Image Pre-Processing Detection Using Deep Reinforcement Learning

Nitin Mukesh
HERE Technologies
Navi Mumbai, MH
India

***Abstract*— Reinforcement learning is a popular domain of Machine learning being actively used in many industries. With proper reward function and problem formulation, a task can easily be solved using Reinforcement learning.**

Computer vision domain often face the problem of noisy data. Even when the model is trained on good quality data, during inference, the noisy data can create problem and thus causing model to fail. An essential step to overcome the problem of bad data is pre-processing of data, but pre-processing of image data is itself a complex problem and requires subject matter experts to decide which kind of preprocessing should be applied on a given image for a particular compute vision task.

To solve this problem of choosing correct pre-processing for a given images, we have proposed a novel approach of automation of image pre-processing using deep reinforcement learning. This approach is developed and tested for one of the most popular problems in image data, which is noise in images, while it has also shown potential how it can change the whole scenario of machine learning models being applied in the field of computer vision.

***Index Terms*— Deep Reinforcement learning, OCR, Computer vision, Deep Q Network**

I. INTRODUCTION

Image noise is random variation of brightness or color information in images and is usually an aspect of electronic noise. Image noise is an undesirable by-product of image capture that obscures the desired information.

In the data collection phase, the digital images are captured using sensors that often contaminated by noise (undesired random signal). In digital image processing task, enhancing the image quality, and reducing the noise is a central process. Image denoising effectively preserves the image edges to a higher extend in the flat regions. Several adaptive filters (median filter, Gaussian filter, fuzzy filter, etc.) have been utilized to improve the smoothness of digital image, but these filters failed to preserve the image edges while removing noise [1].

Removing this undesirable by-product of image capture can be achieved by using Noise Recognition & Reduction. Noise Recognition is a process where a person tries to identify different kind of noise in images using various operations such as PSNR ratio and others.

Existing algorithms [11][12][13] are able to quantify noise but not the type of noise, while for noise reduction from an image, type of noise is also required. Because each type of noise is reduced differently from an image. For example: gaussian noise would be reduced differently than salt n pepper noise. Randomly, Application of various noise reduction techniques to document image leads to an overhead in computation which results in an overall decline in performance of integrated text detection system i.e. OCR pipeline.

Therefore, A solution to intelligently identify type of noises in a document image is needed to perform noise reduction in an efficient manner.

The proposed solution overcomes the problem of manually or randomly selecting noise reduction technique by identifying noises present in text images in a probabilistic manner. Current solution focuses on mainly two types of noises namely gaussian noise and salt & pepper noise.

As artificial intelligence technology continues to develop, reinforcement learning (RL) is evolving as a potent form of artificial intelligence. Reinforcement learning, as a subfield of machine learning, focuses on how to behave in a given situation in order to maximize the expected rewards. Due to the excellent perceptual and decision-making capabilities of RL algorithms, reinforcement learning has been widely used in various fields including medicine, finance, robotics, video games, and computer vision (CV)[2].

II. LITERATURE REVIEW

RL in computer vision is progressed a lot. There are various tasks for which the model can be easily stimulated using RL. Like this work [3] attempted to train an intelligent agent that, given a hyperspectral image, is capable of automatically learning policy to select an optimal band subset without any hand-engineered reasoning. The other paper [4] provides a novel approach for training Deep Decision Network for Multi-Class Image Classification. [5][6][7] used RL to for imbalanced data classification by appropriately constructing the reward function.

Identifying noise in images using deep learning-based model helps us by giving prior information on what pre-processing to apply on kind of noise. [8] investigate the DNN-based better noisy image classification system which experiment with different architecture like autoencoder, CNN based architecture etc.

A. Deep Q Network (DQN)

Deep Q-Learning is a variant of Q-Learning that uses a deep neural network to estimate the Q-function, rather than a simple table of values. This allows the algorithm to handle environments with many states and actions, as well as to learn from high-dimensional inputs such as images or sensor data.

The key technique to achieve stability in DQN is experience replay [9]. In specific, a replay memory is used to store the trajectory of the Markov decision process (MDP). At each iteration of DQN, a mini batch of states, actions, rewards, and next states are sampled from the replay memory [21] as observations to train the Q-network, which approximates the action-value function. The intuition behind experience replay is to achieve stability by breaking the temporal dependency among the observations used in training the deep neural network [10].

The Deep reinforcement learning augments the reinforcement learning framework and utilizes the powerful representation of deep neural networks. Recent works have demonstrated the remarkable successes of deep reinforcement learning in various domains including finance, medicine, healthcare, video games, robotics, and computer vision [15].

In particular, the application of deep reinforcement learning in computer vision can be divided into seven main categories according to their applications in computer vision, i.e. (i) landmark localization (ii) object detection; (iii) object tracking; (iv) registration on both 2D image and 3D image volumetric data (v) image segmentation; (vi) videos analysis; and (vii) other applications. The application of RL in computer vision is still not much explored [14].

There has been no work found in application of Reinforcement Learning in noise detection in any type of images. No framework exists as per our knowledge which can cater recognition of noises using Reinforcement Learning in either non-document or document images.

Here, we are proposing a solution which would be able to solve the problem of detecting the noise in document or non-document images using Deep Reinforcement Learning. Current solution can cater two types of noises which are commonly found in document images which are Gaussian noise and salt n pepper noise.

Gaussian noise: Gaussian Noise is a statistical noise having a probability density function equal to normal (Gaussian) distribution.

Salt n pepper noise: It is added to an image by randomly adding both bright (with 255-pixel value) and dark (with 0-pixel value) all over the image.

III. METHODOLOGY

We formalize the Noise Detection Markov Decision Process (NDMDP) framework which decomposes Noise Detection task into a sequential decision-making problem.

Assume that the training data set is D = {(x_1, y_1),(x_2, y_2), ...,(x_n, y_n)} where x_i is the i-th sample and y_i is the label of the i-th sample. We propose to train a classifier as an agent evolving in NDMDP using described state, rewards and other things.

- ***State S***: The state of environment is determined by the training sample. At the beginning of training, the agent receives the first sample X_1 as its initial state S_1. The state S_t of environment at each time step corresponds to the sample X_t. When the new episode begins, environment shuffles the order of samples in training data set.
- ***Action A***: The action of agent is associated with the label of training data set. The action a_t taken by agent is to predict a class label. For binary classification problem, A = {0, 1} where 0 represents the non-noisy image and 1 represents the Gaussian noised image. For 3 classes, A = {0,1,2} where in addition to earlier one, 2 represent Salt-n-pepper noise.
- ***Reward R***: To guide the agent to learn the optimal classification policy in training data with noisy image, the absolute reward value of sample in noisy images is higher than that in non-noisy image. ***i.e.***, when the agent correctly or incorrectly recognizes noisy image sample, the environment feedback agent a larger reward or punishment.
- ***Transition probability P***: Transition probability p ($s_{t+1}|s_t$, a_t) in NDMDP is deterministic. The agent moves from the current state s_t to the next state s_{t+1} according to the order of samples in the training data set.
- ***Discount factor γ***: $\gamma \in [0, 1]$ is to balance the immediate and future reward.
- ***Episode***: Episode in reinforcement learning is a transition trajectory from the initial state to the terminal state {s_1, a_1, r_1, s_2, a_2, r_2, ..., s_t, a_t, r_t}. An episode ends when all samples in training data set are classified or when the agent misclassifies the sample from noisy class.
- ***Policy πθ***: The policy πθ is a mapping function π: S → A where $\pi_\theta(s_t)$ denotes the action at performed by agent in state S_t. The policy πθ in NDMDP can be considered as a classifier with the parameter θ.

With the definitions and notations above, the Noise Detection problem is formally defined as to find an optimal classification policy π^*: S → A, which maximized the cumulative rewards in NDMDP.

A. Reward function for Noise Detection

To better recognize the Noisy samples, the algorithm should be more sensitive to the Noisy class. A large reward or punishment is returned to agent when it meets a Noisy sample. The reward function is defined as follows:

$$R(s_t, a_t, l_t) = \begin{cases} +1, & a_t = l_t \text{ and } s_t \in D_P \\ -1, & a_t \neq l_t \text{ and } s_t \in D_P \\ \lambda, & a_t = l_t \text{ and } s_t \in D_N \\ -\lambda, & a_t \neq l_t \text{ and } s_t \in D_N \end{cases}$$

Where, D_P represents the noisy image class set and D_N represents non-noisy class set.

B. DQN based Noise detection algorithm

In NDMDP, the classification policy π is a function which receives a sample and return the probabilities of all labels. π(a|s) = P(a_t = a|s_t = s). The classifier agent's goal is to correctly recognize the sample of training data as much as possible. As the classifier agent can get a positive reward when it correctly recognizes a sample, thus it can achieve its goal by maximizing the cumulative rewards.

In reinforcement learning, there is a function that calculates the quality of a state-action combination, called the Q function.

The classifier agent can maximize the cumulative rewards by solving the optimal Q function, and the greedy policy under the optimal Q function is the optimal classification policy π for NDMDP.

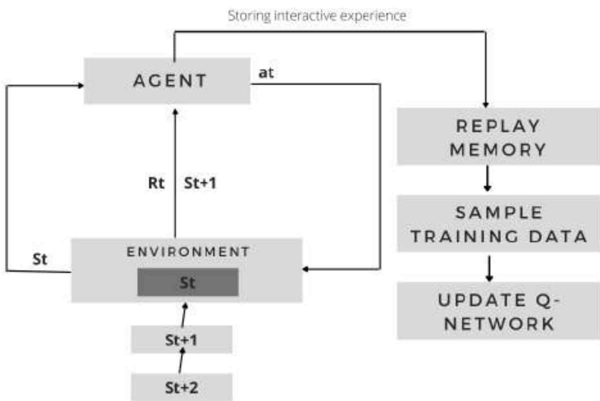

Fig. 1. Noise detection Markov Decision Process

IV. Training Details

We train with following dataset:

A. *Dataset:*

1. MNIST dataset: It is a collection of handwritten digit image from 0 to 9, having total 60000 images in training dataset, and 10000 images in test dataset.
2. Custom data: A collection of documents such as receipts were downloaded using open internet. The combined images were scaled to 64 x 64 and converted to grayscale. Gaussian Noises and Salt n pepper noises were introduced in these images in a controlled manner to for experimental purpose.

We construct the simulation environment according to the definition of NDMDP. The architecture of the Q network depends on the complexity and amount of training data set. The input of the Q network is consistent with the structure of training sample, and the number of outputs is equal to the number of sample categories. In fact, the Q network is a neural network classifier without the final SoftMax layer. The training process of Q network is described in ***Algorithm 1***.

In an episode, the agent uses the ε-greedy policy to pick the action, and then obtains the reward from the environment through the STEP function in ***Algorithm 2***. We trained deep Q-learning algorithm for 180000 iterations (updates of network parameters θ).

Input: Training data $D = \{(x_1, l_1), (x_2, l_2), ..., (x_T, l_T)\}$.
Episode number K.
Initialize experience replay memory M
Randomly initialize parameters θ
Initialize simulation environments ε
for *episode* $k = 1$ *to* K **do**
Shuffle the training data D
Initialize state $s_1 = x_1$
for $t = 1$ *to* T **do**
Choose an action based ϵ-greedy policy:
$a_t = \pi_\theta(s_t)$
$r_t, terminal_t = STEP(a_t, l_t)$
Set $s_{t+1} = x_{t+1}$
Store $(s_t, a_t, r_t, s_{t+1}, terminal_t)$ to M
Randomly sample $(s_j, a_j, r_j, s_{j+1}, terminal_j)$ from M
Set $y_j =$
$$\begin{cases} r_j, & terminal_j\text{=True} \\ r_j + \gamma \max_{a'} Q(s_{j+1}, a'; \theta), & terminal_j\text{=False} \end{cases}$$
Perform a gradient descent step on $L(\theta)$ w.r.t. θ:
$L(\theta) = (y_j - Q(s_j, a_j; \theta))^2$
if $terminal_t$=*True* **then**
break

Algorithm 1: Training DQN

D_p represents the noisy image class set i.e. if current state belongs to gaussian noise class or salt-n-pepper noise class, we give higher reward.

Function $STEP(a_t \in \mathcal{A}, l_t \in L)$
Initialize $terminal_t$=False
if $s_t \in D_P$ **then**
if $a_t = l_t$ **then**
Set r_t=1
else
Set r_t=-1
$terminal_t$=True
else
if $a_t = l_t$ **then**
Set $r_t = \lambda$
else
Set $r_t = -\lambda$
return $r_t, terminal_t$

Algorithm 2: Reward function (STEP)

B. *Network Architecture*

The Q-network architecture that is used for Noise classification has different architectures for DRL-1 and DRL-2. The DRL-2 is slightly deeper architecture. Its detailed parameters are given below.

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 64, 64, 32)	832
activation_1 (Activation)	(None, 64, 64, 32)	0
max_pooling2d_1 (MaxPooling2	(None, 32, 32, 32)	0
conv2d_2 (Conv2D)	(None, 32, 32, 32)	25632
activation_2 (Activation)	(None, 32, 32, 32)	0
max_pooling2d_2 (MaxPooling2	(None, 16, 16, 32)	0
flatten_1 (Flatten)	(None, 8192)	0
dense_1 (Dense)	(None, 256)	2097408
activation_3 (Activation)	(None, 256)	0
dense_2 (Dense)	(None, 2)	514

Architecture 1: DRL-1 model: Architecture of Q-network

DRL-2 is just addition of one more dense layer.

V. Results

It was started with just two categories, a Gaussian noise vs. no noise. The MNIST dataset has been taken for experiment purpose. The 60K images in training dataset have been introduced to Gaussian noise. Combined set of 60K non-noised images and 60K noised images has been used for training the agent. Here, a simple CNN architecture *(see image 2)* has been used as Q-Network.

The Deep Reinforcement Learning approach provided very high accuracy, around 99.8%, on a test dataset of 20K images having 10K images from non-noised and 10K images from Gaussian noise category.

In the final step, three classes were picked: ***no noise, Gaussian, Salt & Pepper.*** For this purpose, a total of 60K images of each category has been used. The overall workflow has been given in ***Image 1.***

Training the agent again using Deep Reinforcement learning on the same dataset gave test accuracy of 99.57%.

The same experiment was repeated with the text-image dataset keeping the same architecture for Q function. The results are summarized in the table below:

TABLE I.

Model Specification	*DRL-1**	*DRL-1*	*DRL-1*	*DR L-1*	*DRL-2*	*DRL-2***
Dataset	*MNIST*	*MNIST*	*Our data*	*Our data*	*Our data*	*Our data*
Classes Considered	*2*	*3*	*2*	*3*	*2*	*3*
No. of Training data	*60K+60K*	*60K+60K+60K*	*1771*	*265 6*	*1771*	*2656*
No. of testing data	*10K+10K*	*10K+10K+10K*	*442*	*665*	*442*	*665*
Epochs trained	*180000*	*180000*	*180000*	*180 000*	*18000 0*	*180000*
Accuracy	*99.8%*	*99.57%*	*98.63%*	*95.3 3%*	*98.12 %*	*97.89%*

Clearly, the experiment shows great results in identifying the noise in both document and non-document image dataset, through the results can improved for document images if more number of training samples are included

VI. Conclusion

This concept of using reinforcement learning for identifying correct image preprocessing for any document image started with a simple use case of identifying noise category for a particular image using Reinforcement learning.

With an appropriate reward function, it has been shown that the training agent using Reinforcement Learning can almost correctly classify the type of noise into Non-noise, Gaussian noise or Salt n pepper noise.

This solution can be extended to more preprocessing like correct angle detection of a document image, blur detection and related preprocessing techniques. And, able to find the correct preprocessing will allow to deal with them separately once identified.

References

[1] Suneetha, Akula and Srinivasa Reddy, E.. "Robust Gaussian Noise Detection and Removal in Color Images using Modified Fuzzy Set Filter" Journal of Intelligent Systems, vol. 30, no. 1, 2021, pp. 240-257. https://doi.org/10.1515/jisys-2019-0211

[2] J. Lu, M. Qin and Y. Tong, "Research on Reinforcement Learning algorithms in Computer Vision," 2022 International Symposium on Advances in Informatics, Electronics and Education (ISAIEE), Frankfurt, Germany, 2022, pp. 246-251, doi: 10.1109/ISAIEE57420.2022.00057.

[3] L. Mou, S. Saha, Y. Hua, F. Bovolo, L. Bruzzone and X. X. Zhu, "Deep Reinforcement Learning for Band Selection in Hyperspectral Image Classification," in IEEE Transactions on Geoscience and Remote Sensing, vol. 60, pp. 1-14, 2022, Art no. 5504414, doi: 10.1109/TGRS.2021.3067096.

[4] Venkatesh N. Murthy, Vivek Singh, Terrence Chen, R. Manmatha, Dorin Comaniciu; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 2240-2248

[5] Lin, E., Chen, Q. & Qi, X. Deep reinforcement learning for imbalanced classification. Appl Intell 50, 2488–2502 (2020). https://doi.org/10.1007/s10489-020-01637-z

[6] Wang S, Liu W, Wu J, Cao L, Meng Q, Kennedy PJ (2016) Training deep neural networks on imbalanced data sets, in Neural Networks (IJCNN). In: 2016 International Joint Conference on. IEEE, pp 4368–4374

[7] Huang C, Li Y, Change Loy C, Tang X (2016) Learning deep representation for imbalanced classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 5375–5384

[8] Roy, S. S., Ahmed, M., & Akhand, M. A. H. (2018). NOISY IMAGE CLASSIFICATION USING HYBRID DEEP LEARNING METHODS. Journal of Information and Communication Technology, 17(2), 233–269. https://doi.org/10.32890/jict2018.17.2.8253

[9] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. Nature, 518(7540):529–533, 2015.

[10] Jianqing Fan, Zhaoran Wang, Yuchen Xie, Zhuoran Yang Proceedings of the 2nd Conference on Learning for Dynamics and Control, PMLR 120:486-489, 2020.

[11] Vimala, B. B., Srinivasan, S., Mathivanan, S. K., Muthukumaran, V., Babu, J. C., Herencsar, N., & Vilcekova, L. (2023). Image noise removal in ultrasound breast images based on hybrid deep learning technique. Sensors, 23(3), 1167.

[12] Wang, F., Huang, H., & Liu, J. (2019). Variational-based mixed noise removal with CNN deep learning regularization. IEEE Transactions on Image Processing, 29, 1246-1258.

[13] Aghajarian, M., McInroy, J. E., & Muknahallipatna, S. (2020). Deep learning algorithm for Gaussian noise removal from images. Journal of Electronic Imaging, 29(4), 043005-043005.

[14] Arulkumaran, K., Deisenroth, M. P., Brundage, M., & Bharath, A. A. (2017). Deep reinforcement learning: A brief survey. IEEE Signal Processing Magazine, 34(6), 26-38.

[15] Ladosz, P., Weng, L., Kim, M., & Oh, H. (2022). Exploration in deep reinforcement learning: A survey. Information Fusion, 85, 1-22.

[16] Gronauer, S., & Diepold, K. (2022). Multi-agent deep reinforcement learning: a survey. Artificial Intelligence Review, 55(2), 895-943.

[17] Roderick, M., MacGlashan, J., & Tellex, S. (2017). Implementing the deep q-network. arXiv preprint arXiv:1711.07478.

[18] Oh, J., Hessel, M., Czarnecki, W. M., Xu, Z., van Hasselt, H. P., Singh, S., & Silver, D. (2020). Discovering reinforcement learning algorithms. Advances in Neural Information Processing Systems, 33, 1060-1070.

[19] AlMahamid, F., & Grolinger, K. (2021, September). Reinforcement learning algorithms: An overview and classification. In 2021 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE) (pp. 1-7). IEEE.

[20] Wawrzyński, P. (2009). Real-time reinforcement learning by sequential actor–critics and experience replay. Neural networks, 22(10), 1484-1497.

[21] Sharma, S., Srinivas, A., & Ravindran, B. (2017). Learning to repeat: Fine grained action repetition for deep reinforcement learning. arXiv preprint arXiv:1702.06054.

[22] Adam, S., Busoniu, L., & Babuska, R. (2011). Experience replay for real-time reinforcement learning control. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 42(2), 201-212.