Boosting Image Recognition with Non-differentiable Constraints

Xuan Li,[1] Yuchen Lu,[2] Peng Xu,[3] Jizong Peng,[4] Christian Desrosiers,[4] Xue Liu[1]
[1]McGill University
[2]Universite de Montreal
[3]Polytechnique Montreal
[4]ETS Montreal
xuan.li2@mcgill.ca, yuchen.lu@umontreal.ca, peng.xu@polymtl.ca, jizong.peng.1@etsmtl.net, christian.desrosiers@etsmtl.ca, xue.liu@mcgill.ca

Abstract

In this paper, we study the problem of image recognition with non-differentiable constraints. A lot of real-life recognition applications require a rich output structure with deterministic constraints that are discrete or modeled by a non-differentiable function. A prime example is recognizing digit sequences, which are restricted by such rules (e.g., *container code detection*, *social insurance number recognition*, etc.). We investigate the usefulness of adding non-differentiable constraints in learning for the task of digit sequence recognition. Toward this goal, we synthesize six different datasets from MNIST and Cropped SVHN, with three discrete rules inspired by real-life protocols. To deal with the non-differentiability of these rules, we propose a reinforcement learning approach based on the policy gradient method. We find that incorporating this rule-based reinforcement can effectively increase the accuracy for all datasets and provide a good inductive bias which improves the model even with limited data. On one of the datasets, MNIST_Rule2, models trained with rule-based reinforcement increase the accuracy by 4.7% for 2000 samples and 23.6% for 500 samples. We further test our model against synthesized adversarial examples, e.g., blocking out digits, and observe that adding our rule-based reinforcement increases the model robustness with a relatively smaller performance drop.

Introduction

There has been a rising interest in applying deep learning to the problem of Optical Character Recognition (OCR), with numerous datasets focusing on this task (Wang, Babenko, and Belongie 2011; Karatzas et al. 2015; Veit et al. 2016; Netzer et al. 2011). However, most of these datasets focus on the perception part of the problem with various kinds of realistic images. Nevertheless, domain-specific rules play an important part in various OCR tasks. For example, Social Insurance Number (SIN) uses Luhn algorithm (Luhn 1960) to generate qualified digits, suggesting that a successful SIN recognition model should take into account the underlying verification rule. We argue that, by equipping our model with the ability of reasoning over these specific rules, the algorithm can perform better in many real-life OCR applications.

To fill in the gap, we create six datasets by combining images from MNIST (LeCun et al. 1998) and Cropped SVHN (Netzer et al. 2011), respectively, using three different rules, inspired by real-life scenarios. A sample of our dataset can be found in Figure 5 and Figure 6.

To test whether existing models can perform well under these datasets, we first implement the Convolutonal Recurrent Neural Network (CRNN) (Shi, Bai, and Yao 2016). The model first uses convolutional layers to extract features, which are then sent to a Bidirectional LSTM. Different from the original work, since we have a fixed number of characters to recognize, we only produce a single prediction for each character and apply cross-entropy loss. An illustration of our architecture can be found in Figure 1. To incorporate non-differentiable rules into training, we add an additional term in the loss function measuring the expected rule reward which is optimized by a policy gradient algorithm. We further investigate the effect of different hyper-parameters and evaluate how the added rule can help the model learn with reduced data, which is a common problem in real-life scenarios. Our contributions can be summarized as follows:

- We propose a suite of rule-based OCR benchmark datasets with three different rules inspired by real-life applications;
- We propose a general method based on reinforcement learning (RL) to include non-differentiable constraints in the learning process of image recognition problems;
- We conduct an in-depth evaluation of our method on the proposed datasets. Our results show that the proposed rule-based reinforcement can help improve accuracy across different kinds of images and rules. We further demonstrate the effect of our method under different hyper-parameter settings, as well as its effectiveness in challenging learning scenarios involving limited data or missing/indistinguishable digits.

Related Works

Numerous datasets have been proposed for OCR and digit recognition tasks. MNIST is one of the classic datasets for such problems, largely popularized by the successful application of deep convolutional neural networks (CNNs) (LeCun et al. 1998). The Street View House Numbers (SVHN)

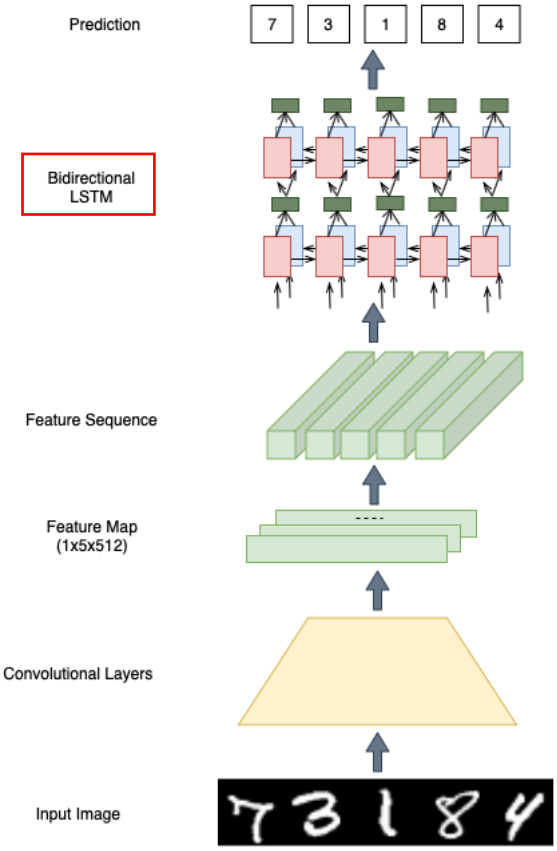

Figure 1: The Network Architecture

is another early dataset with more real-life images (Goodfellow et al. 2013). Following the spirit of large-scale text recognition in the wild, ICDAR robust reading competition was proposed in 2013 (Karatzas et al. 2013) and updated in the following years (Karatzas et al. 2015). It has become one of the standard benchmarks due to its high image quality. Following its success, other large-scale scene text datasets like SVT (Wang 2014) and COCO-Text (Veit et al. 2016) were created. So far, these datasets have mainly focused on the perception part of the problem. In contrast, the main objective of our dataset is to evaluate whether adding discrete rules while training the model can improve its recognition accuracy in testing.

There have been efforts on the application side to combine OCR with rule matching. Early works in container code identification added the verification rule during the inference phase to improve accuracy (Mei et al. 2016). Our method differs from this previous work since it includes rules in the training phase by directly optimizing for it.

Our task is also related to numeric reasoning, which has been an important topic in deep learning (Dehaene 2011). Although there has been a series of work on counting objects in images (Dijkstra et al. 2018) or in a sequential manner (Fang et al. 2018), the harder problem on arithmetic or number theory remain largely untouched. Our work can be thought of as a step toward the direction of marrying arithmetics with computer vision.

Our proposed method is related to structured text generation in natural language processing (NLP). The seminal work of Ranzato et al. (2015) proposes to use RL for optimizing sequence-level metrics like BLEU and apply it to neural machine translation. There have been similar techniques in other tasks like abstractive summarization (Dong et al. 2018) and goal-oriented dialogue (Shah, Hakkani-Tur, and Heck 2016). Although we do not have a sequential generation problem, our proposed rule-based reinforcement method is similar to these works.

Methodology

Suppose we have a dataset $\{x_i, y_i\}_{i=1}^N$. We formally define a rule as a function r mapping y in output space to a real number. Suppose our recognition system is a neural network which models the conditional distribution $p(y|x;\theta)$, where θ are the parameters. The objective to maximize for each example is defined as

$$(1-\alpha)\log p(y_i|x_i;\theta) + \alpha E_{\hat{y}\sim p(\cdot|x_i;\theta)}[r(\hat{y})] \quad (1)$$

The first term is the likelihood term, while the second term is the expected reward under the current model. The weight α is used to balance two objectives. By employing the score function trick of the reinforce algorithm (Williams 1992), the gradient of Eqn. (1) can be expressed as

$$\begin{aligned}
&(1-\alpha)\nabla_\theta \log p(y_i|x_i;\theta) + \alpha \int \nabla_\theta p(\hat{y}|x_i;\theta) r(\hat{y}) d\hat{y} \\
&= (1-\alpha)\nabla_\theta \log p(y_i|x_i;\theta) \\
&\quad + \alpha \int p(\hat{y}|x_i;\theta)\nabla_\theta \log p(\hat{y}|x_i;\theta) r(\hat{y}) d\hat{y} \\
&= (1-\alpha)\nabla_\theta \log p(y_i|x_i;\theta) \\
&\quad + \alpha E_{\hat{y}\sim p(\cdot|x_i;\theta)}[r(\hat{y})\nabla_\theta \log p(\hat{y}|x_i;\theta)] \quad (2)
\end{aligned}$$

In practice, we approximate the second expectation via sampling $\hat{y}_j \sim p(\cdot|x_i;\theta)$, and thus our final gradient is

$$(1-\alpha)\nabla_\theta \log p(y_i|x_i;\theta) + \alpha \sum_{j=1}^{M} r(\hat{y}_j)\nabla_\theta \log p(\hat{y}_j|x_i;\theta) \quad (3)$$

where M is the number of samples. The second term can also be viewed as policy gradients with a one-step MDP. Since our neural network outputs the logits for each character, we can perform categorical sampling.

Dataset

Due to the specific characteristics of our hypothesis, we construct six synthesized benchmark datasets using both MNIST (LeCun et al. 1998) and Cropped SVHN (Netzer et al. 2011) datasets. In our new datasets, each image has 5 digits which are horizontally concatenated by 5 single images. We design three discrete rules to create the datasets and evaluate our method.

Rule 1

This rule requires the fifth digit to be the remainder of the sum of the first 4 digits with respect to modulus 10 (Fig. 2).

To synthesize the dataset following this rule, we fist randomly sample numbers from 0 to 9 for the first 4 digits. Then, for each digit, we randomly sample an image from the original dataset of that number directory. The 5-th digit is obtained by applying Rule 1. Then, we sample an image from the corresponding directory. Each of the synthesized image is comprised of the five horizontally-concatenated images, where images are sampled from MNIST or Cropped SVHN, respectively. The final synthesized image and Rule 1 can be seen in Fig. 2).

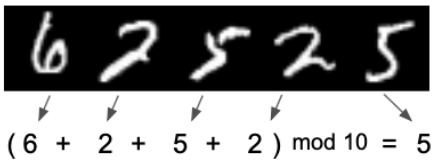

Figure 2: Illustration of Rule 1.

Rule 2

This second rule comes from the ISO 6346 protocol (ISO 1995) used for container code recognition. To verify if a 5-digit sequence satisfies this rule, each of the leading 4 digits is multiplied by 2^{pos} where $\text{pos} = 0, 1, 2, 3$. Then we sum up the resulting numbers, and find its remainder respect to modulus 11. If the remainder is 10, then the last digit should be 0, otherwise it should be the remainder. The generation of the dataset satisfying this rule is similar to Rule 1, once each digit is chosen, we randomly sample images from the corresponding image folder (see Fig. 3).

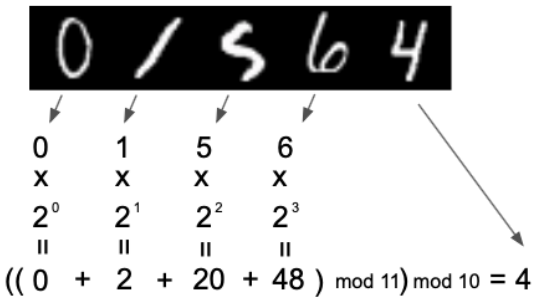

Figure 3: Illustration of Rule 2.

Rule 3

The third rule is from the Luhn algorithm (Luhn 1960) which is commonly used to verify Social Insurance Numbers or credit card numbers. To verify this rule for a sequence, the even-indexed numbers are first multiplied by 2, subtracting 9 if the product is greater than 9. The resulting numbers are then summed up along with the original odd-indexed numbers. According to the rule, the final result is required to be divisible by 10 (Figure 4). To generate the dataset satisfying this rule, a random number between 1000 and 9999 is randomly sampled, and then used as the first 4 digits in the string. To get the qualified 5-th digit, a Luhn generator is used (McLoughlin 2015). The generator takes the 4-digit string as input, and outputs a number that can be appended to the 5-th digit, making the whole 5-digit string Luhn string. The generation of the dataset from the string is similar to the previous rules. We randomly sample images from corresponding image folder.

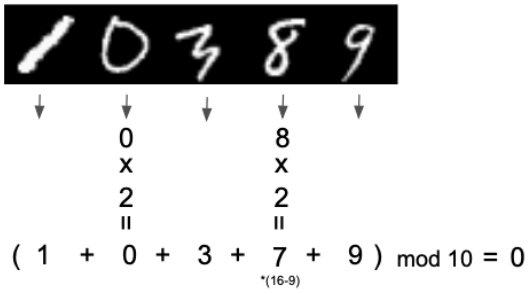

Figure 4: illustration of Rule 3.

As shown in Table 1, we end up with a total of 6 datasets. Each dataset consists of a total of 3000 images: 2000 randomly sampled images for training, 500 for validation, and 500 for testing. Examples of synthesized images are displayed in Fig. 5 and Fig. 6.

Table 1: Synthesized datasets

	MNIST	Cropped SVHN
Rule 1	MNIST_Rule 1	SVHN_Rule 1
Rule 2	MNIST_Rule2	SVHN_Rule2
Rule 3	MNIST_Rule3	SVHN_Rule3

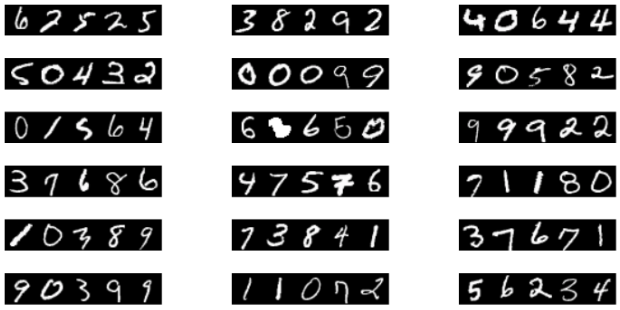

Figure 5: Samples of the synthesized images from MNIST. Top two rows are from Rule 1, middle two rows are from Rule 2, bottom two rows are from Rule 3

Figure 6: Samples of the synthesized images from cropped SVHN. Top two rows are from Rule 1, middle two rows are from Rule 2, bottom two rows are from Rule 3

We conduct our experiments on the six datasets with different combinations of three rules and images from two different sources. The datasets and models will be released upon acceptance.

Experiments

Setup

We adopt CRNN (Shi, Bai, and Yao 2016) as our baseline model to perform sequential recognition. Contrary to other traditional OCR evaluation metric focused on per character accuracy, we argue that the rule-based OCR should have a more strict assessment. We consider sequence-level accuracy:

$$\text{Accuracy}(\hat{y}_i, y_i) = \begin{cases} 1, & \text{if } \hat{y}_i^j = y_i^j \text{ for j = 0 to 4} \\ 0, & \text{otherwise} \end{cases} \quad (4)$$

where $\hat{y}_i^j$ is the j-th digit of i-th image, and y^i is the ground truth of i-th image. The prediction is correct if and only if all 5 digits are correctly recognized. During all experiments, we simply take the argmax of logits as the final predictions. We leave the investigation of more advanced decoding techniques for future work.

In each training, we resize input images to 28×112. We use Eqn. (3) with a batch size of 100 to compute the gradients and employ Adam (Kingma and Ba 2014) to train networks for a total of 200 epochs. The learning rate is set to 1×10^{-3} and divided by 10 every 60 epochs. We use Pytorch (Paszke et al. 2017) for all experiments. We choose $\alpha = 0.05$ and $M = 10000$ if without other specification.

The Effect of α

Parameter α controls the relative weight between the cross-entropy loss and the rule-based reinforcement term. We test the following values: 0, 0.001, 0.005, 0.01, 0.05, 0.1 and 0.5. When $\alpha = 0$, the model only uses the cross-entropy loss. Additionally, we test two strategies to update this parameter dynamically, called adaptive ascending (AA) and adaptive descending (AD). For adaptive ascending, we use

$$AA = \exp\left(1 - \frac{T}{i+1}\right) \quad (5)$$

where i is current epoch number, and T is the total number of epochs. On the other hand, for adaptive descending we set

$$AD = 1 - AA$$

The results can be found in Table 2. We first observe that adding rule-based reinforcement improves the performance across all three datasets consistently, showing the effectiveness of our proposed method. When increasing α, we find that the model's performance first increases but then decreases. This can be explained by the fact that our rule constraint does not dependent on the label, thus the model may only focus on producing well-defined sequence instead of recognizing actual digits. The optimal value of α can different in each dataset, so in practice it should be tuned to get the best performance. We also find that the adaptive ascending scheme (AA) outperforms the baseline model while the adaptive descending scheme (AD) decreases accuracy. We argue that it is more beneficial to let the cross-entropy loss be dominant in the beginning and only use rule-base reinforcement to improve the model in the later stage.

Table 2: Accuracy (%) on different α

α	MNIST_Rule 1	MNIST_Rule2	MNIST_Rule3
0	89.4	89.6	90.4
0.001	88.8	90.7	89.0
0.005	**91.0**	93.4	90.8
0.01	91.0	91.4	91.4
0.05	90.0	92.0	91.8
0.1	90.0	**94.3**	93.2
0.5	72.4	0	92.5
AA	90.0	92.6	**94.4**
AD	42.2	0	0

α	SVHN_Rule 1	SVHN_Rule2	SVHN_Rule3
0	22.1	31.4	28.2
0.001	23.3	31.2	28.0
0.005	22.6	31.5	34.8
0.01	25.2	32.2	34.2
0.05	24.6	35.0	35.8
0.1	**25.9**	**38.6**	**39.8**
0.5	24.3	0	23.2
AA	22.2	35.0	33.2
AD	0	0	0

Learning with Limited Data

This experiment explores the effect of training dataset size with values from $N = 500$ to 2000. Results of this experiments are provided in Table 3 along with relative accuracy gain. In most cases, adding rule-based reinforcement improves the performance regardless of the dataset size. It can be noticed that, in MNIST_Rule2 and MNIST_Rule3, the relative accuracy gain is larger when data is most limited (i.e., 500 samples), with 77.6% relative gain in MNIST_Rule2 and 11.5% in MNIST_Rule3. These results

suggest that, in some scenarios, enforcing the rule introduces a good inductive bias which helps the model learn with limited data. However, the same observation is made for the SVHN based dataset. We hypothesize that SVHN uses real-life images so it requires more data to have reasonable performance.

Missing Image with Blockout Data

In real-life applications, it is common to have missing digits, for instance due to occlusions or noise. In some cases, a model aware of the underlying rule may be able to fill in these missing digits.

We simulate this scenario by randomly blocking out a digit. We modify the test set images for three rules on MNIST, as illustrated in Fig. 7. The results of our model on this modified dataset are shown in Table 4. We see that, in most cases, adding the rule-based reinforcement term improves the accuracy. Moreover, we find that adding this term results in a relatively smaller loss in accuracy compared with original images. This suggests that incorporating rules can increase to some extent the model's robustness against missing numbers.

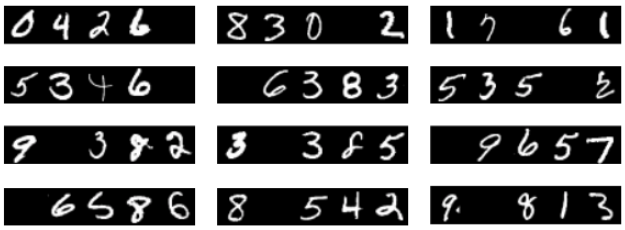

Figure 7: Examples of Blockout images.

Hard Digits

Another source of errors can come from poorly written (e.g., slanted or incomplete) digits. Once more, by restricting possible digit sequences with given rules, the model may be able to guess the correct digit. We design the following experiment to simulate this scenario.

We train an image classifier with ResNet50 (He et al. 2016) to classify MNIST digits. We early stop it so that the test accuracy is around 98.5%. We then locate the test set images that are incorrectly classified. We call these set of images *hard digits* (see Fig. 8). Afterwards, we modify the test datasets by randomly replacing a single digit with a hard digit one correspondong to the same value. We modify the 500 test images for each of the 3 rules on MNIST, as shown in Fig. 9. Results of this experiment can be found in Table 5. Once again, we see that incorporating the proposed rule-bases reinforcement in training can improve the model's robustness against hard digits, resulting in a less severe performance drop.

Discussion

To demonstrate the usefulness of the proposed method, we evaluated it on a small synthetic dataset. To further validate our method, we aim to collect data from real applications like container number detection. A large-scale real-life dataset could be more challenging, however the results can be more transferable to practice. While this work focuses on digit sequence recognition, our method can be easily transferred to other recognition tasks as long as well-defined rules exist for target patterns.

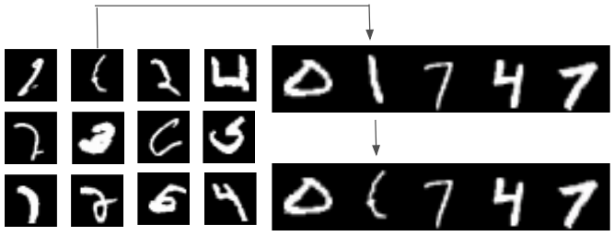

Figure 8: Illustration of the procedure to generate hard samples.

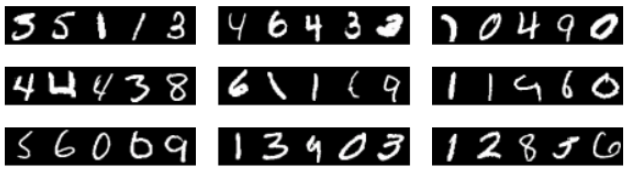

Figure 9: Examples of hard samples.

In future work, we would like to try more advanced algorithms like actor-critic (Konda and Tsitsiklis 2000) to reduce the gradient variance, or TRPO (Schulman et al. 2015) to increase the stability of each update. Value-based algorithms are another promising direction to improve results. For example, we can use the sum of logits as an approximation of Q value and perform Q-Learning (Mnih et al. 2013). Another family of algorithms which could be investigated is oracle regression. In the current framework, we treat rules as a black-box oracle. However, we could also employ a neural net to imitate the behavior of the oracle, and we use the gradient of this neural network to increase the rule matching score (Foster et al. 2018).

Conclusion

In this work, we studied the problem of digit sequence recognition with non-differentiable rules. We synthesized 6 datasets from MNIST and SVHN for digit recognition tasks, using rules rooting from real-life applications. To exploit these rules, we advocate employed a reinforcement learning approach based on the policy gradient algorithm. Using a CRNN as baseline model, our experiments show that the model's accuracy can be improved in a variety of scenarios, including limited data and missing/indistinguishable numbers digits, when the proposed ruled-based reinforcement term is used during training.

Table 3: Accuracy (%) for different dataset sizes N.

	MNIST_Rule 1			MNIST_Rule2			MNIST_Rule3		
N	Baseline	Proposed	Gain	Baseline	Proposed	Gain	Baseline	Proposed	Gain
500	29.6	33.9	14.5	30.4	54.0	77.6	64.4	71.8	11.5
1000	81.0	83.2	2.7	83.9	86.5	3.1	82.8	84.8	2.4
1500	84.8	87.8	3.5	88.8	90.8	2.3	89.6	90.2	0.7
2000	89.4	91.0	1.8	89.6	94.3	5.2	90.0	93.2	3.6

	SVHN_Rule 1			SVHN_Rule2			SVHN_Rule3		
N	Baseline	Proposed	Gain	Baseline	Proposed	Gain	Baseline	Proposed	Gain
500	0.1	0.1	0	0	0.2	-	0	0	-
1000	6.0	6.5	5.5	9.0	9.4	4.4	14.6	15.9	8.9
1500	17.4	20.2	16.1	23.6	31.2	30.2	15.2	20.2	32.9
2000	22.1	25.9	17.2	31.4	38.6	22.9	28.2	39.8	41.1

Table 4: Accuracy (%) on Blockout datasets

	Baseline			Proposed		
	Original	Blockout	Gain	Original	Blockout	Gain
MNIST_Rule 1	89.4	8.0	-91.1	91.0	9.0	**-90.1**
MNIST_Rule2	91.3	14.0	-84.7	91.6	14.0	-84.7
MNIST_Rule3	90.0	8.0	-91.1	92.2	11.0	**-88.1**

Table 5: Accuracy(%) on hard digits datasets.

	Baseline			Proposed		
	Original	Blockout	Gain	Original	Blockout	Gain
MNIST_Rule 1	89.4	33.0	-63.9	91.0	40.0	**-56.0**
MNIST_Rule2	91.3	45.0	-50.7	91.6	63.0	**-31.2**
MNIST_Rule3	90.0	52.0	-42.2	92.2	59.0	**-36.0**

References

[Dehaene 2011] Dehaene, S. 2011. *The number sense: How the mind creates mathematics*. OUP USA.

[Dijkstra et al. 2018] Dijkstra, K.; van de Loosdrecht, J.; Schomaker, L.; and Wiering, M. A. 2018. Centroidnet: A deep neural network for joint object localization and counting. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 585–601. Springer.

[Dong et al. 2018] Dong, Y.; Shen, Y.; Crawford, E.; van Hoof, H.; and Cheung, J. C. K. 2018. Banditsum: Extractive summarization as a contextual bandit. *arXiv preprint arXiv:1809.09672*.

[Fang et al. 2018] Fang, M.; Zhou, Z.; Chen, S.; and McClelland, J. 2018. Can a recurrent neural network learn to count things? In *CogSci*.

[Foster et al. 2018] Foster, D. J.; Agarwal, A.; Dudík, M.; Luo, H.; and Schapire, R. E. 2018. Practical contextual bandits with regression oracles. *arXiv preprint arXiv:1803.01088*.

[Goodfellow et al. 2013] Goodfellow, I. J.; Bulatov, Y.; Ibarz, J.; Arnoud, S.; and Shet, V. 2013. Multi-digit number recognition from street view imagery using deep convolutional neural networks. *arXiv preprint arXiv:1312.6082*.

[He et al. 2016] He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

[ISO 1995] ISO. 1995. Freight Containters – Coding, Identification and Marking (ISO 6346:1995(E)). Standard, International Organization for Standardization, Geneva, CH.

[Karatzas et al. 2013] Karatzas, D.; Shafait, F.; Uchida, S.; Iwamura, M.; i Bigorda, L. G.; Mestre, S. R.; Mas, J.; Mota, D. F.; Almazan, J. A.; and De Las Heras, L. P. 2013. Icdar 2013 robust reading competition. In *2013 12th International Conference on Document Analysis and Recognition*, 1484–1493. IEEE.

[Karatzas et al. 2015] Karatzas, D.; Gomez-Bigorda, L.; Nicolaou, A.; Ghosh, S.; Bagdanov, A.; Iwamura, M.;

Matas, J.; Neumann, L.; Chandrasekhar, V. R.; Lu, S.; et al. 2015. Icdar 2015 competition on robust reading. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, 1156–1160. IEEE.

[Kingma and Ba 2014] Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[Konda and Tsitsiklis 2000] Konda, V. R., and Tsitsiklis, J. N. 2000. Actor-critic algorithms. In *Advances in neural information processing systems*, 1008–1014.

[LeCun et al. 1998] LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P.; et al. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11):2278–2324.

[Luhn 1960] Luhn, H. P. 1960. Computer for verifying numbers. US Patent 2,950,048.

[McLoughlin 2015] McLoughlin, M. 2015. Project title. `https://github.com/mmcloughlin/luhn`.

[Mei et al. 2016] Mei, L.; Guo, J.; Liu, Q.; and Lu, P. 2016. A novel framework for container code-character recognition based on deep learning and template matching. In *2016 International Conference on Industrial Informatics-Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII)*, 78–82. IEEE.

[Mnih et al. 2013] Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

[Netzer et al. 2011] Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; and Ng, A. Y. 2011. Reading digits in natural images with unsupervised feature learning.

[Paszke et al. 2017] Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Automatic differentiation in pytorch.

[Ranzato et al. 2015] Ranzato, M.; Chopra, S.; Auli, M.; and Zaremba, W. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*.

[Schulman et al. 2015] Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; and Moritz, P. 2015. Trust region policy optimization. In *International conference on machine learning*, 1889–1897.

[Shah, Hakkani-Tur, and Heck 2016] Shah, P.; Hakkani-Tur, D.; and Heck, L. 2016. Interactive reinforcement learning for task-oriented dialogue management.

[Shi, Bai, and Yao 2016] Shi, B.; Bai, X.; and Yao, C. 2016. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE transactions on pattern analysis and machine intelligence* 39(11):2298–2304.

[Veit et al. 2016] Veit, A.; Matera, T.; Neumann, L.; Matas, J.; and Belongie, S. 2016. Coco-text: Dataset and benchmark for text detection and recognition in natural images. *arXiv preprint arXiv:1601.07140*.

[Wang, Babenko, and Belongie 2011] Wang, K.; Babenko, B.; and Belongie, S. 2011. End-to-end scene text recognition. In *2011 International Conference on Computer Vision*, 1457–1464. IEEE.

[Wang 2014] Wang, K. 2014. The street view text dataset.

[Williams 1992] Williams, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8(3-4):229–256.