university of groningen / faculty of science and engineering

Deep learning with data augmentation

Bachelor's Project Thesis

Joppe Boekestijn, s2754215, j.w.boekestijn@student.rug.nl,
Supervisors: P.Pawara, M.Wiering

Abstract: Convolutional neural networks (CNNs) learn better when more training data is presented to the network. Data augmentation techniques artificially increase the amount of image data that can be passed to a CNN. This allows the CNNs to extract class-defining features more accurately, resulting in better classification accuracies. In this thesis several data techniques are presented and their performances are compared. Two deep learning architectures are used, GoogLeNet and ResNet, and they are both trained on a plant image dataset ('Tropic10'). The data augmentation techniques used in this paper are rotation, flipping, shifting, cutout, and mix-up. Their performances are compared, as well as the performance of some combination of techniques. This results in 11 different augmentation methods. Both deep learning architectures are configured with either pre-trained weights, trained on the 'ImageNet' dataset, or with randomly initialized weights. Both configurations benefit from data augmentation techniques, in some experiments leading to a 3% increase of classification accuracy. Especially flipping or shifting the images, or combining them, resulted in the best performance.

1 Introduction

Over the recent years machine learning architectures have contributed to large breakthroughs in the fields of image classification (Krizhevsky et al. [2012]). This can be contributed to the use and development of convolutional neural networks (CNNs) (Lecun et al. [1998]). These CNNs require vast amounts of learning data, and incorporate a variety of learning models. With the emergence of large labelled image datasets as shown by Deng et al. [2009], it has become possible to train increasingly deeper and more complex CNNs containing millions of parameters as used by He et al. [2015], Szegedy et al. [2014], Simonyan and Zisserman [2014], and Krizhevsky et al. [2012].

The performance of CNNs can be increased by increasing the number of training samples, which allows for better tweaking of its parameters. This reduces overfitting, increases generalizability, and leads to overall better performances. Although we have seen a rise of large datasets, the CNN will be limited to the contents of the dataset. This has led to a wide variety of data augmentation techniques that artificially increase the image training data for deep learning frameworks. The general gist of data augmentation techniques is altering or transforming data slightly, using varying image processing techniques, while maintaining most of the features in the data, and passing the data to the deep learning framework. The assumption is that the object of interest, which plays a factor in defining its class, will not be changed by image processing operations. The increased training data will then result in better classification scores, as found in Chatfield et al. [2014]. Applying data augmentation techniques has also shown improvements in other fields (Wang et al. [2015] and Ronneberger et al. [2015]).

Various data augmentation techniques have been proposed in the literature. Some standard techniques like rotating images, flipping images, adding blur, adjusting saturation and cropping. More intricate techniques have also been proposed, for example mix-up augmentation as found in Zhang et al. [2017], cutout augmentation from Devries and Taylor [2017], sample pairing (Inoue [2018]), and between-class learning as proposed by Tokozume et al. [2018]. All these techniques have shown an increase of classification accuracy.

Data augmentation techniques have shown to be vital in increasing the overall performance of deep learning frameworks (Howard [2013]). A variation of data augmentation techniques have been applied to a plant dataset as shown in Pawara et al. [2017]. In this research the performance of other augmenta-

tion techniques, and combinations of data augmentation techniques, will be compared, using a dataset of plant images. The performance of the data augmentation techniques will be shown by training two famous CNN architectures (Szegedy et al. [2014], He et al. [2015]).

Contributions: In this thesis the performance of different data augmentation techniques will be shown using two famous CNNs (ResNet and GoogLeNet). The ResNet 50-layer and the Inceptionv3 model will be trained with either pre-trained weights that already have been trained on the 'ImageNet' dataset, or with randomly initialized weights that will be trained from scratch. Single data augmentation techniques will be applied, as well as combinations of techniques, resulting in 11 different experimental setups. The CNNs are trained on three different data splits from a plant image dataset ('Tropic10'). The results show that the data augmentation techniques have a positive effect on the classification accuracy of the CNNs. For single data augmentation techniques, horizontal or vertical shifting show the largest increase in accuracy. For the combination of data augmentation techniques horizontally flipping and shifting the images, as well as vertically flipping and shifting the images, result in the best performance.

Paper outline: The paper is structured as follows. In Section 2 the data augmentation techniques are outlined, as well as details about the dataset. Furthermore, there will be a description of the experimental setup, including which framework is used, in combinations with which deep learning architectures have been used. A general introduction to both deep learning architectures used is given. In Section 3 the results are presented, which will be analysed in Section 4. Section 5 will cover a discussion about the results, in combination with ideas for further research.

2 Methods

This section gives an overview of the different data augmentation techniques that were used, in combination with two famous deep learning frameworks. Furthermore, details are given about the dataset that is used and how we will arrive at our performance measure.

2.1 Dataset

The dataset called 'Tropic10' consists of 2,555 images of leaves spread over 10 classes: ashoka, hibiscus, lime, west indian jasmine, lady palm, umbrella tree, ervatamia, mango, acacia, and sanchezia. Each image has a width and height of 250 pixels over three RGB-channels. There is a high similarity between the images in each class concerning the color variation. However, the pictures of the plants can be from each part of the plant, resulting in high intra-class dissimilarity. Another obstacle is that there is a similarity of shapes and colors between classes, making them more difficult to distinguish.

Table 2.1: Three different data splits of the 'Tropic10' image dataset

Dataset splits	Num. of images	
	Train set	Test set
Experiment 1	2044	511
Experiment 2	1788	767
Experiment 3	1792	763

2.2 Configuration

The images are passed to the different CNNs with an image width of 224 pixels, image height of 224 pixels, spread across RGB-channels. The dataset is split into three different data splits, for experiment 1 to 3, as shown in Table 2.1. The images are presented to the model in batches of 20 images. The model trains on all the train images per epoch, for a total of 50 epochs, resulting in 102,200 iterations. At the end of each epoch the model classifies the unseen test data. The best accuracy, which is the percentage of correct classifications, of all 50 epochs will be the resulting accuracy of an arbitrary experiment. In this way earlier epochs might be chosen to be the resulting model. This is called early stopping, while learning CNNs, sometimes there can be a fluctuation in classification accuracy. By taking the model at the best epochs we ensure that a random fluctuation does not deteriorate the results. For each experimental setup the mean of the accuracy scores on all three data splits is calculated with the accompanying standard deviation. This allows for a more general performance measure, since the classification accuracy of a CNN with the same

parameters is always different, due to the random order of images presented to the network.

The models have been implemented using Keras. The model is trained using the categorical cross entropy loss function and the Adam optimizer. In the literature multiple optimizers and loss functions are used and proposed, with multiple researches favoring the stochastic gradient descent (SGC) optimizer. In this paper the Adam optimizer showed the best all-round performance.

The starting learning rate of the model is 1e-3, but reduces during training. After 20 epochs the learning rate decreases to 1e-4, after 40 epochs to 1e-5. In the beginning a high learning rate leads to fast convergence, but in later epochs a lower learning rate will lead to better fine-tuning, increasing the overall performance.

In total 132 experiments have been performed. For both CNNs there are a total of 11 experiments with different data augmentation techniques and combinations, with two different initialized setups for the weights, which will be discussed in Section 2.4 and subsequent subsections.

2.3 Data augmentation techniques

In the following sections the data augmentation techniques used in this research are explained. Three standard data augmentation techniques are introduced: rotation, flip, and shift, as well as two relatively new data augmentation techniques called cutout, and mix-up. Examples of rotating, flipping, shifting, and cutout can be found in Figure 2.1. Examples of mix-up data augmentation can be found in Figure 2.2. The data augmentation is done online, which means that the augmented images are added in batches while the CNN is learning. As shown in Section 2.2 the images are presented to the deep learning architecture in batches of 20. The data augmentation techniques are randomly, on about half of the images in the batch applied.

2.3.1 Rotation

Images can be rotated from 0° to 90°. The images will be randomly rotated to the max degrees given. We found that 60° had the best results and will be used in the experiments. The extra pixels created by rotating the images are padded.

2.3.2 Flip

To counter a varied orientation of objects in images we can flip them horizontally and vertically. The experiments are conducted with either random horizontal flipping, or random vertical flipping. If either technique is chosen, the flipped images are added to the training dataset.

2.3.3 Shift

The pixels of the images can be shifted in horizontal and in vertical direction. They are shifted with a certain fraction of the respective width and height of the images, either horizontally (horizontal shift), or vertically (vertical shift). The best results were obtained by shifting the images 20% in both directions. This does not affect the resulting width and height of the images. The remaining pixels are padded with the color of the pixels which are adjacent to the newly formed pixels.

2.3.4 Cutout

Recently a new data augmentation technique was proposed called cutout, found in Zhang et al. [2017]. The idea of this technique is to randomly cut or mask out square areas in images and then feeds these images to the CNN. Examples of this can be found in Figure 2.1. The minimum/ maximum value for the erased area can be adjusted, as well as the probability that the cutout is performed. We found the best results with a probability of 50% and with an area cutout with a height and width that are proportional to 20% of the image. The probability that data augmentation techniques are applied at all is about 50%, meaning that cutout will only be applied to about 25% of the images in the batch.

Figure 2.1: 5 sample images with respectively from top to bottom: no data augmentation, flipping, shifting, rotation, and cutout.

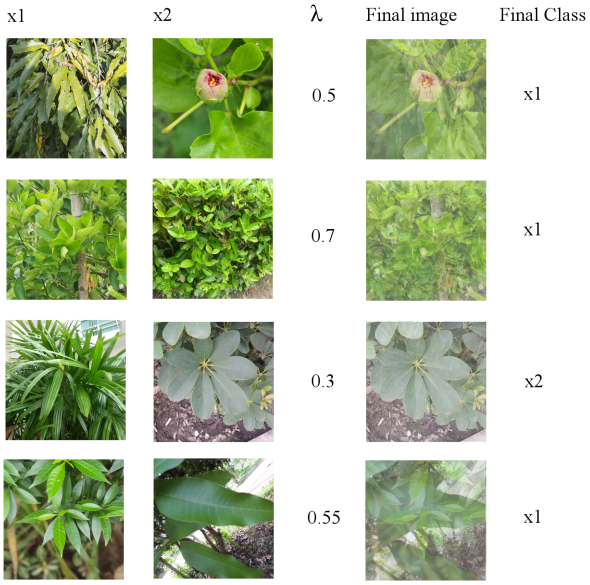

Figure 2.2: 4 examples of mix-up applied to images x1 and x2, with resulting final image and class

2.3.5 Mix-up

To create more images, information of multiple images can be combined. Using mix-up, two images are picked from the batch set and they are mixed up by combining the values of the pixels. If we have two images x1 and x2, the following equation is applied:

$$X = \lambda * X1 + (1 - \lambda) * X2$$

λ is drawn from a beta distribution: Be(α), where we found an α of 0.2 to have the best results. If λ is more than 0.5, the label of the resulting image becomes the label from x1, otherwise the label of x2 is given. In other words, the label of the image which is the most dominant in the resulting image, is given to that image. Four examples of this techniques, where mix-up is applied to images x1 and x2, are illustrated in Figure 2.2.

2.4 Deep learning architectures

The data augmentation techniques are applied to two famous CNNs. The ResNet framework (He et al. [2015]), and the Inception framework by Google (Szegedy et al. [2014]). These CNNs have each achieved low top-1 and top-5 error rates in the LSVRC-competition (Russakovsky et al. [2014]).

2.4.1 ResNet

The ResNet architecture, short for Residual Network, is a deep network consisting of up to 152 layers, created by Microsoft research, and proposed in

Table 2.2: ResNet 50-layer architecture

Layer name	Image size	50-layer
conv1	112×112	7×7, 62, stride 2
conv2_x	56×56	3×3 max pool, stride 2
		$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times3$
conv3_x	28×28	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times4$
conv4_x	14×14	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 512 \end{bmatrix} \times6$
conv5_x	7×7	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times3$
	1×1	average pool, 10-d, softmax

He et al. [2015]. If more and more layers are added to a CNN, there are a few problems that arise. One obstacle is the problem of vanishing gradients, where the error gradient becomes so diminishingly small that the weights are practically unchanged each iterations, resulting in stagnation of learning. The other problem is the degradation problem, if the depth of a network increases, and therefore the parameters, the accuracy on unseen data eventually starts to decrease rapidly.

ResNet was proposed to alleviate these problems. Instead of stacking layers they allow the framework to skip some of the layers, reducing complexity. Moreover, instead of layers learning on the output of their previous layer, they let the layers learn on a residual function. These alterations resulted in a very deep model, with relatively low complexity, and high accuracy. Two configurations of the ResNet model are used. The first configuration contains weights that have been pre-trained on the 'ImageNet' dataset. The other configuration randomly initializes the weights.

The ResNet architecture has been developed with quite some layer configurations. The configurations proposed by He et al. [2015] contain either 18-, 34-, 50-, 101-, or 152-layers. The number of parameters explode in amount when increasing the size of the ResNet architecture. More parameters allow for finer learning on the image data, since more parameters can incorporate more features in the data. However this does increase the complexity of the framework. In this paper the 50-layer ResNet architecture has been chosen, which has shown to be a balance in enough complexity to successfully train the CNN, but does not contain too many parameters. Therefore the CNN is not too slow, and does not require vastly more training and training data to fine-tune all the parameters. The total number of parameters is 23,608,202. The structure of the layers can be found in Table 2.2.

2.4.2 GoogLeNet

In 2014, Google introduced their own CNN architecture called GoogLeNet, or Inception, as shown in Szegedy et al. [2014]. To counter the vanishing gradient problem, and overfitting, they decided to introduce a wider model. Instead of stacking convolutional layers, multiple layers are run at the same time and afterwards concatenated in so called inception modules. This allows for a deep model, but with relatively low complexity. The total model contains 21,823,274 parameters.

The first version of GoogLeNet, called Inception v1, has incrementally been improved over the years. In this paper the Inception v3 model is used, which was proposed in the same paper as Inception v1 (Szegedy et al. [2014]). This model includes factorization of convolutional filters leading to faster computations. This model will also be used with either pre-trained weights, or with randomly initialized weights. The total number of parameters is 25,636,712. The model used is almost a replica of the visualization found in Figure 2.3, except that in the last part the softmax layer has an output of 10, instead of 1001. In the original paper the dataset used had 1000 classes, but the 'Tropic10' dataset has only 10.

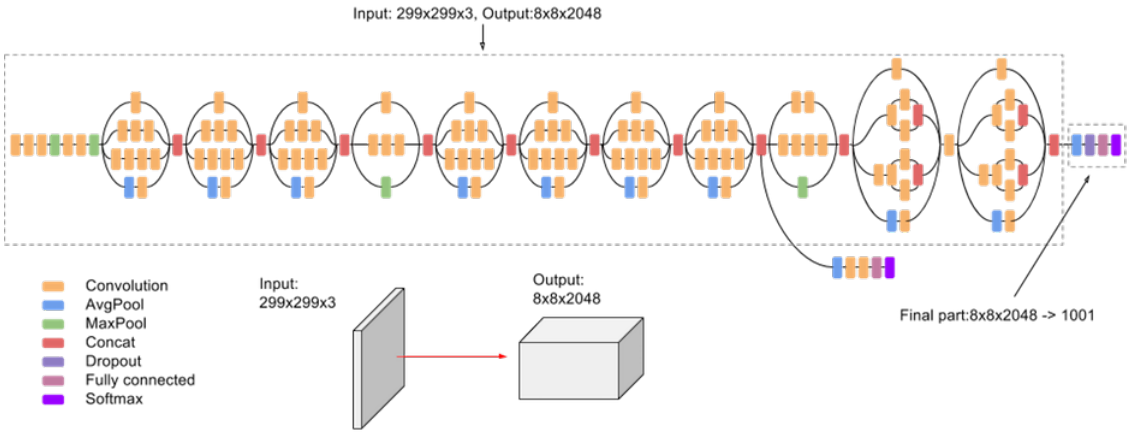

Figure 2.3: Graph showing the layers in the Inception v3 architecture. Source: ***https://cloud.google.com/tpu/docs/inception-v3-advanced***

3 Results

In the following sections the results of the ResNet and GoogLeNet deep learning architectures using different combinations of data augmentation techniques are presented. The accuracy scores for all the experiments can be found in Table 3.1. The bold-faced accuracies are the highest accuracies in their respective category, which is either applying a single data augmentation technique, or a combination of data augmentation techniques.

3.1 ResNet

The accuracies using the ResNet model can be found in Table 3.1. The accuracy of the ResNet model training from scratch is 93.14%, and using pre-trained weights is 97.50%. Using data augmentation techniques only in the case of vertical flipping for the model trained from scratch and applying mix-up resulted in a lower accuracy score (respectively 93.05% and 91.70%). The highest accuracy for single data augmentation techniques is found when applying horizontal shifting to the images, with an 2.62% increase from the model without data augmentation (95.76%). We observe that for single data augmentation techniques horizontally shifting images yields the highest accuracy score.

For the combination of data augmentation technique the highest score for the trained from scratch model is obtained by applying vertical shift, together with vertical shift, resulting in a 95.19% accuracy score. However, this is still a worse score than only applying horizontal shifting of images.

The model with pre-trained weights has the highest accuracy when applying a combination of horizontal flipping, and horizontal shifting of images (99.48%). This is almost a 2% increase from using the model without data augmentation (97.50%). This is not far from the best single data augmentation score, with 99.32%.

3.2 GoogLeNet

Accuracy scores using the GoogLeNet framework (Inception v3) can also be found in Table 3.1. The best combination of data augmentation techniques score using GoogLeNet architecture with randomly initialized weights is found when horizontally shifting the images, in combination with horizontal flipping of the images, with a score of 96.58%. This is an 3% increase from using the model without any data augmentation technique.

Using the pre-trained weights the highest accuracies for combinations of data augmentation techniques are found when vertically flipping, and vertically shifting the images (99.54%). This is however a minor increase from running the CNN without data augmentation (98.93%).

In both cases the best accuracies for single data augmentation techniques yield from vertically shift-

ing the images.

4 Conclusion

From the classification accuracies as shown in Section 3 we can derive the performance of the different data augmentation techniques on the plant image dataset. We observe that in almost all cases applying data augmentation techniques result in a considerable improvement of performance, with maximally an increase of 3%. For each deep learning architecture, with either randomly initialized weights, or pre-trained weights, there is always some data augmentation or combination of data augmentation techniques that results in better scores. However, the improvement might be only marginal, and in some cases a worse accuracy is obtained.

For this dataset flipping or shifting the images has a high performance throughout the experiments. Not only for applying single data augmentation techniques, but also when combining them. For both the ResNet framework as well as the GoogLeNet framework the highest accuracies are found applying either vertical or horizontal shifting, or a combination of techniques incorporating either one of those techniques. This is an interesting finding, since CNNs have been known to be translation invariant, meaning that shifting the object of interest should not have an influence on the performance. The increase in performance might be due to the padding of the pixels using the color of the pixels which are adjacent to the border. It might also be due to the fact that the features in the plant images that define its class usually have a specific orientation. Since the leaves appears together they are usually clustered in a specific location, but these locations can appear anywhere throughout the image. The pictures of the plants also relate to different parts of the whole plant. Shifting and flipping the images allows the algorithm to learn the clusters of leaves in a different location, and different parts of the plant. This might allow the algorithm to learn the features better for different orientations, which leads to better generalizability. This positively influences the classification accuracy.

Since data augmentation allows the CNN to learn features that it would not have been able to extract from the images itself, logically, combining well performing single data augmentation techniques will lead to even better accuracies. Although in all situations except the ResNet model with pre-trained weights the highest score of combinations of data augmentation techniques contains the overall best score, these are only marginal increases. Since flipping and shifting have a great performance throughout, combining high scoring data augmentation techniques result in even better performance, although that performance is only minimal.

If we compare all the single data augmentation scores with the combination of data augmentation scores we conclude that the combination of data augmentation techniques consistently have higher scores. However, when combining multiple techniques the already best scoring single data augmentation techniques were chosen, so therefore the combinations logically also have a high score. We can conclude that combining data augmentation techniques do not deteriorate the performance.

Interestingly, some data augmentation techniques had a performance which was worse than that of the standard CNN without data augmentation, although the difference is nowhere near significant. Although data augmentation increases the image data, if that image data does not benefit the actual learning, the added benefit of more images might be negated. Generally the data augmentation techniques do result in a better accuracy.

5 Discussion

The data augmentation techniques generally increase the classification accuracy of the deep learning frameworks. Although there is quite some discrepancy between the improvements of performance certain trends can certainly be extracted, as discussed in Section 4. The data augmentation techniques that have the highest accuracy seem to be the techniques that would logically increase performance for this specific dataset. As mentioned before the general orientation of the leaves in the images can be anywhere, but the leaves always appear clustered. Furthermore there is always some part of the plant shown, but this can be any part. Although flipping or shifting images seem to be the most rudimentary techniques, they lead to the biggest increase in performance on this plant dataset.

Despite being newer techniques cutout and mix-up data augmentation do not result in dominant

Table 3.1: Accuracy scores and standard deviation

Data augmentation techniques	ResNet		GoogLeNet	
	Scratch	Pre-trained	Scratch	Pre-trained
Original	93.14 ± 0.83	97.50 ± 0.13	93.58 ± 0.90	98.93 ± 0.46
Rotation	94.29 ± 0.44	98.82 ± 0.55	93.84 ± 0.94	99.09 ± 0.47
Horizontal flip	95.16 ± 0.97	98.43 ± 0.42	94.69 ± 0.99	99.24 ± 0.45
Vertical flip	93.05 ± 1.20	98.56 ± 0.67	94.89 ± 0.59	98.93 ± 0.24
Horizontal shift	**95.76 ± 0.43**	**99.32 ± 0.35**	94.95 ± 0.32	99.15 ± 0.19
Vertical Shift	94.45 ± 0.91	99.30 ± 0.27	**95.43 ± 1.12**	**99.50 ± 0.30**
Mix-up	91.70 ± 2.15	99.09 ± 0.11	95.19 ± 0.67	99.33 ± 0.17
Cutout	93.55 ± 0.41	97.95 ± 0.22	95.14 ± 1.48	98.92 ± 0.68
Horizontal flip + horizontal shift	94.93 ± 0.14	**99.48 ± 0.18**	**96.58 ± 0.35**	99.24 ± 0.20
Vertical flip + vertical shift	**95.19 ± 0.61**	99.17 ± 0.30	94.86 ± 1.28	**99.54 ± 0.19**
Vertical flip + cutout + horizontal shift	93.40 ± 1.10	99.43 ± 0.25	95.16 ± 0.38	99.35 ± 0.21

classification results. Cutout augmentation might have better accuracy on image data where the features to be extracted from the images are more evenly spread out. Cutting out parts of the image will then not result in losing valuable features from the images, and might lead to better classification accuracies.

The maximum increase in performance from running the CNN without data augmentation and with an arbitrary data augmentation technique is around 3%. Although this might not seem to be a substantial increase, these algorithms already have high accuracies, of around 95%. In that respect increasing the performance by even 1% can be critical. The CNNs seem to benefit a lot from data augmentation techniques on this dataset. This might be due to the dataset being relatively small. There are image datasets of tens of thousands of images, with the 'ImageNet' that was mentioned earlier in this paper containing up to 14 million images. Since the CNNs used in this research contain upwards of 30 million parameters, a lot of image data is needed to tweak all these parameters. Artificially creating more images data therefore leads to a higher performance increase than when already using a large dataset. This strengthens the notion that data augmentation techniques can lead to significant increases in performance on relatively small datasets.

Combining data augmentation yields consistently high scores, as shown in Section 3. However, in this paper, three combinations of data augmentation techniques were chosen on the basis of combining the best scoring single data augmentation techniques. Logically, combining high scoring augmentation techniques will also lead to higher classification accuracy. Finding out if combining lower scoring data augmentation technqiues will still have significant improvement, might be interesting to find out.

As mentioned before the data augmentation techniques that more logically lead to an increase for this specific dataset indeed have better performances, making it difficult to extract the general effect the data augmentation techniques will have on any image dataset. Instead of some data augmentation techniques being much more efficient and resulting in much better performance than others, it seems that the best data augmentation techniques depend on the features of the images in the dataset. Specifically the orientation of the objects that define its class, but also the saturation of colors, and the various shapes that it might contain. In this research only one dataset is used, containing plant images. For further research it might be interesting to try different kinds of image datasets to see whether or not there are dominant data augmentation techniques, or whether they are purely dependent on the features in the dataset.

References

Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *CoRR*, abs/1405.3531, 2014. URL `http://arxiv.org/abs/1405.3531`.

J. Deng, W. Dong, R. Socher, L. J. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009. doi: 10.1109/CVPR.2009.5206848.

Terrance Devries and Graham W Taylor. Improved Regularization of Convolutional Neural Networks with Cutout. *CoRR*, abs/1708.0, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. ISBN 978-1-4673-8851-1. doi: 10.1109/CVPR.2016.90.

Andrew G. Howard. Some improvements on deep convolutional neural network based image classification. *CoRR*, abs/1312.5402, 2013. URL `http://arxiv.org/abs/1312.5402`.

Hiroshi Inoue. Data Augmentation by Pairing Samples for Images Classification. *CoRR*, abs/1801.0, 2018.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In F Pereira, C J C Burges, L Bottou, and K Q Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

Y Lecun, L Bottou, Y Bengio, and P Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324, nov 1998. ISSN 0018-9219. doi: 10.1109/5.726791.

Pornntiwa Pawara, Emmanuel Okafor, Lambertus Schomaker, and Marco Wiering. Data augmentation for plant classification. In *Advanced Concepts for Intelligent Vision Systems (Acivs 2017)*, 9 2017.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015. URL `http://arxiv.org/abs/1505.04597`.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. *CoRR*, abs/1409.0575, 2014.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.

Yuji Tokozume, Yoshitaka Ushiku, and Tatsuya Harada. Between-class Learning for Image Classification. *CVPR*, 2018.

Limin Wang, Yuanjun Xiong, Zhe Wang, and Yu Qiao. Towards good practices for very deep two-stream convnets. *CoRR*, abs/1507.02159, 2015. URL `http://arxiv.org/abs/1507.02159`.

Hongyi Zhang, Moustapha Cissé, Yann N Dauphin, and David Lopez-Paz. Mixup: Beyond Empirical Risk Minimization. *CoRR*, abs/1710.0, 2017.