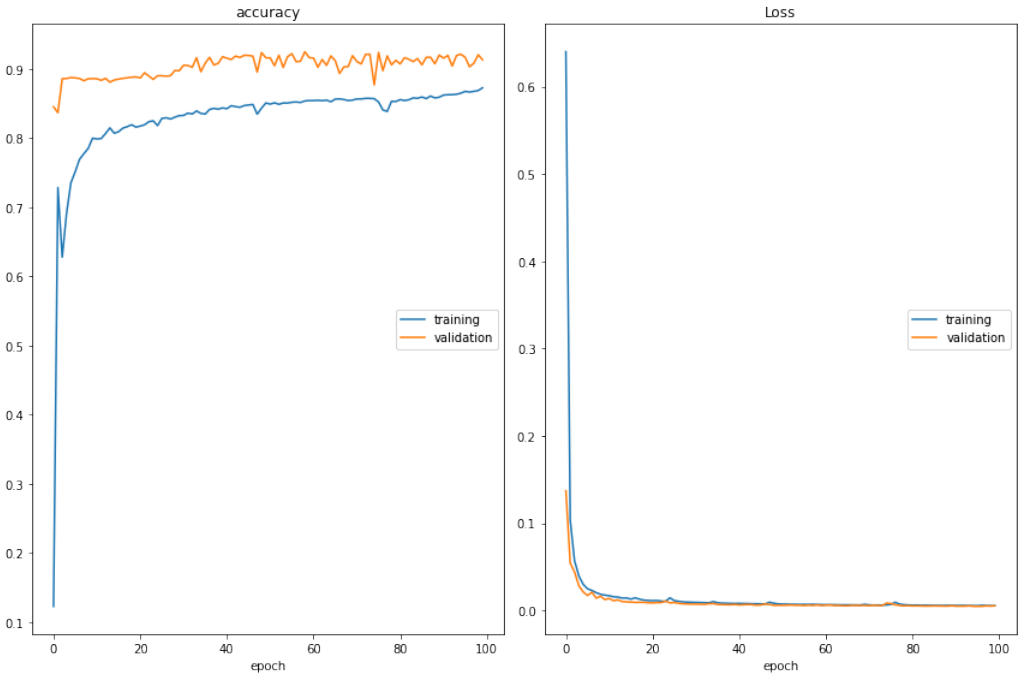

Figure 8: Encoder-decoder CNN accuracy and loss progression over 100 epochs.

Analysis: As evidenced by Figure 9, our encoder-decoder produced output images that were incomparably better than the predictions of the pooling-free and single-pool networks. The encoder-decoder predictions successfully performed facial structure generation, and effectively no traces of the original mask remained present in the output image. Critical facial features, such as the mouth and nose, were unfailingly generated across all testing samples we observed. Furthermore, original colors and background features remained reasonably well-preserved, thus confirming that the model was not substantially modifying non-mask pixels.

Quantitative analysis of our encoder-decoder's performance on larger testing datasets also agreed with our qualitative analysis, confirming that the model performs well on images from [8] that were not included in the training dataset. An average accuracy of 0.893 and an average loss of 0.006 were obtained after testing the model on 2000 non-training example images from [8]. Comparing this accuracy to the maximum validation accuracy of the previous two CNN architectures, which were slightly lower, it becomes evident that our encoder-decoder is superior from both qualitative and quantitative perspectives.

However, our encoder-decoder model's predictions were certainly not without flaws. Chiefly, the predicted images were the blurriest of those produced by our CNN models, which may likely be attributed to the presence of the four pooling layers in the encoder architecture. We speculate that the increase in the number of pooling layers is precisely what allowed for successful facial feature generation, but was also responsible for the undeniable loss in visual clarity.

Another potential flaw can be spotted in the eyes of each prediction in Figure 9. While the sclerae of subjects in both the masked and maskless images can be clearly distinguished from the retina and pupil, the predicted images contain subjects with comparatively homogeneous, dark eye colorations. This loss of previously available facial features is perhaps indicative of the consequences of the encoder-decoder approach, which necessarily loses visual information after each pooling layer is encountered.

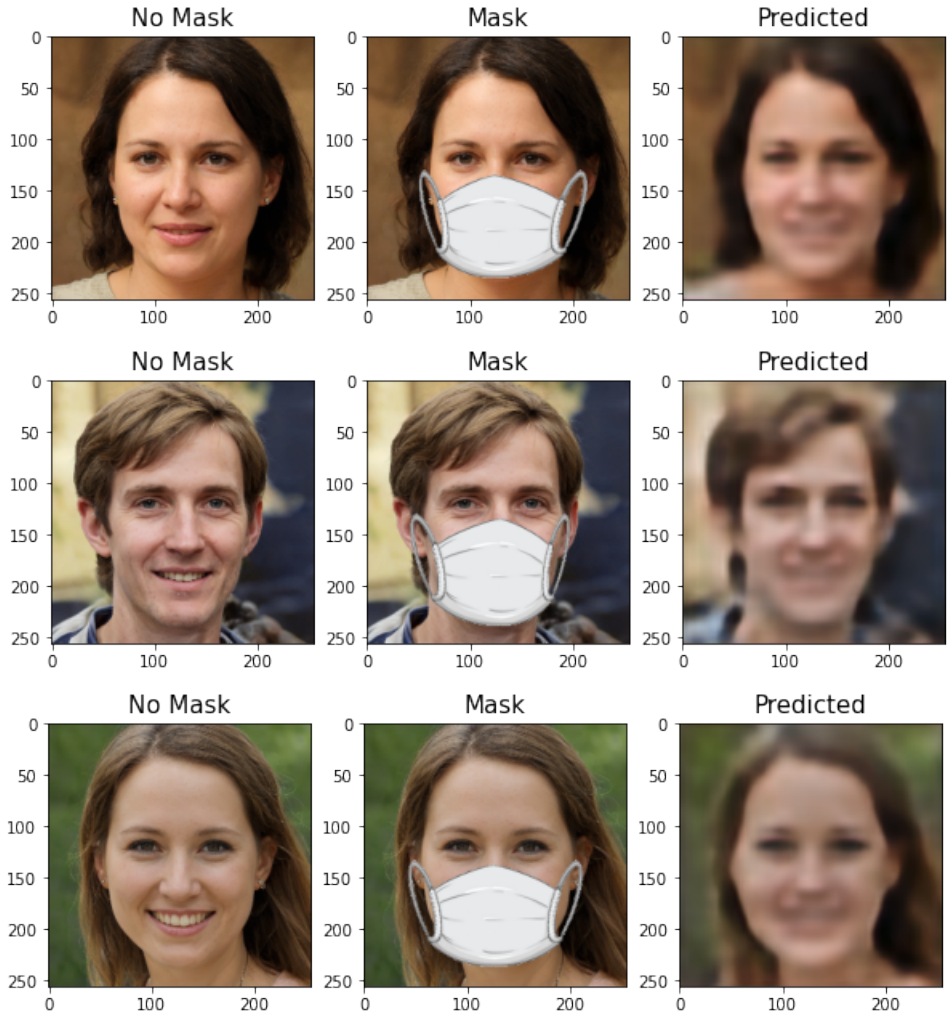

Figure 9: Encoder-decoder CNN predictions after final epoch.

3 Generative Adversarial Network (GAN) Approaches

With the recent success of GAN-based approaches in applications to facial inpainting [1, 6, 9, 12], we also decided to explore the applicability of convolutional GANs to the mask inpainting problem. Given the success of our encoder-decoder CNN, we decided to preserve the exact architecture of this model and utilize it as the generator for a GAN-based approach. A shallow convolutional discriminator architecture was constructed as well, which attempts to classify generated images as either "real" or "fake." After training the encoder-decoder generator, we decided to construct a novel generator architecture which utilizes transposed convolutional layers in place of consecutive upsampling and standard convolutional layers.

In the following subsections, we chronologically present these two GAN architectures. The generators and discriminators of each model were trained using Adam stochastic gradient descent optimizers to prioritize computational efficiency (per [4]), as previously mentioned. Both GAN models also utilized binary cross-entropy loss functions for the generator and discriminator, which effectively made the loss of the generator "how well did I trick the discriminator" and the loss of the discriminator "how often was I tricked."

3.1 Encoder-decoder generator with pooling-free discriminator

For our first attempt at a GAN-based solution to the mask inpainting problem, we decided to train the same encoder-decoder CNN architecture in a GAN-based learning environment. To achieve this goal, a simple convolutional discriminator architecture was constructed to classify the images generated by the encoder-decoder architecture. The key difference between this GAN-based approach and the previous pure-CNN approach can be found in the loss functions: in this approach, the encoder-decoder is penalized for failing to "fool" the discriminator, as opposed to the straightforward MSE loss function utilized in the pure-CNN training approach.

Architecture: As mentioned previously, the architecture of the generator was exactly identical to our aforementioned encoder-decoder CNN (see Figure 7). However, we also constructed a simple convolutional discriminator architecture (see Figure 10) consisting of two convolutional layers, two leaky ReLU layers, one dropout layer, one flattening layer, and one dense layer. The final dense layer contained a single node connected to each of the flattened nodes in the previous layer, thus mapping the input image to a binary classification output. In total, this model contains a sizeable 8,464,257 parameters, all of which are trainable. The large number of parameters is principally due to the dimensionality upscaling introduced by the two convolutional layers, which contain 64 and 128 filters respectively, as these upscaled outputs are subsequently flattened and densely connected to a single output node in the final two layers.

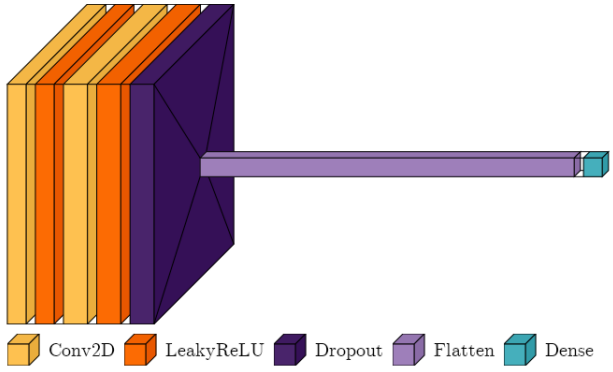

Figure 10: Pooling-free discriminator architecture visualization.

Training: This GAN was trained for 150 total epochs on a 320-image sampling of [8], with no validation images. Given the highly qualitative nature of analyzing the results of GANs, we decided to omit accuracy and loss graphs, which we reasoned would not provide sufficiently meaningful information for analysis. In total, the network took approximately six hours to train in its entirety.

Analysis: As shown in Figure 11, the results for our encoder-decoder GAN were surprisingly poor, especially considering the quality of our results with the exact same architecture trained with a standard RMS loss function. While the GAN was clearly able to generate major facial features, such as noses and mouths, it homogenized its output to a single overall facial structure, effectively discarding background features that should have ideally been left unchanged. The two predictions in Figure 11 clearly demonstrate that our GAN produces visually similar output images on two highly distinct input images, indicative of this indiscriminate homogenization of input images towards one particular facial structure.

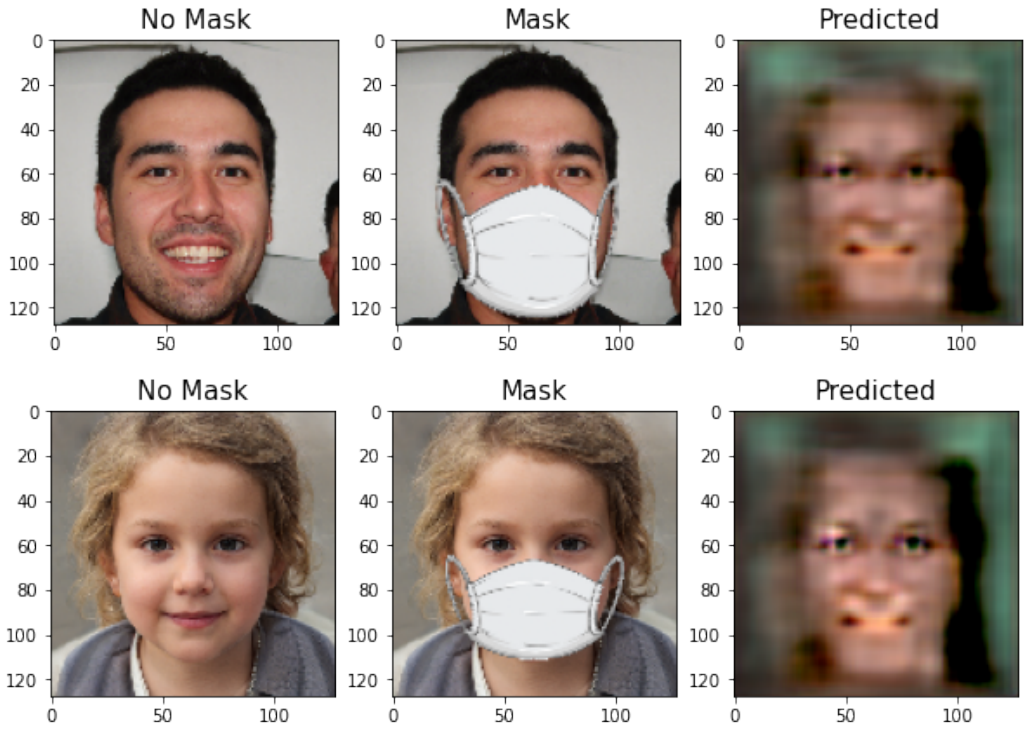

Figure 11: Encoder-decoder generator predictions after final epoch.

3.2 Transposed convolutional generator with pooling-free discriminator

Given the disappointing performance of our initial encoder-decoder GAN, we hoped to improve upon our results by employing a novel generator architecture that was distinct from any of our previous CNN or GAN attempts. After some consideration, we decided to replace the convolutional and upsampling layer pairs with transposed convolutional layers, which we hoped would increase decrease the blurriness of our output images and would provide a greater diversity in generated facial structures.

Architecture: For this GAN, we decided to utilize the same pooling-free discriminator from our previous GAN model. However, we constructed a novel generator architecture (see Figure 12) consisting of four standard convolutional layers, three leaky ReLU layers, two dropout layers, two pooling layers, and two transposed convolutional layers. The introduction of the transposed convolutional layers served as a modernized substitute for the previous convolution-upsampling paradigm utilized in the encoder-decoder and single-pool CNNs. In total, this model contains a modest 391,811 parameters, all of which are trainable.

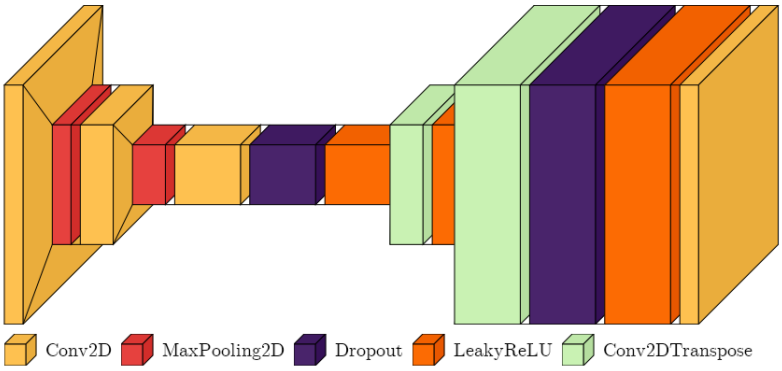

Figure 12: Transposed convolutional generator architecture visualization.

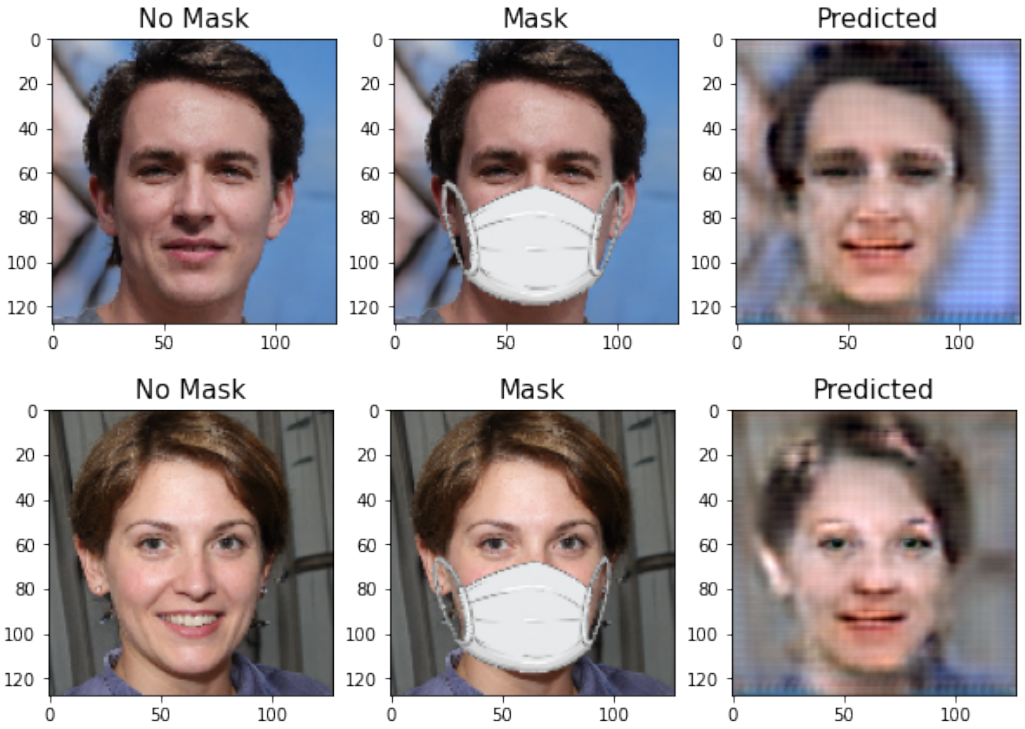

Figure 13: Transposed convolutional generator predictions after final epoch.

This GAN was trained for 150 total epochs on the same 320-image sampling as the encoder-decoder GAN, with no validation images. Again, given the highly qualitative nature of analyzing the results of GANs, we decided to omit accuracy and loss graphs, which we reasoned would not provide sufficiently meaningful information for analysis. In total, the network took approximately twelve hours to train in its entirety.

Analysis: As shown in Figure 13, the results for our transposed convolutional generator were certainly better than our encoder-decoder generator. Non-mask and background pixels were reasonably preserved, and important facial features were clearly generated on all testing predictions. However, much like the encoder-decoder generator, this model homogenized its inpainting to a single overall facial structure, which was still decidedly better than our previous GAN. The two predictions in Figure 13 clearly demonstrate that our GAN produces visually similar smiles on two highly distinct input images, indicative of this facial structure homogenization.

One particularly interesting facet of this model's predictions, however, was its improved eye generation as compared to the encoder-decoder CNN. Instead of removing the sclerae of input images, our transposed convolutional generator was better able to maintain eye features, as evidenced by the second prediction in Figure 13.

4 Results

We have explored various model architectures in order to solve the mask inpainting problem, and found that the CNN model employing the encoder-decoder approach outperformed pooling-free, single-pooling, and GAN-based models. With only 555,459 trainable parameters, 450 training samples, and around 5 hours of training time, this model could successfully remove a mask and add in impressive facial features. Compared to the 6,047,719 trainable parameters and 2300 training images utilized in [2], our encoder-decoder model performs somewhat comparably despite the inherent disadvantages of a reduced parameter space and a smaller training dataset.

However, the encoding step in the model likely results in the "blurry" nature of the output, as the model must first compress the large dimension space before reconstructing the image. Given that the pooling-free and single-pooling approaches did not learn to superimpose facial structures, it is likely that this reduction in dimensionality is precisely what allowed the model to discern facial features. These pooling layers, therefore, represent a double edged sword in the training of our encoder-decoder model, as they help to learn and append facial features while also reducing image clarity. It is possible that increasing training data or the number of epochs could help reduce this fuzziness, but the inherent dimensionality reduction of pooling is more likely the blurry culprit – even with more training, the model would still need to greatly compress the image. Given this limitation, we propose that upscaling our image prior to encoding could remove this compression issue, while still allowing the model to use the encoder-decoder approach that has so effectively and reliably inpainted facial features.

The model was also overfit to the point of producing a single smile. Of course, without much facial context, the model cannot be expected to exactly reproduce the original image, although it *could* certainly vary the facial features it produces. We believe that this overfitting likely comes from using a relatively small number of training samples. Increasing our training pool could potentially fix this issue, although the model would still not have necessary context to accurately recreate the masked person's face.

5 Conclusions

Despite these flaws in the encoder-decoder model, it achieved an impressive 89.3% average accuracy over 2000 testing images. Further, its ability to understand facial features with little data and training time represents an important step towards model efficiency. While the other CNN and GAN approaches discussed in this paper did not perform to the same standard, they still solved interesting aspects of the mask inpainting problem. Both the pooling-free and single-pooling models maintained the unmasked portion of the images and could identify mask location. The GAN approaches, on the other hand, identified mask location and began to understand facial features. They did, however, produce inaccurate, somewhat uncanny-valley-esque results.

In the future, we hope to create models that can intuit faces given previous context. These models would accept unmasked and masked images of a person. They would then gain an understanding of that person's facial features and fill in the masked area with a realistic imitation of that person's face. We also hope to extend these models to a robust database that includes more varied images capturing multiple individuals.

References

[1] Avisek Lahiri, Arnav Jain, Divyasri Nadendla, & Prabir K. Biswas; "Improved Techniques for GAN based Facial Inpainting," *arXiv preprint*, 2018. https://arxiv.org/pdf/1810.08774.pdf

[2] "Black Mamba"; "Autoencoder: Who is behind mask?" 2021. https://www.kaggle.com/code/theblackmamba31/autoencoder-who-is-behind-mask.

[3] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, & Alexei A. Efros; "Context Encoders: Feature Learning by Inpainting," *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. https://arxiv.org/abs/1604.07379

[4] Diederik P. Kingma & Jimmy Ba; "Adam: A Method for Stochastic Optimization," *International Conference on Learning (ICLR)*, 2015. https://arxiv.org/pdf/1412.6980.pdf

[5] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, & Thomas S. Huang; "Generative Image Inpainting With Contextual Attention," *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 5505-5514. https://openaccess.thecvf.com/content_cvpr_2018/html/Yu_Generative_Image_Inpainting_CVPR_2018_paper.html

[6] Jiancheng Cai, Hu Han, Shiguang Shan, & Xilin Chen; "FCSR-GAN: Joint Face Completion and Super-resolution via Multi-task Learning,"*IEEE Transactions on Biometrics, Behavior, and Identity Science*, 2019. https://arxiv.org/pdf/1911.01045v1.pdf

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun; "Deep Residual Learning for Image Recognition," *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. https://arxiv.org/abs/1512.03385

[8] Prasoon Kottarathil; "Face Mask Lite Dataset," 2020. https://www.kaggle.com/datasets/prasoonkottarathil/face-mask-lite-dataset.

[9] Xian Zhang, Canghong Shi, Xin Wang, Xi Wu, Xiaojie Li, Jiancheng Lv, & Imran Mumtaz; "Face inpainting based on GAN by facial prediction and fusion as guidance information," *Applied Soft Computing*, Volume 111, 2021. https://www.sciencedirect.com/science/article/pii/S1568494621005470

[10] Xiaoming Li, Ming Liu, Jieru Zhu, Wangmeng Zuo, Meng Wang, Guosheng Hu, & Lei Zhang; "Learning Symmetry Consistent Deep CNNs for Face Completion," *arXiv preprint*, 2018. https://arxiv.org/pdf/1812.07741v1.pdf

[11] Yi Wang, Xin Tao, Xiaojuan Qi, Xiaoyong Shen, & Jiaya Jia; "Image Inpainting via Generative Multi-column Convolutional Neural Networks," *Conference on Neural Information Processing Systems (NeurIPS)*, 2018. https://proceedings.neurips.cc/paper/2018/file/6f3ef77ac0e3619e98159e9b6febf557-Paper.pdf

[12] Youngjoo Jo & Jongyoul Park; "SC-FEGAN: Face Editing Generative Adversarial Network with User's Sketch and Color," *Proc. International Conference on Computer Vision (ICCV)*, 2019. https://arxiv.org/pdf/1902.06838v1.pdf