Neuro 240 Final Report: Strategies for Day/Night Image Classification Generalizability

Andrew Palacci
School of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138
`andrewpalacci@college.harvard.edu`

Abstract

Convolutional neural networks have become widely used throughout the field of computer vision, proving themselves to be particularly applicable to the tasks of image classification, segmentation, and object detection. However, the vast majority of commonly-used image datasets include do not include lowlight images. As a result, models that use these datasets for training and/or pre-training become very biased towards images in near-ideal lighting conditions. This leads CNNs trained on these data to struggle to generalize well to lowlight/night-time images, as these are far outside of the distribution of their training data. Modern solutions to this problem are data augmentation and nighttime image recreation, but these are compute-intensive and require additional storage. Thus, the goal of this project was to find space- and time-efficient data transformation regimes that consistently lead to improvements in generalizability for lowlight images, while also examining explanations for why such transformations are (or aren't) successful. I found support for the use of RandomBrightnessContrast transformations in some contexts, but my main finding was a reaffirmation of augmentation/nighttime image recreation as preferred strategies for CNN generalizability.

1 Introduction

As humans, we are fundamentally biased towards brightness — we tend to sleep during the nighttime, and we act during the daytime. During these times, we use different types of vision — taking inspiration from human vision literature, we can call daytime vision *photopic* and nighttime vision *scotopic* [1]. With photopic vision, an abundance of light exists, and thus an accurate image of an object can be formed very quickly. This reflects the format of images on CIFAR-10 — detailed, relatively bright, and mostly noiseless. However, in many real-life situations, this brightness and noiselessness is not the case — driving at night is one such salient example. For humans and other animals, this is not an issue as a result of our minds' incredible ability to generalize: to consider our knowledge of different lighting conditions, capture tiny amounts of information at night, and convert it to a rich mental representation extremely quickly. Unfortunately, the same cannot be said about computers. In these low-light situations, image classification, segmentation, and object recognition are problems that are still being consistently advanced in modern computer vision (CV) literature. Furthermore, as CIFAR-10 illustrates, equivalent training data oftentimes doesn't exist for nighttime situations,

[1]From Chen & Perona: "The term 'scotopic vision' literally means 'vision in the dark'. It is usually associated to the physiological state where only rods, not cones, are active in the retina. We use this term to denote the general situation where a visual system is starved for photons, regardless the technology used to capture the image." [1]

thus leading to applications of these models (e.g. self-driving cars or UAVs [2]) facing very out of distribution (OOD) data during test time, such as pedestrians walking at night. Typical solutions to this problem have included simply gathering more nighttime data, as seen in BDD100K, recreating nighttime images using CycleGANs (as with Musat et al.'s work), or at the very least augmenting daytime datasets using a variety of more straightforward augmentations [3, 4, 5].

Despite these modern and effective techniques, scenarios also arise where efficiency is crucial, not only in the sense of time but also storage space. For instance, work in tiny machine learning is becoming far more prevalent on consumer devices — one such example is security cameras. Although security cameras for detecting humans (and potentially their activity) certainly come with models pretrained on rich datasets, these cameras can benefit from applying tinyML to fine-tune on the contexts in which they are installed. Unfortunately, this reveals a clear issue — the security camera will only encounter people that are not the home/storeowners during the daytime, since stores are dark when they are closed and homes are dark when their owners are asleep. Thus, alternative methods to help the tinyML model generalize are needed — ideally, the camera would connect to remote servers to recreate nighttime images and refine its model weights, but it would still be useful for the camera to have some standalone training and generalization ability. This is only one of many instances where my motivating question proves relevant: are there simple, interpretable, efficient transformations that make a noticeable difference in CNN generalizability?

When answering this question, I tried to focus primarily on a few key ideas: accuracy, efficiency, and validity. Firstly, accuracy is obviously crucial — especially in cases such as nighttime self-driving or security imaging, even small increases in classification or detection accuracy can save lives. Next, efficiency is very important — in an age where leading companies and research groups have immense compute resources and infrastructure, it's important to remember that most real-world applications of CV will not have this same fortune. Finally, validity (the concept I grappled with most in this project) is important when an abundance of similarly-distributed testing data does not exist — how can we make sure our testing data is representative of what the model will *actually* see?

My findings faced numerous technical challenges, but for the most part reinforced knowledge that is perpetuated in modern CV literature. Although the RandomBrightnessContrast transformation showed fairly promising results, and other transformations occasionally did as well, these improvements were marginal when compared to other strategies put into place by papers such as Musat et al.'s [4]. In summary, I was able to confirm that simple `albumentations` transformations are indeed *efficient*, but in terms of *accuracy* they were not as helpful as hoped. Furthermore, I had trouble ensuring the *validity* of my results due to a lack of realistic test data.

2 Related Work

WaldNet: an architecture to make fast scototopic decisions After a thorough scraping of various academic databases, it seems like this 2015 work by Bo Chen and Pietro Perona from Caltech is the only published paper that *specifically* analyzes improvements on low-light generalizability for deep neural networks [1]. The paper uses a custom recurrent convolutional network architecture, and also forms a probabilistic and biologically plausible interpretation of what "low-light" even means in terms of photons per pixel (PPP). Although this method lacks the external validation of modern models such as Inception v3 (cite here) and other generative adversarial networks, it still formed a strong (and more easily interpretable) basis for day-night generalizability. Furthermore, although this probabilistic interpretation might not define what humans perceive at night, it might be a more accurate model of what lower-resolution *cameras* perceive in lowlight conditions.

Multi-weather city This paper is a more recent work that uses CycleGANs to layer snow, rain, and darkness on top of overcast weather conditions, motivated by improving generalizability for self-driving car algorithms [4]. It provides a strong foundation for how GANs can augment training data to make image classification / object detection models more robust. It also helped me learn how to benchmark and interpret my results (even though my experiments focused only on classification).

Additional work The above two papers will be foundational for the rest of this project, but I also have taken inspiration from Ye et al.'s work on UAV tracking augmentations and Cowley & Willow's paper on gaudy images to decide some of the augmentations I would like to test [2, 6]. Furthermore, I used Shorten & Khoshgoftaar's survey paper to form a better understanding of the broad environment of data augmentation and transformation — the most efficient methods covered were kernel, color space, and geometric transformations, which led to my choice of experimental data transforms [5].

3 Methods

For this project, I used Google Colaboratory to develop a Jupyter Notebook in Python that contains all of my required code. I used a tutorial from the PyTorch website to develop my training scheme and occasionally found inspiration from other websites, which I also cited within the code. However, the vast majority of the code written was my own, or highly modified — notably, the night-time image recreation methods, model and loader generation, and dataset formats are all entirely original. The code for this project can be found at https://github.com/andrewp2303/day_night_generalizability.

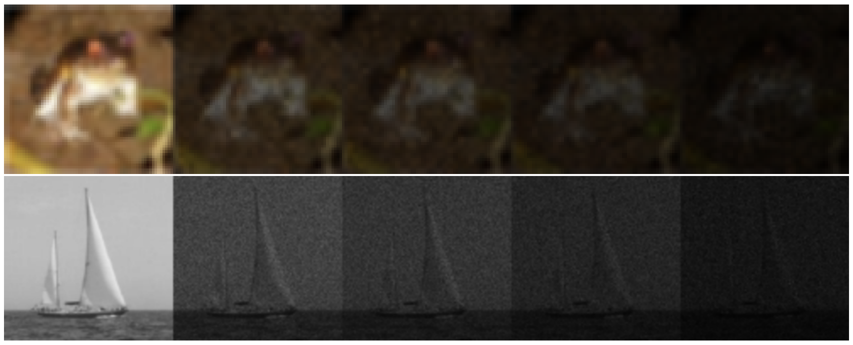
3.1 Datasets, Models, and Other Context

In order to ensure that our transformations were not specific to one (relatively constrained) dataset, I performed each step of our experiment on both the CIFAR-10 and Caltech101 datasets, available through the `torchvision` Python library [7, 8]. These data are pre-labeled and have built-in loading support, thus allowing me to focus more on the generalizability strategies themselves.

For the CIFAR-10 dataset, I did not preprocess the data in any way, since it is composed of only 32x32 color images. However, the Caltech101 dataset is composed of images in both color and grayscale with various dimensions, and thus I first cropped and resized images to 100x100 (maintaining most of their semanticity), before converting all images to grayscale for standardization. This allowed me to have one dataset of lower-quality color images, and one of higher-quality grayscale images — this is relevant in the context of security cameras, which often use grayscale for more affordability and also nighttime efficiency.

For my model selections, I chose two CNN architectures that are applicable and frequently used for image classification, but also easily accessible using `torchvision` and not overly large: Resnet50 and the more modern ResneXt50 [9, 10]. These both used weights from pre-training on ImageNet, in order to shorten the needed training time for the models to converge.

3.2 Nighttime image generation



In order to simulate night-time images for testing, I relied on strategies covered by Chen & Perona, that simulated camera-captured lowlight using simple probabilistic methods [1]. This primarily involves using a photon per pixel (PPP) value to determine the amount of light that each pixel in an image should present. Decreased PPP represents a greater level of darkness — thus a PPP of 0 would be a true "pitch black." This simulates the way that low-light images are captured by cameras, and can be extended to explain why longer exposure times are useful for gathering detail in nighttime images. For those interested in the technical foundations of the method I described, I encourage exploring Chen & Perona's paper further.

In my nighttime image recreation, I aimed to randomly give each photo a photon per pixel (PPP) value of 0.22, 2.2, 22, and 220, respectively, exactly replicating the methods seen in [1]. However, these relied on constants I wasn't able to access/implement, so I instead used my knowledge of the Poisson distribution along with Chen & Perona's sample images in order to create my own night image function, denoted as `make_night_img` in my codebase, taking in an image and a PPP value. Then, I formed another function, `random_night_img`, that uniformly sets an image to one of 4 levels of darkness — all possible outputs of this method can be seen above for both the CIFAR-10 dataset (top) and the Caltech101 dataset (bottom), with the original images displayed at the far left. Although I was not able to validate this method with any external metrics, I found the produced images very visually and characteristically similar to the sample images provided by Chen & Perona. Finally, it's important to note that these images (especially those on the darker end) are significantly less informative than the original images. This was intentional: since the entire test set is composed of these night-time images, I wanted to provide difficult test data in order to broaden the range of possible test accuracies for each transformation.

3.3 Data transformation experiments

In each of my experimental conditions, I tested exclusively on nighttime images formed using my `random_night_img` method, which returns an image at one of 4 different darkness levels with uniform probability. Furthermore, I used a simple evaluation metric for every experiment: percentage of images correctly classified (e.g. Top-1 accuracy) — when recalling my motivating question, cameras should be able to detect (e.g. classify) humans, and anything besides a correct classification is not beneficial. Lastly, I trained the model for 20 epochs in each experiment — by experimentation, this seemed to be around the point where my models stopped truly training and began overfitting instead.

In order to benchmark my results, I first formed a **Control** condition — this condition simply converts images to a tensor, with no other transformations. For each dataset-architecture pairing, this condition tells me *at least* how accurate a transformation should be to even consider using.

For the next 5 experimental conditions, I used the Albumentations `augmentations.transforms` library to perform pixel-level data augmentations in hopes of improving our models' generalizability [11]. Specifically, the transformations I used involved Gaussian noise (**Gauss**, `GaussNoise()`), brightness shifts (**B**, `RandomBrightness()`), contrast shifts (**C**, `RandomContrast()`), and contrast+brightness shifts (**BC**, `RandomBrightnessContrast()`). These made sense to me as initial transformations for testing, because nighttime images are typically associated with different levels of brightness, contrast, and noise than daytime images. As the fifth condition, I also added a normalization transformation (**Norm**, `Normalize()`), which should distort the images and result in far worse test accuracy, confirming the robustness of my results. *After* each of these transforms, I converted images to tensors, for consistency with the control condition.

Finally, I also created a method (`random_PPP`) to randomly set PPP as specified above, exploiting our knowledge of photon perception to mimic different lighting scenarios as training data (**RandPPP**). One important note is that this models for uncertainty over what a nighttime image would actually look like, while the aforementioned `random_night_img` models ground-truth nighttime images. As a result, I made sure that although these have to be distributed similarly (since I had no other mechanism to form test data), images formed by `random_PPP` are not IID to those formed by `random_night_img`.

There are a few important notes to make about these experiments. First, I used data *transformations*, not *augmentations* — this means that the original dataset images, which may have been semantically useful, were not also part of the training set. This poses the question for future experimentation of whether transformations should only be applied with some (potentially low) probability. Additionally, this means that in the cases of **B, C, BC, Gauss, random_PPP**, I effectively enlarged the dataset to size $n_{epochs} \cdot dataset\_size$, since each image will look different during each epoch by virtue of randomness. Finally, these experimental methods test for generalization *irrespective* of image class — I am not looking to examine whether learned characteristics of nighttime images from one class carry over to another, just whether certain transformations applied to the training set are indeed useful when the model tries to generalize on nighttime images during testing.

4 Results

Below are the results for the *accuracy* metric, as mentioned at the beginning of the paper. These reflect the accuracy of a model using the given architecture on either CIFAR-10 or Caltech101, tested on a subset of the data that has not yet been seen and has been converted to a night-time image. Finally, I was only able to gather results for the **RandPPP** condition on the CIFAR-10 dataset, because training on Caltech-101 for that condition proved to be too intensive — my Colab runtimes disconnected every few hours, preventing me from running these tests to consistency with the others.

Table 1: Data Transformation Test Accuracy (%)

Transformation	CIFAR-10		Caltech101	
	ResNet50	ResNeXt50	ResNet50	ResNeXt50
Control	35.39	38.55	60.08	57.32
B	32.92	22.22	59.91	58.35
BC	39.56	40.45	57.95	66.94
C	35.72	30.97	58.29	55.24
Gauss	25.65	40.61	52.94	53.40
Norm	11.65	10.00	0.630	0.350
RandPPP	82.76	77.94		

In the experiments I detailed in the previous section, I was only able to explore how transformations performed on nighttime images created with the heuristics that [1] laid out. One key finding that Chen & Perona made was that it's crucial to include the "night-time" data in training sets, which I was able to simulate using the **RandPPP** condition. This was a reproduction I hoped to make, and it gave me slightly more confidence in the validity of the rest of my results. This also proved that nighttime image recreation — especially when the recreated images are similar to the ground-truth images the model will be applied to — is intensive but valuable.

However, there are many other interesting notes to take in terms of the *accuracy* metric shown above. One key takeaway is that random brightness and contrast shifts (**BC**) resulted in significant improvements from the control in every case except one — and in that case, the control was not much more accurate. Thus, it appears that `RandomBrightnessContrast()` can reliably be used to improve generalizability, especially with some tweaking of its parameters, instead only reducing image brightness and only with a certain probability. For all of the other transformations, I saw results that slightly surprised me — I expected Gaussian noise and brightness/contrast shifts alone to be helpful, but they appeared to typically detract from model performance. As anticipated, normalization was the opposite of helpful (although it would certainly be more helpful if test data were also normalized).

Additionally, I also gathered results for efficiency — namely, the time that each of these methods took per epoch. Once again, I was only able to gather results for the **RandPPP** condition on the CIFAR-10 dataset. I rounded to the nearest second because additional precision was not needed, with all of the values far greater than 1 second.

Table 2: Data Transformation Epoch Training Time (s)

Transformation	CIFAR-10		Caltech101	
	ResNet50	ResNeXt50	ResNet50	ResNeXt50
Control	36	44	47	50
B	47	48	47	51
BC	46	48	48	50
C	46	48	47	50
Gauss	58	57	54	52
Norm	42	49	49	52
RandPPP	589	637		

As we can see above, all Albumentations transformations take approximately the same amount of time, with Gaussian noise being *slightly* slower. This coincides with why I chose them — I wanted them to be hardly less efficient than the control. On the other hand, **RandPPP** is slower than all other transformations by an order of magnitude — part of this is due to lack of optimization on my behalf, but this still highlights the time-accuracy tradeoff for the data transformations that we tested.

5 Discussion

Although this project did not meet the lofty goals of introducing efficient, interpretable, and helpful transformations for generalizability, I still found it to be very productive. For one, I learned not only about various architectures, datasets, and training schemes, but also about why day/night generalizability is important and how it's done in state-of-the-art models. Additionally, this project confirmed that data augmentations and image recreation are worthwhile — while running the **RandPPP** condition, I realized that although these methods increase the dataset size several-fold, they still result in far more efficient training than having attempting to recreate new images or perform complex transformations during training. Beyond that, viewing the outcomes of other related papers has shown me that the strategies of data augmentation and lowlight image recreation are certainly more accurate and valid, but also more efficient than at least some alternatives. This aligns with Madan et al.'s recent work, which claimed that data diversity (e.g. augmentation) is important for generalizability [12]. This makes sense in the paper's context of physical perspective, and would logically follow for lighting conditions as well — more diverse lighting conditions would prevent models from forming overly strong biases that only hold during daytime lighting. With more testing and external validation, expansions on the work done in this paper could even imply that this claim holds for day/night generalizability, as discussed above.

Another aspect of the results I found intriguing was the models' improved performance on the modified Caltech101 dataset, relative to their performance on the CIFAR-10 dataset. This could have resulted for several reasons — the most logical to me is that the images are simply larger, and thus encode more information. Even when nighttime images are formed, the larger (100x100) Caltech101 images are likely to have more bright/informative pixels, simply as a result of of PPP being distributed Poisson. However, an alternative explanation that could be interesting to test is whether multi-channel grayscale images provide higher generalizability than multi-channel color images. Although this seems illogical at first (it contradicts the idea of more information), it's also possible that CNNs trained on daytime images form too strong an association between color and class. As humans, we know not to do this — we generalize first based off shape, and then by color/brightness, in a phenomenon known as the shape bias [13]. However, CNNs don't have these same priors (and have actually been shown to form a texture bias), so when they are tested on nighttime images that lack color, it's possible that this missing color is what causes incorrect classification [14]. In grayscale, this effect is taken away, so CNNs can more easily generalize to nighttime images, unhindered by a learned association that is misaligned with what humans rely on in order to generalize well.

Finally, there are some potential implications as to nighttime image characteristics and how they relate to the transformations detailed. One example is that nighttime images likely result in variance in brightness and color together, as opposed to separately — this could explain why the **B** and **C** conditions underperformed relative to the control. Furthermore, Gaussian noise also had somewhat lackluster results — this could be for a similar reason as above, begging the question of whether random shifts in brightness, contrast, *and* noise would perform even better than the **BC** condition. Additionally, it's important to note that each of these transformations exists for a reason — normalization is not useless, and when applied in conjunction with other transformations and equivalently applied to the testing set, it's often very useful as a regularizing factor and results in higher accuracy and faster model convergence [15].

5.1 Challenges

As a student relatively new to machine learning, I (expectedly) encountered a long stream of challenges that I have yet to address with my project. First were the simpler challenges — I couldn't implement gaudy images as specified in Cowley & Pillow, or CycleGANs as seen in Musat et al., by virtue of not being able to find publicly-available and replicable code for their methods [6, 4]. Additionally, I was limited by my computational power in Colab, while running into several issues with runtime interruptions along the way — this is what led to not having full results for the Caltech101 dataset (although it's likely they would mirror the trends seen for CIFAR-10).

However, my larger, more difficult challenge was finding a way to tackle *validation*. To me, the most logical way to do this was to simply find a dataset that included daytime and nighttime images, and only present nighttime images during testing — after all, the best ground truth is simply a *real* nighttime image. For classification tasks, I was not able to find such a dataset. I came close with

Madan et al.'s Biased Cars dataset, but the images were not distinguishable between daylight and lowlight (unless I tried to filter them using a day/night classifier) [12]. Furthermore, for image segmentation, I found BDD100K, which has almost as many nighttime images as daytime, but even after several days of attempts I was unable to adequately build a framework for image segmentation training, especially since it's a large, high-quality dataset that isn't available through PyTorch [3]. Finally, I was able to recreate Musat et al.'s modified Cityscapes dataset, which used CycleGANs to make photorealistic nighttime images, but I ran into the familiar problems of image segmentation being difficult and the dataset training extremely slowly [4]. Although this challenge was particularly pernicious and I had to resort to my original methods, I was still grateful to gain more breadth in various CV tasks and how day/night generalizability can apply to them.

6 Future Directions

As hinted earlier, there are several intriguing directions in which the work done here can be taken. For one, there are a vast array of other transformations in the Albumentations library and other similar libraries that could prove more useful than any of the ones covered above. Additionally, the transformations already seen above could take different parameters — for instance, only applying transformations with a given probability, or only reducing brightness / decreasing contrast. Additionally, these data transformations could be compared to the equivalent data *augmentations*, which could reveal whether this distinction significantly affects accuracy and/or efficiency.

Additionally, the methods used in this project could be tested against real nighttime images (or other more realistic images than those generated by `random_night_img`). I'm curious whether the same results would hold with other test data, or whether my choice of nighttime image reconstruction greatly affected the transformations' performance.

Finally, as hinted to in the discussion, I think it would be particularly interesting to experiment with grayscale images — if they are in fact more accurate for classification tasks, this could prove to be a very useful transformation to make. Additionally, research could be done on changing the bias in CNNs to be more reflective of human priors, by somehow enforcing the development of a shape bias.

References

[1] B. Chen and P. Perona, "Scotopic visual recognition," in *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, 2015, pp. 659–662.

[2] J. Ye, C. Fu, G. Zheng, Z. Cao, and B. Li, "Darklighter: Light up the darkness for uav tracking," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 3079–3085.

[3] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving dataset for heterogeneous multitask learning," 2020.

[4] V. Mușat, I. Fursa, P. Newman, F. Cuzzolin, and A. Bradley, "Multi-weather city: Adverse weather stacking for autonomous driving," in *2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, 2021, pp. 2906–2915.

[5] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, p. 60, 2019. [Online]. Available: https://doi.org/10.1186/s40537-019-0197-0

[6] B. R. Cowley and J. W. Pillow, "High-contrast "gaudy" images improve the training of deep neural network models of visual cortex," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, ser. NIPS'20. Red Hook, NY, USA: Curran Associates Inc., 2020.

[7] A. Krizhevsky, "Learning multiple layers of features from tiny images," pp. 32–33, 2009. [Online]. Available: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf

[8] F.-F. Li, M. Andreeto, M. Ranzato, and P. Perona, "Caltech 101," Apr 2022.

[9] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” 06 2016, pp. 770–778.

[10] S. Xie, R. B. Girshick, P. Dollár, Z. Tu, and K. He, “Aggregated residual transformations for deep neural networks,” *CoRR*, vol. abs/1611.05431, 2016. [Online]. Available: http://arxiv.org/abs/1611.05431

[11] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, “Albumentations: fast and flexible image augmentations,” *ArXiv e-prints*, 2018.

[12] S. Madan, T. Henry, J. Dozier, H. Ho, N. Bhandari, T. Sasaki, F. Durand, H. Pfister, and X. Boix, “When and how convolutional neural networks generalize to out-of-distribution category–viewpoint combinations,” *Nature Machine Intelligence*, vol. 4, no. 2, pp. 146–153, 2022. [Online]. Available: https://doi.org/10.1038/s42256-021-00437-5

[13] B. Landau, L. B. Smith, and S. S. Jones, “The importance of shape in early lexical learning,” *Cognitive Development*, vol. 3, no. 3, pp. 299–321, 1988. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0885201488900147

[14] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel, “Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness,” *CoRR*, vol. abs/1811.12231, 2018. [Online]. Available: http://arxiv.org/abs/1811.12231

[15] F. Schilling, “The effect of batch normalization on deep convolutional neural networks,” 2016.

A Appendix

In addition to the tables included above, I’m including a more visual representation of the experimental data on both accuracy and efficiency in the form of a histogram. See the following two images below: