# Face Mask Detection and Defend Against One Pixel Attack

**Jeong Ho Park**
mandudu@live.unc.edu

**Kangda Wei**
kangda@cs.unc.edu

**Renee Du**
karends@email.unc.edu

**Mingli Zhang**
mingliz@cs.unc.edu

**Sara Qi**
xiaoyuqi@email.unc.edu

**Xinjie Qian**
qianqxj@live.unc.edu

## 1 Introduction

The COVID-19 pandemic changed the world in 2020, and its impact continues until the present. People are required or recommended to wear medical masks to mitigate the propagation of such airborne diseases. There is thus an increasing demand for medical mask detection in public places, such as hospitals, railway stations, and shopping malls, to remind people to wear masks. Medical mask detection using machine learning saves laborers and minimizes unnecessary contact between pedestrians and checkers. Our research project thus uses CNN and Residual Neural Network (ResNet) to implement a model to detect whether the person in a picture is appropriately wearing a medical mask. To improve the robustness of our detection model, we will also be attacking using a one-pixel attack Su et al. [2019], a flavor of adversarial attack, using a differential evolution algorithm and limiting the attacking pixel to be on the person's face or mask. We intend to mimic the real-life scenario where a person has a mark on their face or face mask, either intentionally or unintentionally, as it is more likely than someone creating a tailored attacking filter to alter the detection results. Some examples of one-pixel-attack are shown in Figure **??**. Last but not least, we will implement some defense mechanisms, such as preprocessing the images or modifying the model, to protect the models from one-pixel attacks. However, this is an attack and defense game, so the attacking algorithm can improve the attack more efficiently after the defense. Our goal is at least to defend against the "first wave" of the attack and potentially improve by iterating the attack and defense cycle.

## 2 Related Work

Nowadays, CNN and ResNet are widely used in lots of fields. Lecun et al. [1998] demonstrated that a CNN model, which aggregates simpler features into progressively more complicated features can be used for handwritten character recognition. Krizhevsky et al. [2012] described the winning AlexNet model, which achieved state-of-the-art performance labeling pictures in the ImageNet challenge. He et al. [2015] introduced a deep residual learning framework using unhindered skip connections to avoid the vanishing gradient problem. These two classical methods were used in this project to perform classification.

There are many types of adversarial attacks, such as gradient-based attacks, score-based attacks, and hard-label attacks. Goodfellow et al. [2014] proposed a "fast gradient sign" algorithm for calculating effective perturbation based on a hypothesis in which the linearity and high dimensions of inputs are the main reason that a broad class of networks is sensitive to small perturbation. Papernot et al. [2016] utilized the Jacobian matrix to build an "Adversarial Saliency Map," which indicates the effectiveness of conducting a fixed length perturbation through the direction of each axis. However, their perturbation is conducted on about 4% of the total pixels and can be evident to the human eyes.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).


Figure 1: Examples of Successful Attacks

In this project, we use the one-pixel attack proposed by Su et al. [2019] as our attacking method. Vargas and Su [2019] used propagation maps to analyze the one-pixel attack and showed how a pixel modification causes an influence throughout the layers, culminating in the change of the class. Peng Wang and Li [2021] proposed detection methods to analyze the vulnerability of DNN models and gave the most suspected pixel modified by the one-pixel attack.

Husnoo and Anwar [2021] introduced an image recovery and reconstruction approach for defending state-of-the-art DNNs against one-pixel attacks. The algorithm can achieve the effect of passive defense against one-pixel attacks without the need to train another model by recovering the malicious pixel in the adversarial sample and, thus, reconstructing the original image. Chen et al. [2019] proposed a Patch Selection Denoiser (PSD) approach that removes the few potential attacking pixels in local patches without changing many pixels in a whole image. The model can defend against 98.6% one-pixel attacks without bringing side effects on clean images not subject to one-pixel attacks in the experiment. However, it is essential to note that this approach has received less acclaim due to the degradation of the image through the denoising neural network, even though the authors claimed to have achieved a successful defense rate of 98.6 percent against one-pixel attacks.

Additionally, Liu et al. [2020] proposed a 3-step image reconstruction method to remove the target image's potential attacking pixels. However, the paper claims that the parameters could have been more optimal, and therefore, this approach might lead to information loss. Lastly, Shah et al. [2020] proposed using an Adversarial Detection Network (Adnet) to detect one-pixel attacks. The drawbacks of this method include the complete discrimination and rejection of the adversarial input, the training of a new network that is computationally expensive, and the accuracy of the solution proposed is lower for the detection of the one-pixel attack.

Following the review of the previous work against one-pixel attacks, there still is a need for a computationally efficient and robust defense strategy. In our paper, we did not design a new architecture of Neural Network but instead utilized the Seam Carving method to defend against the attack. The seam Carving method is more computationally efficient and more suitable for us to implement, given the time and limitation to computation power.

## 3 Dataset

For this project, we are using a face mask detection data set from Kaggle. The data set can be found with the following link: `https://www.kaggle.com/datasets/andrewmvd/`

`face-mask-detection`. This data set contains around 7500 examples, with 3750 examples containing "with mask" labels and 3750 containing "without mask" labels. Examples are shown in Figure 7. We only reported the classification accuracy for the first data set for the preliminary results. The reported accuracy is the validation accuracy, and the validation set contains 10% of the first dataset that is randomly selected.

Figure 2: Examples of the dataset used for preliminary results.

## 4 Models

There are many existing image recognition models, and we want to expand our project beyond the scope of one model. Therefore we chose two models to conduct our test, a trained-from-scratch CNN model and the award-winning Deep neural network ResNet (He et al. [2015]). The choice of CNN is simple; apart from being the foundation of most image recognition models, it is simple and easy to train and will often get accuracy on par with other models. Therefore, it is also interesting to us how robust a simple CNN model can be when facing a one-pixel attack. On the other hand, we want to use a pre-trained model built on top of the CNN model to test out how good of a foundation CNN is and the potential of building off that foundation. Therefore, ResNet was chosen because, according to the ResNet paper, it is easier to optimize, which can ease our computational complexity. At the same time, it can still get great accuracy by increasing the model's depth (He et al. [2015]).

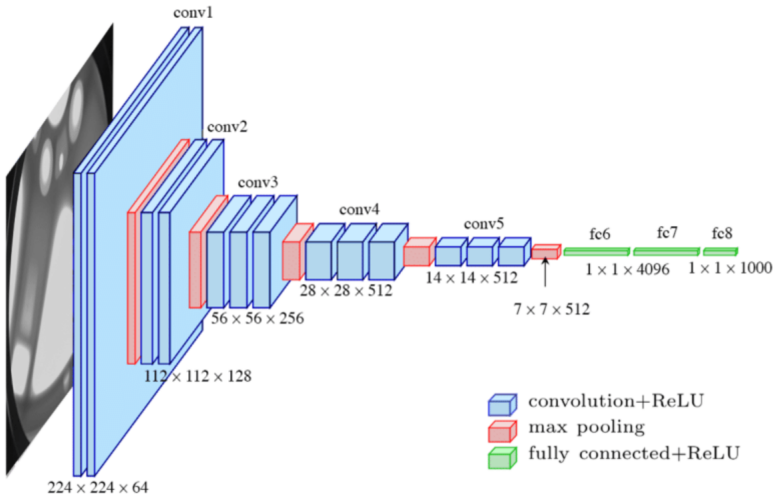


Figure 3: CNN Model Architecture

Above is a figure demonstrating the neural network logic of a simple CNN network. As we can see from the figure, CNN is a highly customizable network, as any step of the process can be altered to adjust the model and provide different model behaviors.

# 5 Method

## 5.1 One-Pixel Attack

To perform the attack, we use an Evolutionary Algorithm called Differential Evolution (DE) that iteratively generates adversarial images, which are acquired by selecting a pixel and changing its color to try to minimize the confidence of the neural network's classification. We first generate several adversarial images for a specific image by picking and modifying a random pixel. We then run the images through the neural network to get the probability distribution of the classification. Next, we combine the positions and colors of the previous images' random pixels to generate new adversarial images and run them through the neural network. If the newly generated pixels reduce the network's confidence from the previous step, we treat them as the current optimal solutions. We acquire the final answer by repeating the former steps and returning the adversarial image on the last iteration, which decreases the network's confidence the most. A successful attack will have an adversarial image whose confidence in the correct category would be reduced so much that a new category now has the highest classification confidence. The illustration of Differential Evolution is shown in Figure 4

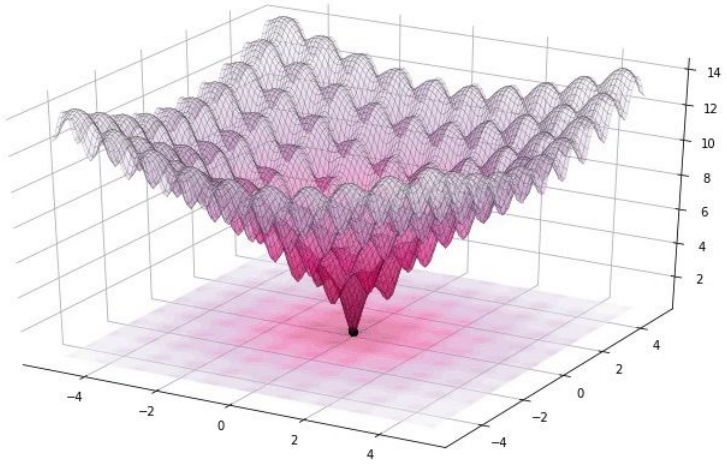


Figure 4: Visualization of Differential Evolution

Now we describe the attacking process step by step. First, formulate it as an optimization problem: in an untargeted attack, minimize the confidence of the correct class, and in a targeted attack, maximize the confidence of a target class.

When performing black-box optimizations such as the one-pixel attack, it can be extremely difficult to find an efficient gradient-based optimization that will work for the problem. It would be nice to use an optimization algorithm that can find optimal solutions without relying on the smoothness of the function. In our case, we have discrete integer positions ranging from 0 to 31 and color intensities from 0 to 255, so the function is expected to be jagged.

Differential evolution is a type of evolutionary algorithm where a population of candidate solutions generates offspring that compete with the rest of the population according to their fitness. Each candidate solution is shown by a vector of real numbers which are the inputs to the function we would like to minimize. The lower the output of this function, the better the fitness. The algorithm works by initializing a (usually random) population of vectors, generating new offspring vectors by combining (mutating) individuals in the population, and replacing worse-performing individuals with better candidates.

In the context of the one-pixel attack, our input will be a flat vector of pixel values:

$$X = (x_1, y_1, r_1, g_1, b_1, x_2, y_2, r_2, g_2, b_2, ...)$$

These will be encoded as floating-point values but will be floored back into integers to calculate image perturbations. First, we randomly initialize adversarial images where the x-y coordinates

of each adversarial image are from the uniform distribution of the image dimensions, in this case, $x_i, y_i \sim U(1, 32)$, and the RGB values are from the normal distribution with mean and standard deviation being 128 and 127 respectively such that $r_i, g_i, b_i \sim N(\mu = 128, \sigma = 127)$. Next, we generate a random population of $n$ perturbations

$$\mathbf{P} = (X_1, X_2, \ldots, X_n)$$

Then, on each iteration, we calculate $n$ new mutant children using the formula

$$X_i = X_{r1} + F(X_{r2} - X_{r3})$$

such that

$$r1 \neq r2 \neq r3$$

where $r1, r2, r3$ are random indices into our population $\mathbf{P}$, and $F = 0.5$ is a mutation parameter. We pick 3 random individuals from the previous generation and recombine them to make a new candidate solution. If this candidate $X_i$ gives a lower minimum at position $i$ (i.e., the attack is closer to success), replace the old $X_i$ with this new one. This process repeats for several iterations until we find an image that completes the attack.

### 5.2 Defense Against One-Pixel Attack

There are two mainstream ways of defending against one-pixel attacks. The first and most obvious way is to modify the neural network to be more robust. For instance, Papernot et al. [2016] proposed a defensive mechanism called defensive distillation in which a neural network is added to learn and predict the class probabilities generated by the original algorithm's output in 2016. However, this approach often means retraining the entire neural network, which can use up many resources and computing power while failing to deliver significant results Papernot et al. [2016]. The second method we will be focusing on is introducing some image preprocessing into the network pipeline that will denoise the image or ensure the image does not contain much useless information during the resizing process. The most promising method is Patch Selection Denoiser (PSD), proposed by Chen et al. [2019]. Patch Selection Denoiser uses neural network denoiser and local-patch method to remove the potential attacking pixels. However, the method is computationally consuming and still requires retraining a new model Liu et al. [2020]. Thus, we intend to try out a relatively newer and more computationally efficient defense method using seam carving. The following sections introduce the seam carving process we implemented and why we chose this method.

#### 5.2.1 What is Seam Carving

Figure 5: Visualization of Seam Carving

Due to the data set images varying in size and the CNN/ResNet model needing the input images to be of the same size, we need to resize the images before passing those images into the models. The standard way to perform image resizing is through super-sampling or sub-sampling. These ways are fast and lightweight, but they are also very unintelligent; they do not care about what the image content is and thus will likely keep useless data or even enlarge them in the resizing process. Therefore, to combat this behavior, we will opt for a relatively new technique called seam carving

Avidan and Shamir [2007]. This technique uses a weight map of each pixel, usually the entropy map, of the image and dynamic programming to find the vertical or horizontal seam with the least total weight and remove that seam. It can also introduce new seams without disturbing the balance of the overall entropy of the image. The process of seam carving is shown in Figure 5. We also show two seam-carved images from the face mask dataset in Figure **??**.

Figure 6: Visualization of Seam Carved Face Mask Images

### 5.2.2 Why Seam Carving

Of course, there are other image-resizing methods, but why did we choose seam carving? The most fundamental reason is the content-aware properties that seam carving possesses, but it is only part of the picture. First, seam carving will produce a better-balanced image, as pixels with more weight are more likely to remain in the final result. Well-balanced image data will make the one-pixel attack harder as the attack algorithm decides which pixel to attack based on the pixel weights. Another reason is the customizability of seam carving. Due to the carving being based on the weight map of the image, we have the liberty to customize the weight map calculation. For example, we tried to add weights to the person's face area using a face detection model; this way, the seam carving algorithm is less likely to modify the person's face and will focus on unimportant pixels, such as the backgrounds, first. However, during our testing, we found that seam carving is generally intelligent enough to carve the face as little as possible. The intensive computational cost outweighed the advantages of weight manipulation using a face detection model.

## 6 Experiments and Results

We train two different models for the face mask detection part. The first model is a trained-from-scratch CNN model, and the second is a ResNET34 model pre-trained on the ImageNet dataset. We use Stochastic Gradient Descent as the optimizer with a max learning rate of 0.001. The batch size is set to 256. We train the CNN model for 40 epochs and fine-tune the ResNET34 model for 40 epochs. ResNET34 has a validation accuracy of 89.8%, and CNN has a validation accuracy of 84.6%.

### 6.1 Attack and Results

For the attack, we perform the differential evolution algorithm mentioned in Section 5.1. We sampled 100 images from the test set that the models successfully predicted for the attack. We ran the differential evolution algorithm for 100 iterations for each image that is attacked before giving up and failing the attack. We generate 100 adversarial images during each iteration using the differential evolution algorithm. We report the results in Table 1. Two successful attack examples are shown below in Figure 1

From Table 1, we can see that the success rate for the one-pixel attack is 0%, this is because we only have access to Macbooks and the computational power of this hardware is not enough to run many tests. If we increase the number of tries to even higher, the whole program will crash due to memory limitations. Therefore, we increased the number of pixels attacked to 3 and 5 and we were able to get none zero attack successful rates. The more pixels we attack at the same time, the higher the attack success rate will be. From the table, we can tell that our trained-from-scratch CNN model has lower accuracy and robustness than using the pre-trained ResNet model. For the CNN model, the pure accuracy rate is around 85% while the one-pixel attack successful rate using three and five

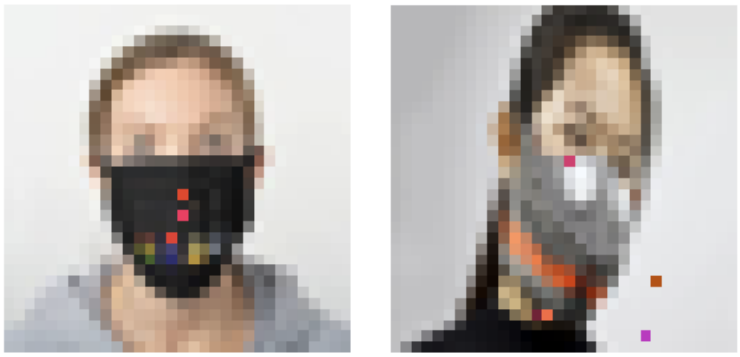


Figure 7: Examples of n-pixel-attack success.

pixels are 6% and 11% respectively. On the other hand, the ResNet model has a pure accuracy of almost 90% while the one-pixel attack successful rate using three and five pixels is only 1% and 3%.

| model | accuracy | pixels | attack success rate |
|---|---|---|---|
| CNN | 84.6% | 1 | 0% |
| | 84.6% | 3 | 6% |
| | 84.6% | 5 | 11% |
| ResNET | 89.8% | 1 | 0% |
| | 89.8% | 3 | 1% |
| | 89.8% | 5 | 3% |

Table 1: Attack results

### 6.2 Defense and Results

For the defense, we used seam carving to resize the original image into 32 x 32 images and unleashed the one-pixel attack on these carved images. We also tested how much the defense mechanism impacts our models' accuracy. After all, it will not be a suitable defense procedure if the defense itself severely jeopardizes the model's accuracy.

| model | accuracy | pixels | attack success rate |
|---|---|---|---|
| CNN | 73.6% | 1 | 0% |
| | 73.6% | 3 | 2% |
| | 73.6% | 5 | 4% |
| ResNET | 87.6% | 1 | 0% |
| | 87.6% | 3 | 0% |
| | 87.6% | 5 | 1% |

Table 2: Defense results

From the above table, the defense mechanism worked and can defend against the one-pixel attack to some degree while not hindering the model accuracy too much. Since we already know that our CNN model is less robust than the ResNet model, the model's accuracy dropped by around 10% just using seam-carved test data. However, the ResNet model only dropped by about 2%, showing its sturdiness again. In general, the defense mechanism cut the one-pixel attack success rate by more than half or even lowered it to 0%. This shows that our expectation of using a more balanced out and the equal-weighted image will increase the difficulty level for a black box one-pixel attack algorithm, reducing the attack success rate.

## 7 Conclusion

Our project group has successfully implemented a from-scratch CNN model and a pre-trained ResNet model for mask detection with an accuracy rate of $84.6\%$ for the CNN model and $89.8\%$ for the ResNet model. Since there are many variations to the data set in the real world by having different colored masks, transparent masks, and dotted masks, we came up with the idea of removing N-pixels within the masks to enhance the robustness of our model.

Therefore, we implemented an N-pixel-attack strategy in each mask on the 100 image samples in the dataset by utilizing the differential evolution method. As the data is shown in Table 1: Attack results, we found that the ResNet model had higher accuracy and robustness than the from-scratch CNN model with the implementation of an N-pixel attack.

For the defense mechanism, our primary strategy was to use a data pre-processing technique called the Seam Carving method. We found out that the accuracy rate decreased by $13\%$ for the from-scratch CNN model and $2\%$ for the ResNet model when the Seam Carving method was implemented; however, the data pre-processing defense mechanism we used drastically lowered our attack-success-rate by about $60\%$.

Although it laid a controversy by depicting a trade-off of giving up the accuracy rate by a small percentage and a massive decrease in the attack-success-rate due to the Seam Carving method, it clearly shows that the method is a valid defense mechanism to combat against N-pixel-attack.

This paper would be helpful for the future audience mainly because we provide a more robust framework for face mask detection by incorporating seam carving as a defense mechanism. This pipeline can be easily applied to real-world situations when there is a patch or a paint spot on people's masks and may cause the deep learning models to misclassify.

## 8 Future Plans

In this study, we designed a path to increase the robustness of a deep learning-based face mask detection model. It can easily be applied to related fields such as face recognition. In the future study, we would like to try more models to ensure the results are consistent across different models, such as lenet and CapsNet. Lenet was used for handwritten digit recognition and could easily outperform all other existing methods. The LeNet architecture was quite simplistic, with five layers composed of 5*5 convolutions and 2*2 max pooling, but it paved the way to better and more complex models. The LeNet model made hand engineering features redundant because the network automatically learns the best internal representation from raw images. The CapsNet is also known as Capsule Neural Network. Therefore, it is defined as a neural net architecture that profoundly impacts deep learning. CapsNet primarily works for computer vision. We also want to test different attacking parameters in more experiments if given more computational power. In addition, different attacking methods and defense method combinations can be conducted to make our method more robust. For example, we can further fine-tune the seam carving resizing by using pictures of people wearing masks. If given more time, we can validate our method and test its robustness using different real-world datasets such as transparent masks.

We also planned to utilize the second data set, which contains images with a group of people on top of individuals. The second dataset can be found with the following link: `https://www.kaggle.com/datasets/omkargurav/face-mask-dataset?resource=download`. This dataset contains around 4000 examples. It has three different classes: "with mask," "without a mask," and "mask worn incorrectly." Due to the limitation of computation power and time restrictions, we could not utilize the data set. However, training the model with this data set would help us train the model in real-life settings.

## References


S. Avidan and A. Shamir. Seam carving for content-aware image resizing. *SIGGRAPH*, 26, 07 2007. doi: 10.1145/1276377.1276390.

D. Chen, R. Xu, and B. Han. Patch selection denoiser: An effective approach defending against one-pixel attacks. In T. Gedeon, K. W. Wong, and M. Lee, editors, *Neural Information Processing*, pages 286–296, Cham, 2019. Springer International Publishing. ISBN 978-3-030-36802-9.

I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples, 2014. URL https://arxiv.org/abs/1412.6572.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015. URL https://arxiv.org/abs/1512.03385.

M. A. Husnoo and A. Anwar. Do not get fooled: Defense against the one-pixel attack to protect iot-enabled deep learning systems. *Ad Hoc Networks*, 122:102627, 2021. ISSN 1570-8705. doi: https://doi.org/10.1016/j.adhoc.2021.102627. URL https://www.sciencedirect.com/science/article/pii/S1570870521001499.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Z.-Y. Liu, P. S. Wang, S.-C. Hsiao, and R. Tso. Defense against n-pixel attacks based on image reconstruction. In *Proceedings of the 8th International Workshop on Security in Blockchain and Cloud Computing*, SBC '20, page 3–7, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450376099. doi: 10.1145/3384942.3406867. URL https://doi.org/10.1145/3384942.3406867.

N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 582–597, 2016. doi: 10.1109/SP.2016.41.

D. K. Peng Wang, Zhipeng Cai and W. Li. Detection mechanisms of one-pixel attack. *Wireless Communications and Mobile Computing*, 2021. URL https://doi.org/10.1155/2021/8891204.

S. A. A. Shah, M. Beugre, N. Akhtar, M. Bennamoun, and L. Zhang. Efficient detection of pixel-level adversarial attacks. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 718–722, 2020. doi: 10.1109/ICIP40778.2020.9191084.

J. Su, D. V. Vargas, and K. Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, oct 2019. doi: 10.1109/tevc.2019.2890858. URL https://doi.org/10.1109%2Ftevc.2019.2890858.

D. V. Vargas and J. Su. Understanding the one-pixel attack: Propagation maps and locality analysis, 2019. URL https://arxiv.org/abs/1902.02947.


## A Appendix

Optionally include extra information (complete proofs, additional experiments, and plots) in the appendix. This section will often be part of the supplemental material.