Methods for Exploring the Search Space of Image Dataset Augmentation Policies

Daniel Harborne
Student
David Marshall
Supervisor
Kirill Sidorov
Moderator

School of Computer Science
Cardiff University
Cardiff, UK

1 Project Description

In the field of image classification, approaches using Deep Learning have become the standard for achieving state of the art performance [4, 14]. These algorithms "learn" how to perform their tasks rather than being explicitly programmed and in order to be effective, they often require large volumes of task-relevant examples in the form of labeled training data [13]. This poses two main problems. Firstly, performing the manual labelling of training images takes the commitment of a large number of human hours. Secondly, in many domains, rare edge cases exist which are hard to collect examples of. This ultimately means that even when all the collected images are labelled, these edge cases still prove to cause problems for the Deep Learning models performing the classification.

One approach to mitigate this issue is to use image augmentation [13, 18, 20] which produces "new" training images using image operations such as transformation, contrast change and mirroring. If this process is applied to already labelled images, the new images can be automatically labeled with the same label as the originals. Though it is widely recognized as being beneficial, there is a lack of studies that have explored developing best practises for image augmentation. Many state of the art developments to Deep Learning techniques have been achieved with augmentation as part of their training pipeline but the strategy used was developed manually through trial and error. As stated in [8], the Deep Learning training process would benefit if the selection of effective augmentation strategies could be automated.

This is the motivation behind the recent paper from Google [2], which explicitly outlines a possible search space of image augmentation techniques. In addition, the paper details an algorithm for automatically trialing samples of augmentations from this space on a sub-set of a dataset in order to find effective strategies to utilize in the final training on the full dataset. Although, the paper itself states that the search algorithm they propose (AutoAugment) may likely be improved upon, it already helped train models to beyond state of the art performance on many well established image classification challenge datasets such as CIFAR10 [10] and ImageNet [3]. Although improvements to Deep Learning model architecture and techniques has been explored regularly within the literature [5, 6, 7, 14, 15, 16, 17, 19, 22], the gains seen by this pursuit are shrinking [2] and so pursuit of successful augmentation strategies appears to be a promising path for improvements.

This project aims to contribute to this effort in two ways. Primarily, the aim is to provide a publicly available implementation of the AutoAugment algorithm proposed in [2]. Secondly, the project aims to explore possible adaptations of AutoAugment as well as alternative approaches in order to improve on its ability to recommend augmentation strategies. If the search for improvement is successful, the findings will be written up as a research paper and submitted to a suitable venue.

2 Project Aims and Objectives

This project has two aims (as outlined in 2.1 and 2.2). The first, implementing and open sourcing an implementation of the AutoAugment algorithm, will be the focus of the project and the main deliverable. The project will be considered a success if the requirements of the first objective are met. However, if time allows, the second objective will be pursued. The potential also exists to explore the second objective outside the time frame of the project in the future.

2.1 Implementing AutoAugment Algorithm

Core Requirements:

- Accurately reproduce the strategy documented in the paper [2].
- Develop the algorithm as a toolbox which can be applied to any dataset that adheres to some supplied minimal formatting and organizational criteria.
- Input and output signature of the toolbox allows for integration within TensorFlow[1] training pipelines.
- Publish the toolbox on GitHub[2] as open source with necessary documentation to allow the community to easily make use of it in future research and production applications.

2.2 Improving upon AutoAugment

Core Requirements:

- Explore possible strategies that could improve upon AutoAugment.
- For each strategy explored, produce results that conclude whether it provides an improvement to AutoAugment.
- If significant results are produced, write the findings up as a paper and publish to a suitable venue.
- Briefly outline and document any unexplored strategies that have potential for improving upon AutoAugment as future work.

[1] https://www.tensorflow.org/
[2] https://github.com/

3 Work Plan

In this section, the tasks to be performed during the project will be outlined with details about what they involve. To begin, in this section, some general justification for the structure of the work plan are outlined. In section 3.1, task specific considerations are provided . These decisions have been developed from consideration to minimizing the risks as outlined in section 3.2.

Due to the time constraints of the project and the nature of training deep learning models, it's important to structure the workload such that training of models can begin as soon as possible and be parallelized with other tasks.

In addition, the AutoAugment algorithm includes the need for training many models, as such some baseline timings for training a single model needs to be established so a calculation can be produced for how long running the algorithm could take once implemented.

Finally, as the implementation of AutoAugment is the priority, the work plan has been structured to allow for a large amount of slack time for the implementation in case it runs over estimate. If used, this comes at the cost of having to push back or abandon the objective of publishing an improvement to the AutoAugment algorithm (as seen in the Gantt chart in 3.2.3.

3.1 Individual Work Items

1. **Dataset Collection and Curation**
   In this stage, datasets that will be required for the project will be downloaded. In addition, smaller subsets of some datasets will be curated for use during development. Some datasets (ImageNet and SVHN [12]) won't be needed until the testing of an extension to AutoAugment or a novel approach is developed and thus these are given a high amount of slack time.

2. **Testing Computation Time with Provided Google Codebase**
   This stage focuses on providing a baseline time for training a single instance of the model used in the AutoAugment paper (Wide-ResNet-28-10 [21]) across a range of available resources. In addition, the time will be measured when training a model on a simpler dataset (MNIST [11]). This will allow for suitable contingencies to be put in place if the training time for CIFAR10 on the computational resources available looks to be too long that using CIFAR10 creates the risk of not completing the project in time.

3. **Implement AutoAugment**
   Notably, for the actual task of implementing AutoAugment, time has been allocated to allow for the knowledge acquisition needed to implement the AutoAugment algorithm. It is also worth noting that implementation is currently scheduled to first take place using the MNIST dataset which will allow for quicker model training during development. After the implementation is complete, the results can be generated for CIFAR10 (which can run in parallel to other tasks).

4. **Improve on AutoAugment**
   For this task, a number of possible augmentation policy selection strategies are outlined with time allocated to explore them and generate results. Systematic search has been targeted first because the algorithm to perform a systematic search should be simple to develop but will take a long time to run. As such, starting the results generation as soon as possible allows for other techniques to be explored in parallel. Two approaches that build directly from AutoAugment have been allocated time to explore as these shouldn't be too time consuming to produce once the implementation of the original AutoAugment is in place. Genetic algorithms is listed within this task but not currently allocated time. Whilst it does seem like a strategy with potential (as stated by [2]) it seems like the strategy from the list that will be the most time consuming to implement and as such, poses the highest risk to explore. It is still listed as an option as it could become viable if the time estimated to make progress on other tasks has been overestimated.

5. **Write Up**
   This task represents two sub tasks of writing up the work performed. Firstly, if results from novel strategies have shown to be significant, a paper can be written up within the project. Completing the paper within the project's time frame gives the benefit of being a strong outcome of the project and also allows for the paper to be targeted towards BMVC 2019 a very suitable venue for such a paper. It is worth noting that this paper could also be undertaken outside the scope of the project in the future. In which case, this time can be used to provide slack time for previous tasks. Finally, the project must be written up and submitted by 10th May and suitable time has been allocated to write drafts and final versions of the project report.

3.2 Risk Analysis

3.2.1 Risk Table

#	Risk	Likelihood	Impact	Strategies to Minimize Risk
1	Computation power is insufficient or the algorithm is too complex such that the aims of the project aren't achievable in allocated time frame. (The likelihood and impact are both High/Large only when considering all objectives, which include multiple attempts at innovating upon AutoAugment.)	High	Large	- Early assessment of computational time (Task 1 and 2). - Plan in place to first implement the algorithm using smaller , less complex dataset (Task 3.2). - Prioritizing a deliverable implementation first, innovation second (Task 3 vs Task 4). This greatly reduces the level of this type of risk interfering with the primary objective (from High to Low - As seen in Risk 3). - Prioritizing innovations in an order that prioritizes the ones easiest to explore as the project code base develops (Task 4).
2	Current knowledge required (TensorFlow, Reinforcement Learning) and/or the time it takes to achieve the true level of knowledge required is underestimated.	Low	Moderate	- Starting the background reading and implementation requiring unfamiliar concepts early in order to leave time to react to any underestimate in the time required. (Task 3.1 and 3.2).
3	Project fails to produce an output that can be submitted. For example, if work focuses on innovation only and the end results is purely a non-functional code base with no contribution.	Low	Large	- As with Risk 1, focus is to be placed on an implementation of AutoAugment using a less complex and sizeable dataset in order to minimize the time to a working solution.
4	New publications supersedes the work being pursued	Low	Modrate	- Literature review will be periodically carried out to identify new innovations. - The primary objective of the project is to focus on an implementation of AutoAugment which will still maintain value even with notable innovations being published. - The scenario of a paper being published that supersedes AutoAugment in every metric and that publishes it's code as a useable tool box would damage the usefulness of the project. This scenario is difficult to avoid and it's likelihood is somewhat reduced by the short timescale of the project.
5	The AutoAugment codebase is released publicly, removing the need to re-implement the algorithm from the paper.	Low	Large	- Google have published a codebase for their paper already which does not include the AutoAugment algorithm . This makes it less likely that they will release the full code for the algorithm itself. - Google have also published blog posts about the work which also seems like an opportune time to publish the code for the algorithm itself (which did not happen). - Like Risk 4, it is difficult to fully mitigate this risk but the short timescale of the project does help.
6	After producing results, a flaw in the approach is discovered which invalidate them. If this happens towards the end of the project then there may not be sufficient time to resolve the flaw and reproduce results.	Medium	Moderate	- Results will be generated as early as possible to allow for close inspection and review before looking to write them up. - Careful checking of the code, including test cases where appropriate can be used to ensure that the code is behaving as expected and the results generated are accurate.
7	The paper the project is based on (or an aspect of it) is shown to be flawed.	Low	Small	- The paper has been peer reviewed and as such has already been through a scrutiny process. - In the scenario that a flaw is discovered, it likely won't completely invalidate the produced AutoAugment implementation and it may need small modifications. Depending on the time remaining when this happened, the modifications could be carried out during the project or outlined as future work.

Figure 1: Risks table showing notable project risks that should be considered and mitigated against.

3.2.2 Risk Matrix

Scale of Impact	Liklihood of Occurrence: High	Medium	Low
Large	(1)		(3), (5)
Moderate		(6)	(2), (4)
Small			(7)

Figure 2: Risks matrix[9] for assessing overall risk of project.

3.2.3 Gantt Chart

ID	Task	Start Date	End Date	Slack Date
1	**Dataset Collection and Curation**	*28-Jan*	*10-Feb*	*25-Mar*
1 .1	*Download Datasets*	*28-Jan*	*10-Feb*	*25-Mar*
1 .1 .1	MNIST	*28-Jan*	03-Feb	
1 .1 .2	CIFAR-10	*28-Jan*	03-Feb	
1 .1 .3	ImageNet	*28-Jan*	10-Feb	25-Mar
1 .1 .4	SVHN	*28-Jan*	10-Feb	25-Mar
1 .2	*Curate and Reduce Datasets to Workable Size*	*28-Jan*	*10-Feb*	*25-Mar*
1 .2 .1	MNIST	28-Jan	03-Feb	
1 .2 .2	CIFAR-10	28-Jan	03-Feb	
1 .2 .3	"CIFAR-10 Reduced" (from paper)	28-Jan	10-Feb	25-Mar
2	**Testing Computation Time with Provided Google Code Base**	*28-Jan*	*10-Feb*	
2 .1	*Tests on Local Machine*	*28-Jan*	*03-Feb*	
2 .1 .1	Train model using reduced CIFAR10	28-Jan	03-Feb	
2 .1 .2	Tran model using full CIFAR10	28-Jan	03-Feb	
2 .2	*Tests on Cloud Computing*	*28-Jan*	*03-Feb*	
2 .2 .1	Train model using reduced CIFAR10	28-Jan	03-Feb	
2 .2 .2	Tran model using full CIFAR10	28-Jan	03-Feb	
2 .3	*Test on Local Machine Using MNIST*	*28-Jan*	*10-Feb*	
2 .3 .1	Adapt codebase to use different dataset	28-Jan	10-Feb	
2 .3 .2	Train model using reduced MNIST	28-Jan	10-Feb	
2 .3 .3	Train model using full MNIST	28-Jan	10-Feb	
3	**Implement AutoAugment**	*04-Feb*	*10-Mar*	*07-Apr*
3 .1	Reinforcement Learning Knowledge Gathering	04-Feb	10-Feb	
3 .2	*AutoAugment Using Reduced MNIST*	*04-Feb*	*24-Feb*	*07-Apr*
3 .2 .1	Implementation	04-Feb	17-Feb	07-Apr
3 .2 .2	Generate Results	18-Feb	24-Feb	07-Apr
3 .3	*AutoAugment Using Reduced CIFAR10*			
3 .3 .1	Generate Results	25-Feb	10-Mar	07-Apr
4	**Improve On AutoAugment**	*25-Feb*	*31-Mar*	
4 .1	*Systematic Search*	*25-Feb*	*10-Mar*	
4 .1 .1	Implementation	25-Feb	10-Mar	
4 .1 .2	Results Generation	11-Mar	31-Mar	
4 .2	A.A. on a further reduced dataset	11-Mar	24-Mar	
4 .3	Consecutive A.A.	25-Mar	31-Mar	
4 .4	Genetic Algorithm	-	-	
5	**Write Up**	*01-Apr*	*10-May*	
5 .1	*Paper Submission*	*01-Apr*	*15-Apr*	
5 .1 .1	Paper Draft	01-Apr	14-Apr	
5 .1 .2	Paper Final	15-Apr	15-Apr	
5 .2	*Project Submission*	*15-Apr*	*10-May*	
5 .2 .1	Project Draft	15-Apr	05-May	
5 .2 .2	Project Final	06-May	10-May	
	Milestones			

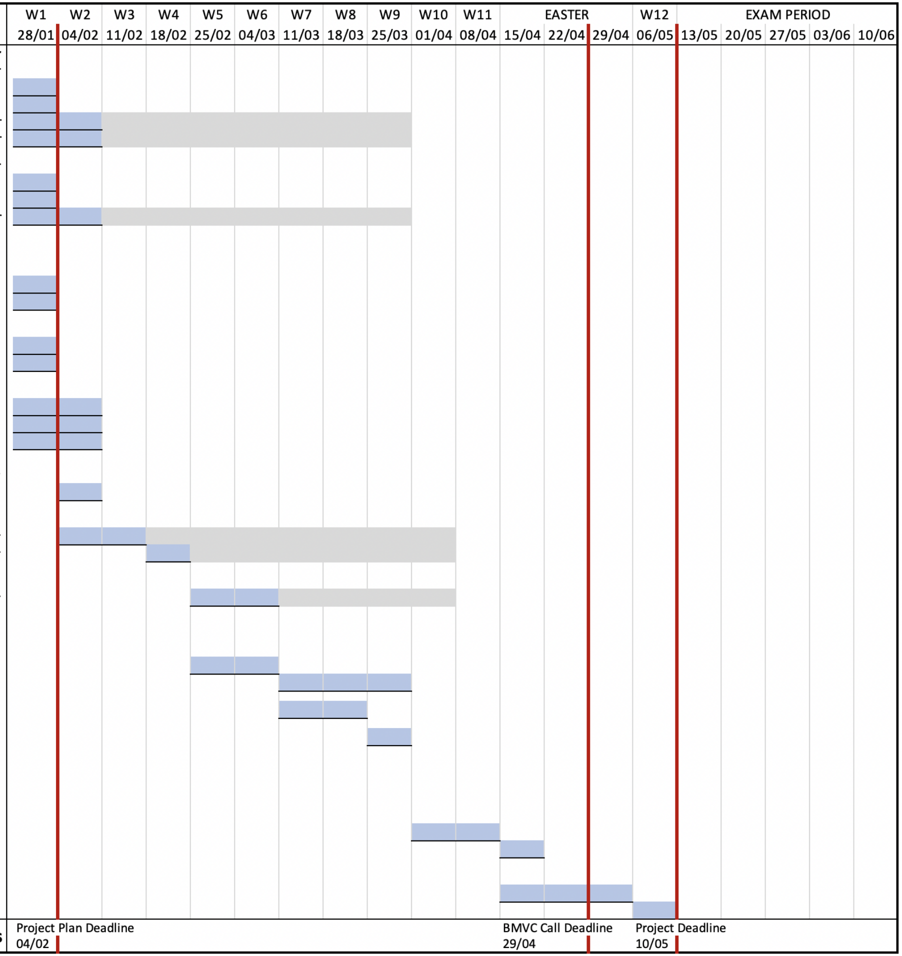

Figure 3: Gantt chart [1] showing tasks (left), the time allocated for them (blue bars), their slack (grey bars) and milestones (red lines).

References

[1] Wallace Clark. *The Gantt chart: A working tool of management*. Ronald Press, 1923.

[2] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. Ieee, 2009.

[4] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[5] Dongyoon Han, Jiwhan Kim, and Junmo Kim. Deep pyramidal residual networks. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 6307–6315. IEEE, 2017.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[7] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7132–7141, 2018.

[8] Tim Salimans Ilya Sutskever, John Schulman and Durk Kingma. Requests for research 2.0, 2018. URL https://blog.openai.com/requests-for-research-2.

[9] I ISO. Iso/iec 31010: 2009-risk management–risk assessment techniques'. *Int. Organ. Stand*, 2009.

[10] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[11] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[12] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 5, 2011.

[13] Luis Perez and Jason Wang. The effectiveness of data augmentation in image classification using deep learning. *arXiv preprint arXiv:1712.04621*, 2017.

[14] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. *arXiv preprint arXiv:1802.01548*, 2018.

[15] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[16] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[17] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, volume 4, page 12, 2017.

[18] Cristina Nader Vasconcelos and Bárbara Nader Vasconcelos. Increasing deep learning melanoma classification by classical and expert knowledge based image transforms. *CoRR, abs/1702.07025*, 1, 2017.

[19] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 5987–5995. IEEE, 2017.

[20] Yan Xu, Ran Jia, Lili Mou, Ge Li, Yunchuan Chen, Yangyang Lu, and Zhi Jin. Improved relation classification by deep recurrent neural networks with data augmentation. *arXiv preprint arXiv:1601.03651*, 2016.

[21] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *British Machine Vision Conference, 2016*, 2016.

[22] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8697–8710, 2018.