NOT-MIWAE
PGM-DGM PROJECT REPORT

Emilien BIRE
emilien.bire@student-cs.fr

Damien BOUET
damien.bouet@student-cs.fr

Gabrielle CAILLAUD
gabrielle.caillaud@student-cs.fr

1 INTRODUCTION

This project reviews and implements the not-MIWAE (Not Missing at Random Importance Weighted Autoencoder) model Ipsen et al. (2021), a novel approach to handling missing data in machine learning tasks. The not-MIWAE model builds upon Importance Weighted Autoencoders (IWAE) Burda et al. (2016) and MIWAE Mattei & Frellsen (2019), with a specific focus on addressing the challenges posed by Missing Not At Random (MNAR) data.
In this project, we re-implement MIWAE in a pytorch code, and we run new experiments :

- New types of missing not at random data on a UCI dataset
- New type of missing data on a new dataset, the CIFAR-10 dataset.

1.1 BACKGROUND ON MISSING DATA MECHANISMS

Understanding the mechanisms behind missing data is crucial for developing effective models. Three main categories of missing data are commonly recognized:

1. Missing At Random (MAR): The missing pattern is unrelated to the missing values themselves.
2. Missing Completely At Random (MCAR): The missing pattern is unrelated to both the missing and observed data.
3. Missing Not At Random (MNAR): The missing pattern is related to the missing values, requiring prior information about the distribution of the missing pattern.

The not-MIWAE model specifically targets the MNAR scenario, which is often more challenging but more realistic in real-world applications. In our experiments, we specifically tackle MNAR data.

1.2 KEY FEATURES OF NOT-MIWAE

The not-MIWAE model introduces several key innovations, in order to effectively handle missing not at random data:

1. A latent variable model that creates a stochastic mapping parametrized by θ from a latent variable $z \sim p(z)$ to the data $x \sim p_\theta(x|z)$, which is partially observable. This is the classic decoder part in a variational auto-encoder (VAE).
2. A missing model, that is a stochastic mapping from the data x to the missing mask $s \sim p_\phi(s|x)$. The missing mask s has the same shape as x and has binary values which represents the absence and presence of data. The architecture of the missing model can vary, the article tackles three different missing model architectures : self-masking, self-masking known - imposing the weights to be positive- and linear (called *agnostic* in the paper).
3. The not-MIWAE model aims, like the MIWAE and the IWAE model, to maximize a lower bound of the joint likelihood. The difference is that the lower bound used is slightly different. Compared to MIWAE, the not-MIWAE model adds a reparametrization trick in the data space due to the missing model to the MIWAE model.

1.3 EXPERIMENTS IN THE NOT-MIWAE PAPER

The authors of the not-MIWAE paper conducted extensive experiments to validate their approach:

- **Metrics:** Root Mean Square Error (RMSE) was used to evaluate the model's performance.
- **Baselines:** The model was compared against established methods such as missForest, MICE, and mean imputation.
- **Datasets:**
  1. UCI database: MNAR conditions were simulated by self-masking in half of the features (entries higher than the feature mean were marked as missing).
  2. SVHN images: Data was clipped if pixel values exceeded 0.75.
  3. Yahoo! R3: A real-world MNAR dataset was used to test the model's performance in practical scenarios.

This project aims to review the not-MIWAE approach, implement key components, and critically analyze its performance and potential improvements.

2 OUR CONTRIBUTIONS

In this review and implementation project, we aim to extend the work presented in the original not-MIWAE paper. Our contributions focus on two main areas:

- Re-implementation of the model in pytorch
- Exploration of new type of missing data MNAR on a UCI dataset
- Exploration of new dataset : the CIFAR-10 dataset, with the implementation of a Convolutional MIWAE.

3 RE-IMPLEMENTATION OF MIWAE IN PYTORCH

In this section, we present our first major contribution, that is the reimplementation of the source not-MIWAE model[1] using PyTorch. The original implementation was done in TensorFlow 1.0, which has since become outdated. Our work to modernize and enhance the code base aims to make it more accessible and understandable, and leverage PyTorch's dynamic computation graph for potentially more flexible learning experimentation. Providing an up to data code base helps to validate the reproducibility of the original results. The reimplementation of the model has also allowed us to gain deeper insights on how the model works.

3.1 MAIN CHALLENGES TACKLED

In our implementation in Pytorch, we tried different distribution for the observation distribution. The Bernouilli distribution was present in the original tensorflow implementation -though no result using this distribution was presented in the paper-, but we weren't able to use it in pytorch. Sampling from a discrete distribution isn't differentiable, so pytorch's autograd can't propagate any gradients through the Bernouilli sampling step.

In the original code, the input data was normalized using the mean and the standard deviation computed on both the train and validation data, which can cause data leakage and could produce better val / test results than what we should get. We specifically paid attention to this issue and clearly separated train and validation / test data in our experiments. However, as the original authors righteously note in their paper, there is still a form of data leakage when computing test results, because the test and train data's missing values have been generated not at random and following the same pattern. Therefore, we try in our experiments to minimize information leakage between train and val/test as much as possible.

[1] Available on github here : https://github.com/nbip/notMIWAE

4 RESULTS ON UCI CANCER DATASET WITH NEW TYPE OF MISSING DATA

Since medical data can often have missing data, for example because of doctors or patients not filling every line of a form due to lack of time, we found interesting to use a UCI medical dataset in our first experience. Therefore we chose the Breast Cancer UCI dataset. This dataset was also used by the original authors, so we could compare our results with theirs. We trained our models for much less iterations, and got relatively similar results.

For all the results presented in this section, we used each time the same hyperparameters in order to effectively compare the performances. We used $lr = 10^{-4}, epochs = 500, batch\_size = 32$. We used a hidden dimension of $n_{hidden} = 128$, and a latent dimension of $n_{latent} = 10$. We trained our models with the Adam optimizer and the learning rate scheduler OneCycleLR from the Pytorch library. We used $pct\_start = 0.2, final\_div\_factor = 1e4$ as parameters for OneCycleLR.

4.1 FIRST TYPE OF MISSING DATA

We first tried, as it was tested in the paper, to introduce missing data by removing at randoms values superior to the mean on some features. The results are presented in 1. We observe that the RMSE is consistently higher when we use a Student distribution rather than a Gaussian distribution.

Table 1: Results on cancer data with first missing data type : removing values superior to the mean. Best validation RMSE achieved during training.

Missing data model	**Gaussian distribution**	**Student t distribution**
self-masking	1.127	1.180
self-masking known	1.195	1.260
linear	0.868	0.896
non-linear	0.862	0.873

In this experiments, the results are better when using a non-linear missing model. Curiously, results using a non-linear missing models are not described in the original paper, though its implementation is present in the original code on GitHub. We also found that between the three missing models self-masking, self-masking known and non-linear, the one that performed the best was linear, which wasn't the case in the paper. Furthermore, we note that the resuls are consistent when the output distribution changes : the non-linear missing process is the best one whether the distribution is Gaussian or Student t.

4.2 SECOND TYPE OF MISSING DATA

We also tried a new type of missing data : missing data is introduced by removing, on half of the features, the 25% lowest and the 25% largest values. The results are presented in 2.

Table 2: Results on cancer data with second missing data type : Removing extreme values

Missing data model	**Gaussian distribution**	**Student t distribution**
self-masking	1.683	1.826
self-masking known	1.895	1.905
linear	1.415	1.501
non-linear	1.512	1.538

It seems that the model struggles more to predict accurately the missing values for this type of missing data, since the RMSE values are larger. This is to be expected: when we remove all the lowest and all the largest values, the model will only see values around the mean and therefore will not predict lower or larger values. In this experiment, we get the best results for the linear missing model. This might be because the linear model is more capable of predicting values that are away from the mean. The results are consistent when changing the output distribution: the linear is

the best model for both tested output distribution and the self-masking-known produces the largest RMSE for both distributions.

4.3 THIRD TYPE OF MISSING DATA

We tried a third type of missing data not at random: Removing a specific percentage of values around the mean. For our experiments, we took $p = 0.30$. The results are presented in 3 .

Table 3: Results on cancer data with third missing data type: Removing values around the mean

Missing data model	**Gaussian distribution**	**Student t distribution**
self-masking	1.288	1.303
self-masking known	1.438	1.326
linear	1.097	1.209
non-linear	0.925	0.960

We see that the best missing model for this missing data type is the non-linear one.
As it is the case for the experiments on the other missing data, the self-masking and self-masking known models perform worse than linear and non-linear. The number of weights in the missing decoder for the self-masking and self-masking known model is twice to the number of features in the dataset, so there is only very few parameters. For the cancer dataset, it is 58 parameters. The non linear missing model we tested here has 7581 parameters, and the linear one has 870. Therefore, what the original paper failed to mention is that there is a huge difference in the number of parameters in the missing data decoder. This explains why, in out experiments, the selfmasking and selfmasking known missing models perform worse than the linear and nonlinear ones.

5 EXPERIMENTS ON CIFAR-10 DATASET

5.1 DESCRIPTION

In our experiments, we tried a new dataset, the CIFAR-10 dataset, and a new type of missing data not a random. On these RGB images, we introduced missing data by comparing the values of the three color channels for each pixel. If the blue channel value was superior to the red's and the green's, we put the blue channel to zero. This reflects some realistic scenarios of information loss, for example when a sensor has trouble encoding specifically one RGB channel if the values are too high. In the original paper, the authors were clipping values in grayscale images, so this is a new MNAR process. Some pictures and some classes specifically will have more missing data than others. An illustration of this can be found in the Appendix on figure 3. This is also due to the *not at random* property of the missing data we introduced: we cannot expect that every sample will be affected the same way.

We tested two types of VAEs: one with dense layers and the other with convolutional layers. As expected, the one with convolutional layers produces better results on the CIFAR-10 dataset. In the paper, the authors also used a convolutional structure when working with images. Since we used pixel values in the range [0,1], we added a sigmoid activation function at the end of the decoder. Due to this simple difference in the architecture, the RMSE absolute values are much lower for the experiment on the CIFAR-10 dataset than on the Breast cancer dataset.

5.2 RESULTS

We present here our best results on the CIFAR-10 dataset. The results on figure 1 were obtained using a convolutional not-MIWAE, using the hidden-channels 64, 128 and 256 for both the encoder and the decoder, and filter widths of 3. We also used a learning rate of 0.001 and trained the model for 100 epochs. Our best results on CIFAR-10 were obtained using the linear missing process, and we have an RMSE on validation data of 0.1485. Other results using different missing process are presented in details in the table 4.Each time, for the imputation, we use 10 importance samples.

We tried to test the baselines used in the original paper: missForest and MICE on our CIFAR-10 missing data, but due to computational reasons (out of memory error) we weren't able to produce any results. These baselines are indeed coded in the scikit-learn library and cannot run on GPUs. We tried to scale things down by using small CIFAR-10 datasets but that didn't work. Since an image from CIFAR-10 is of size (3,32,32), that makes 3072 features per sample and our theory is that it is too big for the baselines and the CPUs we had at our disposal.

However, we still tried the mean imputation baseline: replacing missing blue pixel values by the mean over the existing blue pixel values on the same image. We got an RMSE of 0.254, as presented in 4.

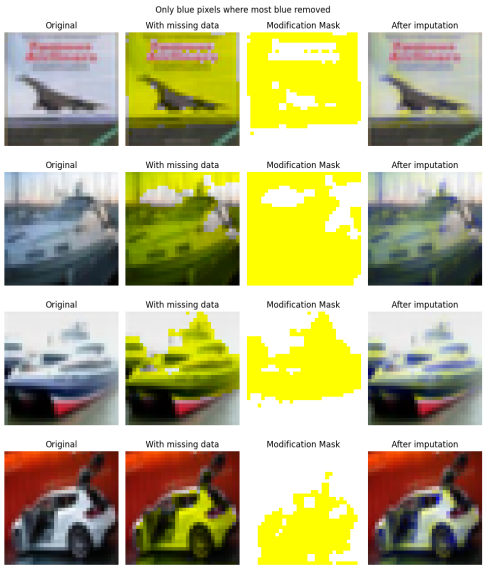

Figure 1: Results on test data on the CIFAR-10 dataset, using a linear missing process.

5.2.1 Interesting findings

To our surprise, we found that the convolutional VAE used for our experiments was highly dependent on the batch size when training on the CIFAR-10 dataset. Large batch size (larger than 16) was producing bad results, regardless of other hyperparameters. To have good results, and to compare the influence of other hyperparameters, we used a batch size of 8. We plotted the validation RMSE during training for different batch sizes in the figure 2. We don't have a clear explanation on why the batch size is so important. It could be that if the batch size is large then the model isn't able to determine the subtleties between each classes or each image and predicts the average blue intensity on the dataset. That produces worse results than the mean baseline.

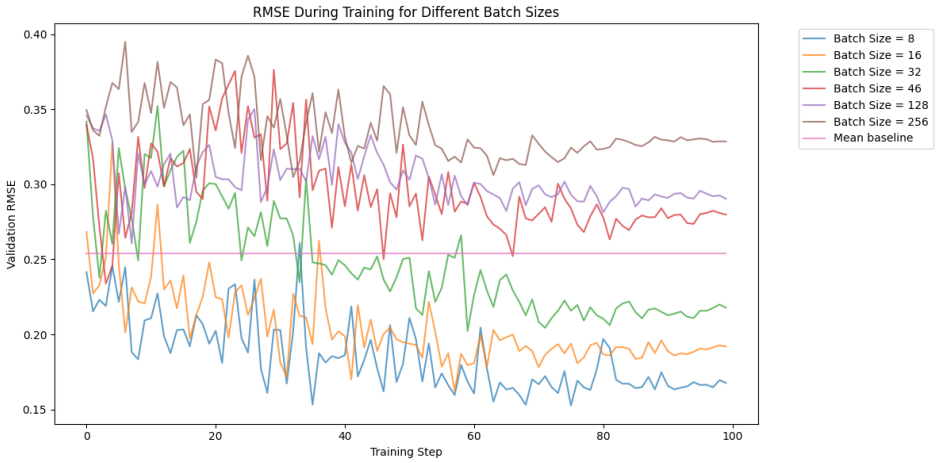

Figure 2: Validation RMSE during training for 100 epochs for multiple batch sizes.

5.2.2 Comparison between missing models

Results are shown in 4. The self-masking missing process achieves similar results to the missing process made of two convolutional layers. Different missing processes involve different numbers of trainable parameters. We displayed on the table the total number of trainable parameters for each missing process. As we can see in 4, the best validation RMSE is achieved with a linear

missing model, but this model has approximately five times more trainable parameters than the others models. This higher accuracy can also be due to the fact that the CIFAR-10 dataset is actually simple, that the train and validation dataset are alike and therefore the model overfits. For the non-linear missing model, we used a hidden dimension of 512. In the missing model, the input and output size is large : (3,32,32) for the Cifar-10 dataset. This explains why the linear model we used has more parameters than the non-linear one. In the table 4 below, we display the best validation rmse during training. We see that our models perform better than the mean baseline. An examples of images reconstructed with out models and the mean imputation can be found in the Appendix in figure 5

Table 4: Results on CIFAR-10 when removing blue pixels not at random

Missing data model	**RMSE on validation data**	**Number of weights in the model**
self-masking	0.1526	2,616,262
self-masking known	0.162	2,616,262
convolutional	0.1597	2,610,202
2 convolutional layers	0.1525	2,638,281
non-linear	0.166	5,759,430
linear	0.1485	12,050,374
Replacing missing values with mean	0.254	-

5.2.3 Limits of the Not-MIWAE model

The results presented above on the CIFAR-10 dataset are all obtained using missing data on the blue pixels. We also tried to introduce missing data on the other RGB channels, red and green, also by removing the values when the other channels had smaller values. Some results obtained by introducing missing not at random data on the red channel can be found in the figure 6. The results are similar as with missing blue pixels.

Since the results were similar, we wondered if the models trained on different missing colors behaved differently and if they showed some interesting characteristics when used on a missing color they were not trained on. The results are shown in the figures 5 and 4 in Appendix. As it could be expected, the zero-shot inference on new missing data produces rather bad results, almost leaving untouched the missing pixels. These experiments show that when training a model to predict missing not at random data, the model is inherently dependent on the missing data. It has strong difficulties to generalize to new missing data even if the new missing data and the training missing data very much alike.
This raises the question of the utility and scalabity of the not-MIWAE model: to use it, we would have to train it on large amount of data for which we know the ground-truth values of what is missing; and use it on a dataset for which we know for sure the missing data has exactly the same pattern as the training missing data. In practice, having these two conditions is rarely the case. Often, we cannot know for sure that the missing pattern is the same.

6 Conclusion

In this project, we focus on the paper that introduced the Not-MIWAE model for missing not at random data Ipsen et al. (2021). We re-implement this model in pytorch for reproducibility and clarity reasons. We run some experiments on new types of missing data on a UCI dataset, and run experiments on the CIFAR-10 dataset. These experiments on the CIFAR-10 dataset show the limits of the not-Miwae model: it struggles to generalize to similar but different types of missing data.

6.1 Contributions statement

In our project, we split the work equally. Emilien implemented the general structure of the model, Damien ran experiments on the UCI dataset, and Gabrielle was in charge of the experiments on the CIFAR-10 dataset.

REFERENCES

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance Weighted Autoencoders, November 2016. URL http://arxiv.org/abs/1509.00519. arXiv:1509.00519.

Niels Bruun Ipsen, Pierre-Alexandre Mattei, and Jes Frellsen. not-MIWAE: Deep Generative Modelling with Missing not at Random Data, March 2021. URL http://arxiv.org/abs/2006.12871. arXiv:2006.12871.

Pierre-Alexandre Mattei and Jes Frellsen. MIWAE: Deep Generative Modelling and Imputation of Incomplete Data, February 2019. URL http://arxiv.org/abs/1812.02633. arXiv:1812.02633.

A APPENDIX

A.1 RESULTS ON THE CIFAR-10 DATASET

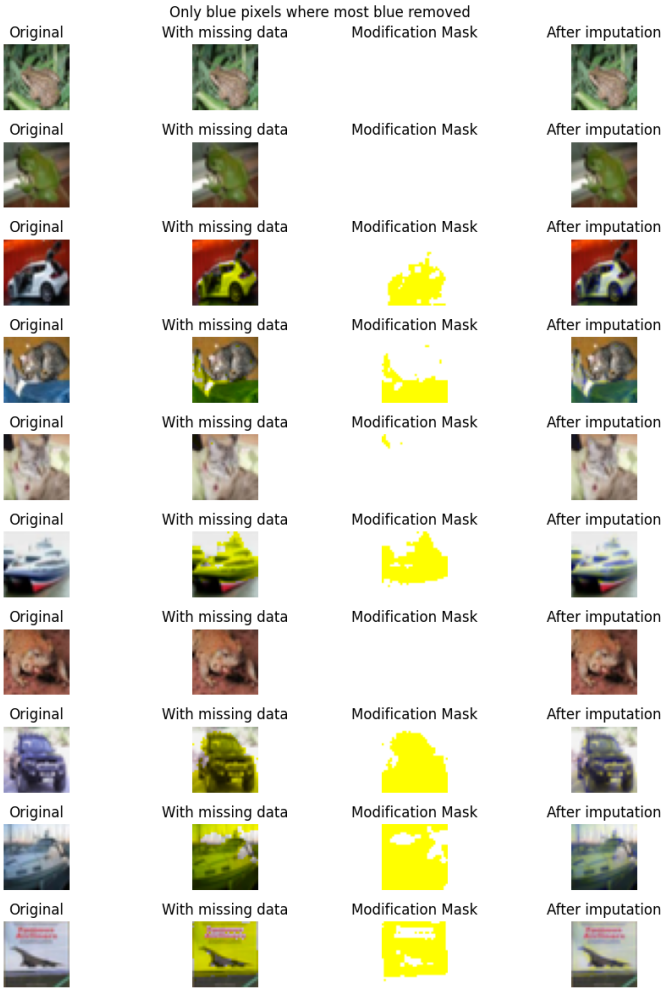

Figure 3: Results on some samples of the CIFAR 10 dataset using a linear missing model

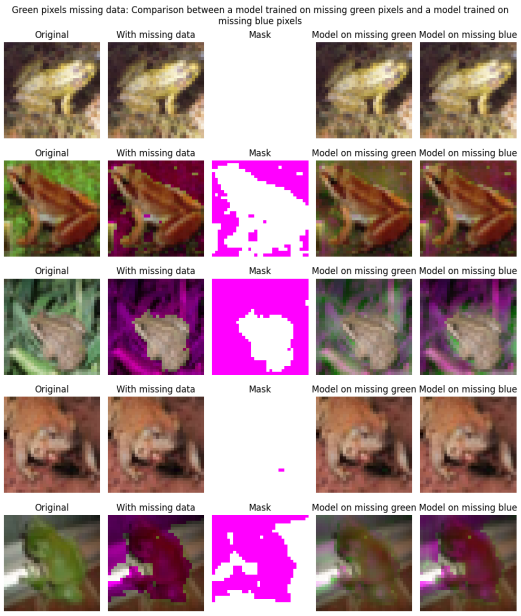

Figure 4: Comparison between a model trained on missing green data, evaluated on missing green data, and zero-shot inference on a model trained on missing blue data. Only selecting the frog class, which is more likely to have missing green data than the other classes

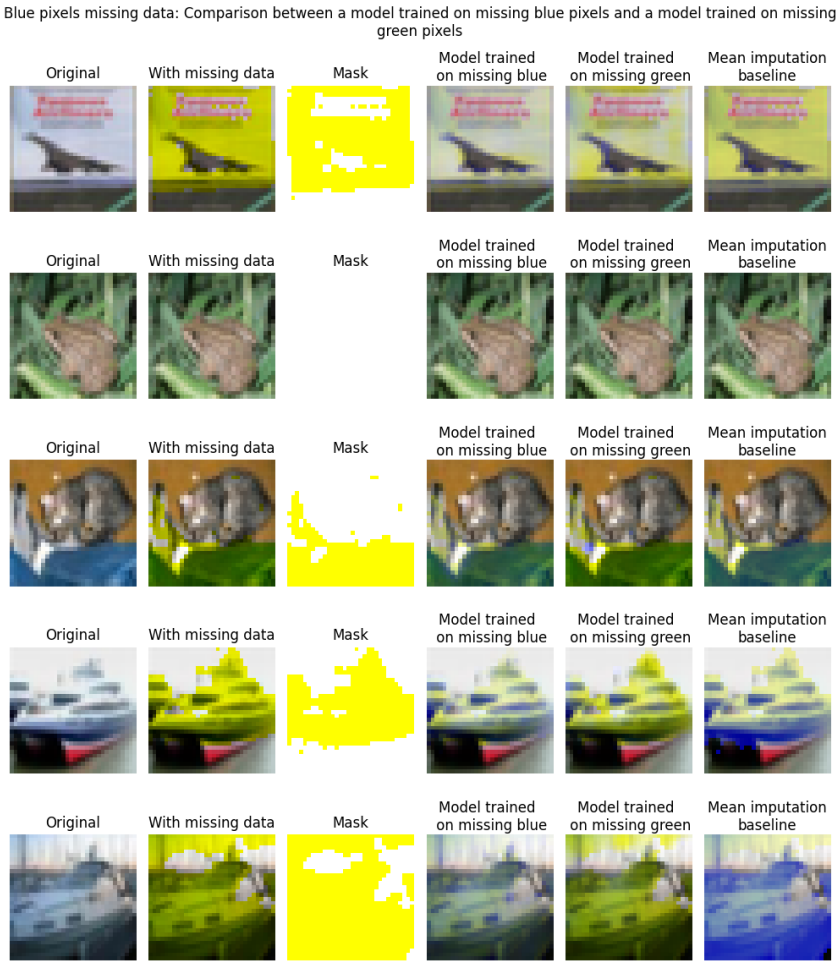

Figure 5: Comparison between a model trained on missing blue data, evaluated on missing blue data, and zero-shot inference on a model trained on missing green data. Comparison with baseline mean imputation.

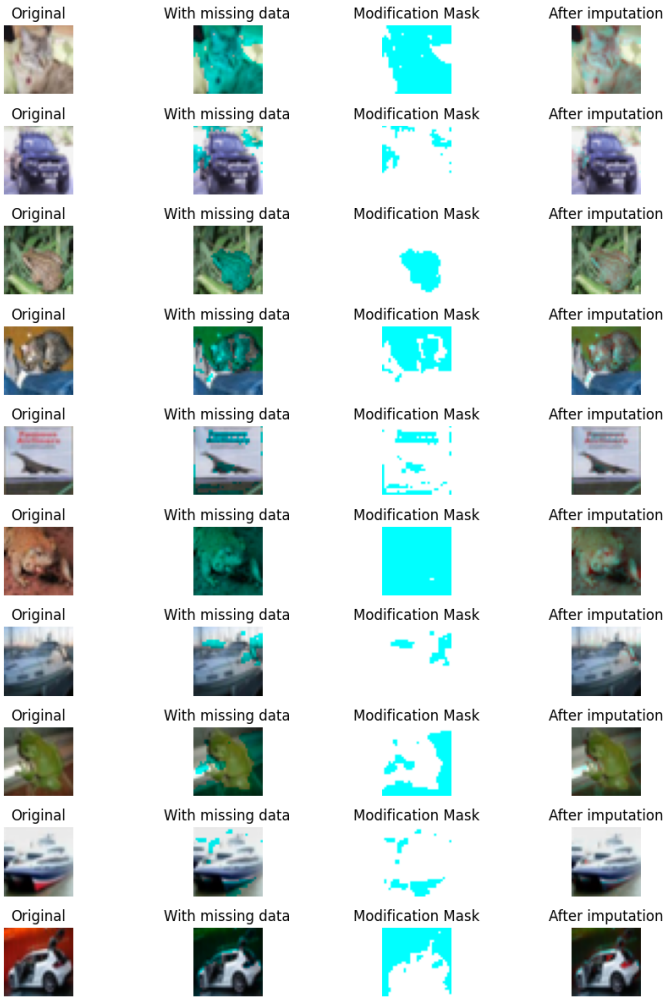

Figure 6: Results when removing red pixels when red is the most intense color. Using the selfmasking missing process