DATA AUGMENTATION INSTEAD OF EXPLICIT REGULARIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Modern deep artificial neural networks have achieved impressive results through models with very large capacity—compared to the number of training examples—that control overfitting with the help of different forms of regularization. Regularization can be implicit, as is the case of stochastic gradient descent or parameter sharing in convolutional layers, or explicit. Most common explicit regularization techniques, such as dropout and weight decay, reduce the effective capacity of the model and typically require the use of deeper and wider architectures to compensate for the reduced capacity. Although these techniques have been proven successful in terms of results, they seem to *waste* capacity. In contrast, data augmentation techniques reduce the generalization error by increasing the number of training examples and without reducing the effective capacity. In this paper we systematically analyze the effect of data augmentation on some popular architectures and conclude that data augmentation alone—without any other explicit regularization techniques—can achieve the same performance or higher as regularized models, especially when training with fewer examples.

1 INTRODUCTION

Regularization plays a central role in machine learning. Loosely defined, regularization is any modification applied to a learning algorithm that helps prevent overfitting and improve generalization. Whereas in simple machine learning algorithms the sources of regularization can be easily identified as explicit terms in the objective function, in modern deep neural networks the sources of regularization are multiple and some of them are not explicit, but implicit.

Although the terms explicit and implicit regularization have been used recently in the literature (Neyshabur et al., 2014; Zhang et al., 2016), their distinction is rather subjective. We propose the following definitions:

- **Explicit regularization** techniques are those *specifically* and solely designed to constrain the effective capacity of a given model and therefore reduce overfitting. Besides, explicit regularizers are not a structural or essential part of the network architecture, the data or the learning algorithm and can typically be added or removed easily.
- **Implicit regularization** is the reduction of overfitting or generalization error provided by characteristics of the network architecture, the training data or the learning algorithm, which are not specifically designed to constrain the effective capacity of the given model.

Examples of explicit regularizers are weight decay (Hanson & Pratt, 1989), which penalizes large weights; dropout (Srivastava et al., 2014), which randomly removes a fraction of the neural connections during training; or stochastic depth (Huang et al., 2016), which drops whole layers instead. Implicit regularization effects are provided by the popular stochastic gradient descent (SGD) algorithm, which tends to converge to solutions with small norm (Zhang et al., 2016); convolutional layers, which impose parameter sharing based on prior knowledge about the data; batch normalization (Ioffe & Szegedy, 2015), whose main goal is reducing the the internal covariate shift, but also implicitly regularizes the model due to the noise in the batch estimates for mean and variance.

Driven by the efficient use and development of GPUs, much research efforts have been devoted to finding ways of training deeper and wider networks of larger capacity (Simonyan & Zisserman,

2014; He et al., 2016; Zagoruyko & Komodakis, 2016), Ironically, their effective capacity is eventually reduced in practice by the use of weight decay and dropout, among other explicit regularizers. It is known, for instance, that the gain in generalization provided by dropout comes at the cost of using larger models and training for longer (Goodfellow et al., 2016). Hence, it seems that with such an approach deep networks are wasting capacity (Dauphin & Bengio, 2013). As a matter of fact, unlike traditional machine learning models, deep neural networks seem not to need explicit regularizers to generalize well, as recently suggested by Zhang et al. (2016).

One popular technique that improves generalization is data augmentation. Importantly, it differs from explicit regularizers mainly in that it does not reduce the effective capacity of the model. Data augmentation is a very old practice in machine learning (Simard et al., 1992) and it has been identified as a critical component of many models (Ciresan et al., 2010; Krizhevsky et al., 2012; LeCun et al., 2015). However, although some authors have reported the impact of data augmentation on the performance of their models and, in some cases, a comparison of different amount of augmentation (Graham, 2014) the literature lacks, to our knowledge, a systematic analysis of the impact of data augmentation on different models compared to the most popular regularization techniques.

1.1 Our contributions

In this paper, we systematically analyze the role of data augmentation in deep neural networks for object recognition, compare it to some popular explicit regularization techniques, discuss its relationship with model capacity and test its potential to enhance learning from less training data.

1.1.1 Data augmentation and explicit regularization

Zhang et al. (2016) recently raised the thought-provoking idea that *explicit regularization may improve generalization performance, but is neither necessary nor by itself sufficient for controlling generalization error.* They came to this conclusion from the observation that turning off the explicit regularizers of a model does not prevent the model from generalizing, although the performance becomes degraded. This contrasts with traditional machine learning involving convex optimization, where regularization is necessary to avoid overfitting.

However, in Zhang et al. (2016) data augmentation is considered an explicit form of regularization similar to weight decay and dropout. We instead believe that data augmentation deserves a different classification because of some fundamental properties: Notably, data augmentation does not reduce the effective capacity of the model. Explicit regularizers are often used to counteract overfitting, but as a side effect the architecture needs to be larger and the training process longer (Krizhevsky et al., 2012; Goodfellow et al., 2016). In contrast, data augmentation increases the number of training examples—although not in an independently distributed way—and the robustness against input variability. This has the welcome side-effect of implicitly regularizing the model and improving generalization.

Here, we build upon some of the ideas and procedures from Zhang et al. (2016) and perform some experiments to assess the role of data augmentation in deep neural neural networks and in particular in contrast to explicit regularizers (weight decay and dropout). In our experiments, we consider two levels of augmentation, light and heavier, as well as no augmentation at all. Then, we test them on two popular successful network architectures: the relatively shallow all convolutional network net (Springenberg et al., 2014) and the deeper wide residual network (Zagoruyko & Komodakis, 2016), trained on CIFAR-10 and CIFAR-100 (Krizhevsky & Hinton, 2009). Our central conclusion can be summarized as:

In a deep convolutional neural network trained with sufficient level of data augmentation, optimized by SGD, explicit regularizers (weight decay and dropout) might not provide any additional generalization improvement.

1.1.2 Data augmentation and training with fewer examples

Augmented data might be regarded as artificial and very similar to the source examples, therefore with limited contribution for making a network learn more useful representations. However, it has proven to be very useful in extreme cases such as one-shot learning, where only one or few training examples are available (Vinyals et al., 2016).

In order to provide a better insight of the usefulness of data augmentation, we train the networks with 80%, 50 % and 10 % of the available data and test the effect of data augmentation, again in contrast to explicit regularizers. The summary of our findings in this regard can be summarized as:

When a deep neural network is trained with a subset of the training data, the heavier the data augmentation, the smaller the gap with respect to the baseline model, especially if no explicit regularization is used. Thus, data augmentation seems to serve as true data to a great extent.

1.1.3 DATA AUGMENTATION AND ADAPTABILITY

One of the disadvantages of explicit regularization is that the parameters highly depend on the architecture. Therefore, if the architecture changes, one has to fine tune the regularization parameters to achieve comparable results. In order to analyze how data augmentation adapts to different architectures, we test several augmentation schemes on shallower and deeper versions of the network, with and without explicit regularization. Our finding is the following:

Data augmentation easily adapts to different depths without tuning its parameters if no explicit regularization is used. We observe that a shallower network achieves slightly worse results and a deeper architecture achieves better results.

1.2 RELATED WORK

Regularization is a central research topic in machine learning as it is a key component for ensuring good generalization (Girosi et al., 1995; Müller, 2012). In the case of deep learning, where networks tend to have several orders of magnitude more parameters than training examples, statistical learning theory (Vapnik & Chervonenkis, 1971) indicates that regularization becomes even more crucial. Accordingly, a myriad of tools and techniques have been proposed as regularizers: early stopping (Plaut et al., 1986), weight decay (Hanson & Pratt, 1989) and other L^p penalties, dropout (Srivastava et al., 2014) or stochastic depth (Huang et al., 2016), to name a few examples. Besides, other successful techniques have been studied for their regularization effect, despite not being explicitly intended as such. That is the case of unsupervised pre-training (Erhan et al., 2010), multi-task learning (Caruana, 1998), convolutional layers (LeCun et al., 1990), batch normalization (Ioffe & Szegedy, 2015) or adversarial training (Szegedy et al., 2013).

Data augmentation is another almost ubiquitous technique in deep learning, especially in computer vision, which can be regarded as a regularizer because it improves regularization. It was already used in the late 80's and early 90's for handwritten digit recognition (Simard et al., 1992) and it has been identified as a very important element of many modern successful models, like AlexNet (Krizhevsky et al., 2012), All-CNN (Springenberg et al., 2014) or ResNet (He et al., 2016), for instance. In some cases, data augmentation has been applied heavily with successful results (Wu et al., 2015). In domains other than computer vision, data augmentation has also been proven effective, for example in speech recognition (Jaitly & Hinton, 2013), music source separation (Uhlich et al., 2017) or text categorization (Lu et al., 2006).

Bengio et al. (2011) focused on the importance of data augmentation for recognizing handwritten digits (MNIST) through greedy layer-wise unsupervised pre-training (Bengio et al., 2007). The main conclusion of that work was that deeper architectures benefit more from data augmentation than shallow networks. Zhang et al. (2016) included data augmentation in their analysis of the role of regularization in the generalization of deep networks, although it was considered an explicit regularizer similar to weight decay and dropout. A few works have reported the performance of their models when trained with different types of data augmentation levels, as is the case of Graham (2014). Recently, the deep learning community seems to have become more aware of the importance of data augmentation and new techniques, such as cutout (Devries & Taylor, 2017) or augmentation in the feature space (DeVries & Taylor, 2017), have been proposed. Very interestingly, models that automatically learn useful data transformations have also been published very recently (Hauberg et al., 2016; Lemley et al., 2017; Ratner et al., 2017).

2 EXPERIMENTS AND RESULTS

This section describes the experimental setup for systematically analyzing the role of data augmentation in modern deep neural networks and presents the most relevant and interesting results.

2.1 SETUP

All the experiments are performed on the neural networks API Keras (Chollet et al., 2015) on top of TensorFlow (Abadi et al., 2015) and on a single GPU NVIDIA GeForce GTX 1080 Ti.

2.1.1 NETWORK ACRCHITECTURES

We perform our experiments on two popular architectures that have achieved successful results in object recognition tasks: the all convolutional network, All-CNN (Springenberg et al., 2014) and the wide residual network, WRN (Zagoruyko & Komodakis, 2016). We choose these networks not only because of their effectiveness, but also because they have simple architectures, which is convenient for drawing clearer conclusions. All-CNN has a relatively small number of layers and parameters, whereas WRN is rather deep and has many more parameters.

All convolutional net. All-CNN consists of only convolutional layers with ReLU activations (Glorot et al., 2011), it is relatively shallow (12 layers) and has about 1.3 M parameters. The architecture can be described as follows:

$$2\times96\text{C3(1)–}96\text{C3(2)–}2\times192\text{C3(1)–}192\text{C3(2)–}192\text{C3(1)–}192\text{C1(1)}$$
$$\text{–}N.Cl.\text{C1(1)–Gl.Avg.–Softmax}$$

where $K\text{C}D(S)$ is a $D \times D$ convolutional layer with K channels and stride S, followed by batch normalization and a ReLU non-linearity. *N.Cl.* is the number of classes and Gl.Avg. refers to global average pooling. The network is identical to the All-CNN-C architecture in the original paper, except for the introduction of the batch normalization layers. We set the same training parameters as in the original paper in the cases they are reported. Specifically, in all experiments the All-CNN networks are trained using stochastic gradient descent with batch size of 128, during 350 epochs, with fixed momentum 0.9 and learning rate of 0.01 multiplied by 0.1 at epochs 200, 250 and 300. The kernel parameters are initialized according to the Xavier uniform initialization (Glorot & Bengio, 2010).

Wide Residual Network. WRN is a modification of ResNet (He et al., 2016) that achieves better performance with fewer layers, but more units per layer. Although in the original paper several combinations of depth and width are tested, here we choose for our experiments the WRN-28-10 version (28 layers and about 36.5 M parameters), which is reported to achieve the best results on CIFAR. It has the following architecture:

$$16\text{C3(1)–}4\times160\text{R–}4\times320\text{R–}4\times640\text{R–BN–ReLU–Avg.(8)–FC–Softmax}$$

where KR is a residual block with residual function BN–ReLU–KC3(1)–BN–ReLU–KC3(1). BN is batch normalization, Avg.(8) is spatial average pooling of size 8 and FC is a fully connected layer. The stride of the first convolution within the residual blocks is 1 except in the first block of the series of 4, where it is 2 to subsample the feature maps. As before, we try to replicate the training parameters of the original paper: we use SGD with batch size of 128, during 200 epochs, with fixed Nesterov momentum 0.9 and learning rate of 0.1 multiplied by 0.2 at epochs 60, 120 and 160. The kernel parameters are initialized according to the He normal initialization (He et al., 2015).

2.1.2 DATA

We perform the experiments on the two highly benchmarked data sets CIFAR-10 and CIFAR-100 (Krizhevsky & Hinton, 2009), which are labeled according to 10 and 100 object classes respectively. Both data sets consist of 60,000 32 x 32 color images split into 50,000 for training and 10,000 for testing. In all our experiments, the input images are fed into the network with pixel values normalized to the range $[0, 1]$ and with floating precision of 32 bits. So as to analyze the role of data augmentation, we test the network architectures presented above with two different augmentation schemes as well as with no data augmentation at all:

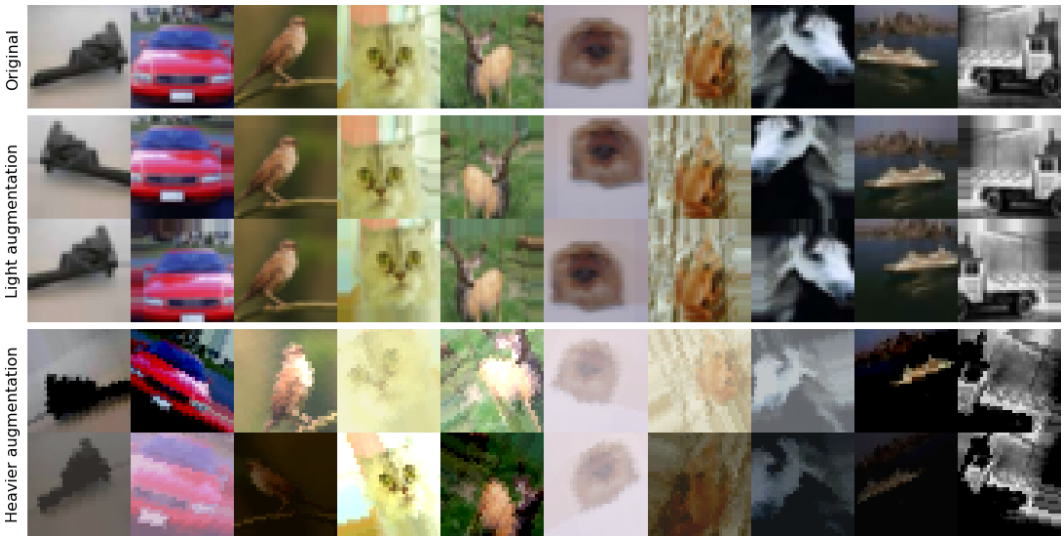


Figure 1: Random images from CIFAR10 transformed according to the augmentation schemes used in our experiments, choosing extreme values from the augmentation parameters. Note that these images are very unlikely to be used during training.

Table 1: Description and range of possible values of the parameters used for the heavier augmentation. $B(p)$ denotes a Bernouilli distribution and $\mathcal{U}(a, b)$ a uniform distribution.

Parameter	Description	Range
f_h	Horizontal flip	$1 - 2B(0.5)$
t_x	Horizontal translation	$\mathcal{U}(-0.1, 0.1)$
t_y	Vertical translation	$\mathcal{U}(-0.1, 0.1)$
z_x	Horizontal scale	$\mathcal{U}(0.85, 1.15)$
z_y	Vertical scale	$\mathcal{U}(0.85, 1.15)$
θ	Rotation angle	$\mathcal{U}(-\frac{\pi}{180}22.5, \frac{\pi}{180}22.5)$
ϕ	Shear angle	$\mathcal{U}(-0.15, 0.15)$
γ	Contrast	$\mathcal{U}(0.5, 1.5)$
δ	Brightness	$\mathcal{U}(-0.25, 0.25)$

***Light* augmentation.** This scheme is adopted from previous works, for example (Goodfellow et al., 2013; Springenberg et al., 2014), and performs only horizontal flips and horizontal and vertical translations of 10% of the image size.

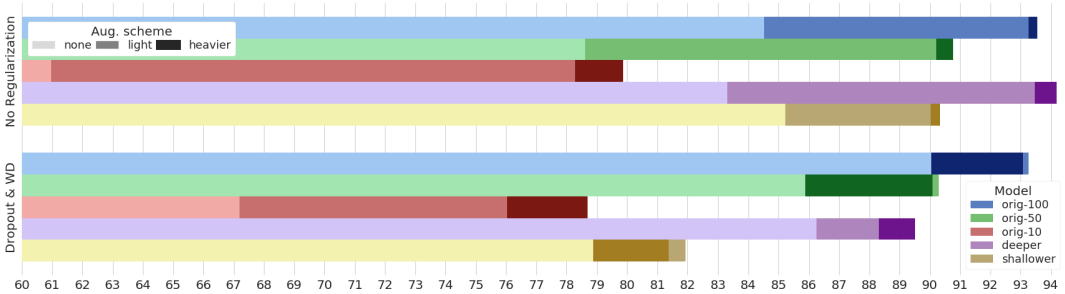
***Heavier* augmentation.** This scheme performs a larger range of affine transformations, as well as contrast and brightness adjustment:

- Affine transformations:

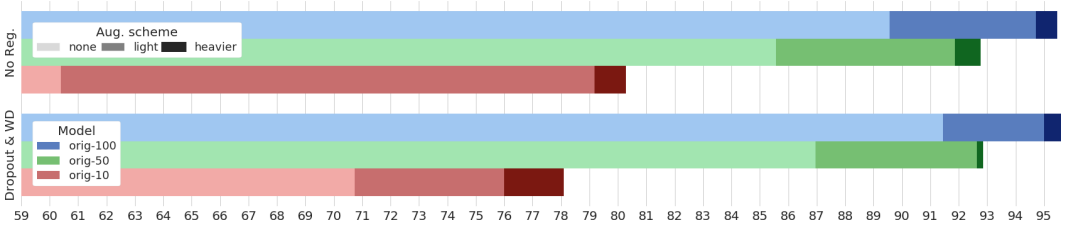
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} f_h z_x \cos(\theta) & -z_y \sin(\theta + \phi) & t_x \\ z_x \sin(\theta) & z_y \cos(\theta + \phi) & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Contrast adjustment: $x' = \gamma(x - \overline{x}) + \overline{x}$
- Brightness adjustment: $x' = x + \delta$

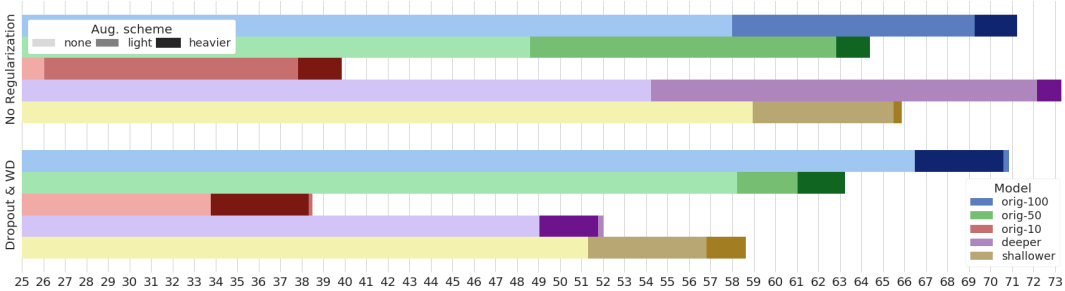
The description and range of values of the parameters are specified in Table 1 and some examples of transformed images with extreme values of the parameters are provided in Figure 1.



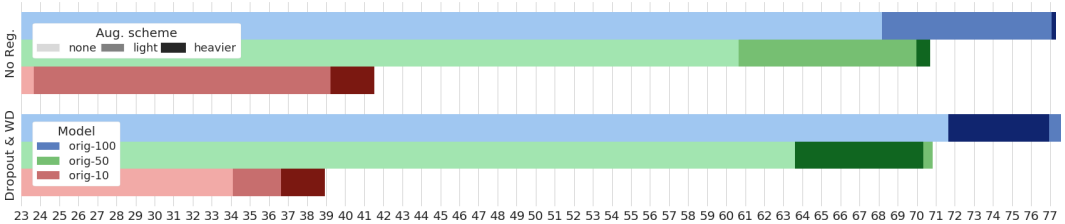
(a) All-CNN - CIFAR10



(b) WRN - CIFAR10



(c) All-CNN - CIFAR100



(d) WRN - CIFAR100

Figure 2: Test accuracy of the networks All-CNN and WRN on CIFAR-10 and CIFAR-100, trained without any explicit regularization (upper set of bars) and with both dropout and weight decay (lower bars), as in the original papers. The different bars represent different models (original, deeper and shallower) and different percentage of training images (100, 50 and 10 %). The different shades within each bar show the result of training with each data augmentation scheme (none, light and heavier). In most cases, the models without regularization achieve the same performance as the explicitly regularized models, or even significantly higher accuracy.

2.2 A SUBSTITUTE FOR EXPLICIT REGULARIZATION

In order to analyze the role of data augmentation and test the hypothesis that it might serve as a substitute for explicit regularization techniques, we first try to replicate the results of All-CNN and WRN provided in the original papers, achieved with both weight decay and dropout. Then, we train the models without weight decay and finally without neither weight decay nor dropout. We test all these different models with the three data augmentation schemes presented above: light, heavier and no augmentation. Additionally, in many cases we test the effect of removing the batch normalization.

As reported by previous works (Krizhevsky et al., 2012; Simonyan & Zisserman, 2014), when the models are trained with data augmentation, at test time slightly better results are obtained by augmenting the test set as well. Therefore, the test accuracy reported here comes from the averaged softmax posteriors over 10 random *light* augmentations.

The main results of the different experiments are shown in Figure 2 (blue bars) and the full report of all experiments can be found in Table 2 of the appendix. As expected, both explicit regularization —weight decay and dropout—and data augmentation are successful in reducing the generalization error. However, some relevant observations can be made. Most notably, it seems that data augmentation alone is able to regularize the model as much as in combination with weight decay and dropout and in some cases it clearly achieves better performance, as in the case of All-CNN. Another observation is that without explicit regularization, heavier augmentation always provides better results than light augmentation, whereas with regularization this effect is counterintuitively not consistent.

2.3 Fewer available training examples

We extend the analysis of the data augmentation role by training the same networks with fewer training examples. Similarly, we also analyze the combination of data augmentation and explicit regularization in this case. All models are trained with the same random subset of data and tested in the same test set as the previous experiments in order to enable fairer comparisons. Figure 2 (green and red bars) shows the main results with 50 % and 10 % of the available data and the full report, as well as additional results from training with 80 % and 1 % of the data, is given in Table 3 of the appendix.

As expected, the performance decays with the number of available training examples. However, as the level of data augmentation increases, the difference with respect to the baseline performance (by training with all examples) significantly decreases. This indicates that data augmentation serves, to a great extent, as true data. Therefore, this confirms the effectiveness of this technique when not many training examples are available.

Furthermore, in all the experiments with a reduced set of the available data, the observations presented above become even more clear. It seems that if explicit regularization is removed, data augmentation alone better resists the lack of data. This can probably be explained by the fact that explicit regularization reduces the effective capacity, making it more difficult to take advantage of the augmented data. As a matter of fact, we observe that some of the models trained with explicit regularization underfit the data.

2.4 Shallower and deeper architectures

Finally, we perform the same kind of experiments on shallower and deeper versions of All-CNN, in order to analyze how data augmentation is handled by architectures of different depth, compared to explicit regularization. We test a shallower network with 9 layers instead of 12 and 374 K parameters instead of 1.3 M:

$$2\times96\text{C3}(1)\text{–}96\text{C3}(2)\text{–}192\text{C3}(1)\text{–}192\text{C1}(1)\text{–}\textit{N.Cl.}\text{C1}(1)\text{–Gl.Avg.–Softmax}$$

and a deeper network with 15 layers and 2.4 M parameters:

$$2\times96\text{C3}(1)\text{–}96\text{C3}(2)\text{–}2\times192\text{C3}(1)\text{–}192\text{C3}(2)\text{–}2\times192\text{C3}(1)\text{–}192\text{C3}(2)\text{–}192\text{C3}(1)\text{–}192\text{C1}(1)$$
$$\text{–}\textit{N.Cl.}\text{C1}(1)\text{–Gl.Avg.–Softmax}$$

The results in Figure 2 (purple and brown bars) together with the detailed report of results in Table 4 of the appendix show that if the explicit regularization is removed and data augmentation applied, the shallower network achieves slightly worse results and the deeper network slightly better results than the original network. This behavior can be explained by the reduced or increased depth and number of parameters. However, with the explicit regularization active, the results dramatically decrease in both cases. The most probable explanation is that the regularization parameters are not adjusted to the architecture, whereas in the original models the parameters where finely tuned by the authors to obtain state of the art results. This highlights another important advantage of data augmentation: the adjustment of its parameters depends mostly on the training data, rather than on the particular architecture, which offers much more flexibility compared to using explicit regularization.

3 DISCUSSION AND CONCLUSION

In this work, we have presented a systematic analysis of the role of data augmentation in deep neural networks for object recognition, focusing on the comparison with popular techniques of explicit regularization. We have built upon the work by Zhang et al. (2016), in which the authors concluded that explicit regularization is not necessary, although it improves generalization performance. Here, we have shown that it is not only unnecessary, but also that the generalization gain provided by explicit regularization can be achieved by data augmentation alone.

The importance of these results lies in the fact that explicit regularization is the standard tool to enable the generalization of most machine learning methods. However, according to Zhang et al. (2016), explicit regularization plays a different role in deep learning, not explained by statistical learning theory (Vapnik & Chervonenkis, 1971). We argue instead that the theory still holds in deep learning, but one has to properly consider the very important role of implicit regularization. Explicit regularization is no longer necessary because its contribution is already provided by the many elements that implicitly regularize deep neural networks: stochastic gradient descent, convolutional layers or data augmentation, among others.

Whereas explicit regularizers, such as weight decay and dropout, succeed in mitigating overfitting by blindly reducing the effective capacity of a model, implicit regularization operates more effectively at capturing important characteristics of the data (Neyshabur et al., 2014). For instance, convolutional layers successfully reduce the capacity of a model by imposing a parameter sharing strategy that incorporates some essential prior domain knowledge, as well as data augmentation by transforming the training examples in a meaningful and plausible way.

In this regard it is worth highlighting some of the advantages of data augmentation, which is the main concern of this work. Not only does data set augmentation not reduce the effective capacity of the model, but it increases the number of training examples, which, according to statistical learning theories, reduces the generalization error. Furthermore, if the transformations are such that they reflect the variations of the real world, it increases the robustness of the model and it can be regarded as a data-dependent prior, similarly to unsupervised pre-training (Erhan et al., 2010). Besides, unlike explicit regularization techniques, data augmentation does not increase the computational complexity because it can be performed in parallel to the gradient updates on the CPU, making it a computationally free operation. Finally, in Section 2.4 we have shown how data augmentation adapts smoothly to architectures of different depth, whereas explicitly regularized models need manual adjustment of the regularization parameters.

Deep neural networks can especially benefit from data augmentation because they do not rely on precomputed features and because the large number of parameters allows them to shatter the augmented training set. Actually, if data augmentation is included for training, we might have to reconsider whether deep learning operates in an *overparameterization* regime, since the model capacity should take into account the amount of training data, which is exponentially increased by augmentation.

One may think that despite the advantages of data augmentation, it is a limited approach because it depends on some prior expert knowledge and it cannot be applied to all domains. However, we argue instead that expert knowledge should not be disregarded but exploited. Besides, some recent works show that it is possible to learn the data augmentation strategies (Lemley et al., 2017; Ratner et al., 2017) and future research will probably yield even better results.

Finally, it is important to note that, due to computational limitations, we have performed a systematic analysis only on CIFAR-10 and CIFAR-100, which consist of very small images. These data sets do not allow performing more agressive data augmentation since the low resolution images can easily show distortions that hinder the recognition of the object. However, some previous works (Graham, 2014; Springenberg et al., 2014) have shown impressive results by performing heavier data augmentation on higher resolution versions on CIFAR-10. We plan to extend this analysis to higher resolution data sets such as ImageNet and one could expect even more benefits from data augmentation compared to explicit regularization techniques.

ACKNOWLEDGMENTS

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 641805.

REFERENCES

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `http://tensorflow.org/`. Software available from tensorflow.org.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *Advances in neural information processing systems*, pp. 153–160, 2007.

Yoshua Bengio, Arnaud Bergeron, Nicolas Boulanger-Lewandowski, Thomas Breuel, Youssouf Chherawala, Moustapha Cisse, Dumitru Erhan, Jeremy Eustache, Xavier Glorot, Xavier Muller, et al. Deep learners benefit more from out-of-distribution examples. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 164–172, 2011.

Rich Caruana. Multitask learning. In *Learning to learn*, pp. 95–133. Springer, 1998.

François Chollet et al. Keras. `https://github.com/fchollet/keras`, 2015.

Dan Claudiu Ciresan, Ueli Meier, Luca Maria Gambardella, and Jrgen Schmidhuber. Deep big simple neural nets excel on handwritten digit recognition. *CoRR*, abs/1003.0358, 2010. URL `http://dblp.uni-trier.de/db/journals/corr/corr1003.html#abs-1003-0358`.

Yann N Dauphin and Yoshua Bengio. Big neural networks waste capacity. *arXiv preprint arXiv:1301.3583*, 2013.

Terrance Devries and Graham W. Taylor. Improved regularization of convolutional neural networks with cutout. *CoRR*, abs/1708.04552, 2017. URL `http://arxiv.org/abs/1708.04552`.

Terrance DeVries and Graham W Taylor. Dataset augmentation in feature space. *arXiv preprint arXiv:1702.05538*, 2017.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(Feb):625–660, 2010.

Federico Girosi, Michael Jones, and Tomaso Poggio. Regularization theory and neural networks architectures. *Neural computation*, 7(2):219–269, 1995.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *JMLR W&CP: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2010)*, volume 9, pp. 249–256, may 2010.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 315–323, 2011.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

Ian J. Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron C. Courville, and Yoshua Bengio. Maxout networks. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, pp. 1319–1327, 2013. URL `http://jmlr.org/proceedings/papers/v28/goodfellow13.html`.

Benjamin Graham. Fractional max-pooling. *arXiv preprint arXiv:1412.6071*, 2014.

Stephen José Hanson and Lorien Y Pratt. Comparing biases for minimal network construction with back-propagation. In *Advances in neural information processing systems*, pp. 177–185, 1989.

Søren Hauberg, Oren Freifeld, Anders Boesen Lindbo Larsen, John Fisher, and Lars Hansen. Dreaming more data: Class-dependent distributions over diffeomorphisms for learned data augmentation. In *Artificial Intelligence and Statistics*, pp. 342–350, 2016.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

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*, pp. 770–778, 2016.

Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European Conference on Computer Vision*, pp. 646–661. Springer, 2016.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pp. 448–456, 2015.

Navdeep Jaitly and Geoffrey E Hinton. Vocal tract length perturbation (vtlp) improves speech recognition. In *Proc. ICML Workshop on Deep Learning for Audio, Speech and Language*, pp. 625–660, 2013.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Yann LeCun, Bernhard E Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne E Hubbard, and Lawrence D Jackel. Handwritten digit recognition with a back-propagation network. In *Advances in neural information processing systems*, pp. 396–404, 1990.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

Joseph Lemley, Shabab Bazrafkan, and Peter Corcoran. Smart augmentation-learning an optimal data augmentation strategy. *IEEE Access*, 2017.

Xinghua Lu, Bin Zheng, Atulya Velivelli, and ChengXiang Zhai. Enhancing text categorization with semantic-enriched representation and training data augmentation. *Journal of the American Medical Informatics Association*, 13(5):526–535, 2006.

Klaus-Robert Müller. Regularization techniques to improve generalization. In *Neural Networks: Tricks of the Trade*, pp. 49–51. Springer, 2012.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In search of the real inductive bias: On the role of implicit regularization in deep learning. *CoRR*, abs/1412.6614, 2014. URL `http://arxiv.org/abs/1412.6614`.

D.C. Plaut, S.J. Nowlan, and G.E Hinton. Experiments on learning by back propagation. 1986.

Alexander J Ratner, Henry R Ehrenberg, Zeshan Hussain, Jared Dunnmon, and Christopher Ré. Learning to compose domain-specific transformations for data augmentation. *arXiv preprint arXiv:1709.01643*, 2017.

Patrice Simard, Bernard Victorri, Yann LeCun, and John Denker. Tangent prop-a formalism for specifying selected invariances in an adaptive network. In *Advances in neural information processing systems*, pp. 895–903, 1992.

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

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958, 2014.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Stefan Uhlich, Marcello Porcu, Franck Giron, Michael Enenkl, Thomas Kemp, Naoya Takahashi, and Yuki Mitsufuji. Improving music source separation based on deep neural networks through data augmentation and network blending. *Submitted to ICASSP*, 2017.

V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probab. and its Applications*, 16(2):264–280, 1971.

Oriol Vinyals, Charles Blundell, Tim Lillicrap, koray kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 29*, pp. 3630–3638. Curran Associates, Inc., 2016. URL `http://papers.nips.cc/paper/6385-matching-networks-for-one-shot-learning.pdf`.

Ren Wu, Shengen Yan, Yi Shan, Qingqing Dang, and Gang Sun. Deep image: Scaling up image recognition. *arXiv preprint arXiv:1501.02876*, 7(8), 2015.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

A EXTENDED AND DETAILED EXPERIMENTAL RESULTS

This appendix details the results of the main experiments shown in Figure 2 and provides the results of many other experiments. For example, the results of the models trained with dropout, but without weight decay are only provided here. Similarly, the results of training with 80 % and 1 % of the data are not shown in Figure 2 either due to space limitations.

Additionally, for most experiments we train a version of the network without batch normalization. These results are provided within brackets in the tables. Note that the original All-CNN results published by Springenberg et al. (2014) did not include batch normalization. In the case of WRN, we remove all batch normalization layers except the top-most one, before the spatial average pooling, since otherwise many models would not converge.

Table 2: Test accuracy of the networks All-CNN and WRN on CIFAR-10 and CIFAR-100, comparing the performance with and without explicit regularizers and the different augmentation schemes. Results within brackets show the performance of the models without batch normalization

Network	WD	Dropout	Aug. scheme	Test CIFAR-10	Test CIFAR-100
All-CNN	yes	yes	no	90.04 (88.35)	66.50 (60.54)
	yes	yes	light	93.26 (91.97)	70.85 (65.57)
	yes	yes	heavier	93.08 (92.44)	70.59 (68.62)
	no	yes	no	77.99 (87.59)	52.39 (60.96)
	no	yes	light	77.20 (92.01)	69.71 (68.01)
	no	yes	heavier	88.29 (92.18)	70.56 (68.40)
	no	no	no	84.53 (71.98)	57.99 (39.03)
	no	no	light	93.26 (90.10)	69.26 (63.00)
	no	no	heavier	93.55 (91.48)	71.25 (71.46)
WRN	yes	yes	no	91.44 (89.30)	71.67 (67.42)
	yes	yes	light	95.01 (93.48)	77.58 (74.23)
	yes	yes	heavier	95.60 (94.38)	76.96 (74.79)
	no	yes	no	91.47	71.31
	no	yes	light	94.76 (93.52)	77.42 (74.62)
	no	yes	heavier	95.58 (94.52)	77.47 (73.96)
	no	no	no	89.56 (85.45)	68.16 (59.90)
	no	no	light	94.71 (93.69)	77.08 (75.27)
	no	no	heavier	95.47 (94.95)	77.30 (75.69)

An important observation from Table 2 is that the interaction of weight decay and dropout is not always consistent, since in some cases better results are obtained with both explicit regularizers active and in other cases, only dropout achieves better generalization. However, the effect of data augmentation seems to be clearer: just some light augmentation achieves much better results than training only with the original data set and performing heavier augmentation almost always further improves the test accuracy, without the need for explicit regularization.

Not surprisingly, batch normalization also contributes to improve the generalization of All-CNN and it seems to combine well with data augmentation. On the contrary, when combined with explicit regularization the results are surprisingly not consistent in the case of All-CNN: it seems to improve the generalization of the model trained with both weight decay and dropout, but it drastically reduces the performance with only dropout, in the case of CIFAR-10 and CIFAR-100 without augmentation. A probable explanation is, again, that the regularization hyperparameteres would need to be readjusted with a change of the architecture.

Furthermore, it seems that the gap between the performance of the models trained with and without batch normalization is smaller when they are trained without explicit regularization and when they include heavier data augmentation. This can be observed both in Table 2 and Table 3, which contains the results of the models trained with fewer examples. It is important to note as well the benefits of batch normalization for obtaining better results when training with fewer examples.

The results in Table 3 clearly support the conclusion presented in Section 2.3: data augmentation alone resists better the lack of training data compared to explicit regularizers. Already with 80% and

Table 3: Test accuracy of the networks All-CNN and WRN on CIFAR-10 and CIFAR-100 when trained with only 80 %, 50 %, 10 % and 1 % of the available training examples. Results within brackets show the performance of the models without batch normalization

Network	Pct. Data	Explicit Reg.	Aug. scheme	Test CIFAR-10	Test CIFAR-100
All-CNN	80 %	yes	no	89.41 (86.61)	63.93 (52.51)
		yes	light	92.20 (91.25)	67.63 (63.24)
		yes	heavier	92.83 (91.42)	68.01 (65.89)
		no	no	83.04 (75.00)	55.78 (35.95)
		no	light	92.25 (88.75)	69.05 (56.81)
		no	heavier	92.80 (90.55)	69.40 (63.57)
	50 %	yes	no	85.88 (82.33)	58.24 (44.94)
		yes	light	90.30 (87.37)	61.03 (54.68)
		yes	heavier	90.09 (88.94)	63.25 (57.91)
		no	no	78.61 (69.46)	48.62 (31.81)
		no	light	90.21 (84.38)	62.83 (47.84)
		no	heavier	90.76 (87.44)	64.41 (55.27)
	10 %	yes	no	67.19 (61.61)	33.77 (19.79)
		yes	light	76.03 (69.18)	38.51 (22.79)
		yes	heavier	78.69 (64.14)	38.34 (26.29)
		no	no	60.97 (41.07)	26.05 (17.55)
		no	light	78.29 (67.65)	37.84 (24.34)
		no	heavier	79.87 (70.64)	39.85 (26.31)
	1 %	yes	no	27.53 (29.90)	9.16 (3.60)
		yes	light	37.18 (26.85)	9.64 (3.65)
		yes	heavier	42.73 (26.87)	9.14 (2.52)
		no	no	38.89 (35.68)	9.50 (5.51)
		no	light	44.35 (29.29)	9.87 (5.36)
		no	heavier	47.60 (33.72)	11.45 (3.57)
WRN	80 %	yes	no	90.27	70.41
		yes	light	94.07	75.66
		yes	heavier	94.57	75.51
		no	no	88.98	66.10
		no	light	93.97	75.07
		no	heavier	94.84	75.38
	50 %	yes	no	86.96	63.60
		yes	light	92.65	70.83
		yes	heavier	92.86	70.33
		no	no	85.56	60.64
		no	light	91.87	69.97
		no	heavier	92.77	70.72
	10 %	yes	no	70.73	34.11
		yes	light	76.00	36.65
		yes	heavier	78.10	38.93
		no	no	60.39	23.65
		no	light	79.19	39.24
		no	heavier	80.29	41.44
	1 %	yes	no	33.45	7.47
		yes	light	34.13	7.50
		yes	heavier	41.02	8.37
		no	no	38.63	9.47
		no	light	43.84	9.91
		no	heavier	47.14	11.03

50% of the data better results are obtained in some cases, but the differences become much bigger when training with only 10% and 1% of the available data. It seems that explicit regularization prevents the model from both fitting the data and generalizing well, whereas data augmentation

Table 4: Test accuracy of the shallower and deeper versions of the All-CNN network on CIFAR-10 and CIFAR-100. Results in parentheses show the difference with respect to the original model.

Network	Explicit Reg.	Aug. scheme	Test CIFAR-10	Test CIFAR-100
All-CNN shallower	yes	no	76.45	51.31
	yes	light	82.02	56.81
	yes	heavier	86.66	58.64
	no	no	85.22	58.95
	no	light	90.02	65.51
	no	heavier	90.34	65.87
All-CNN deeper	yes	no	86.26	49.06
	yes	light	85.04	52.03
	yes	heavier	88.46	51.78
	no	no	83.30	54.22
	no	light	93.46	72.16
	no	heavier	94.19	73.30

provides useful transformed examples. Interestingly, with only 1% of the data, even without data augmentation the models without explicit regularization perform better.

The same effect can be observed in Table 4, where both the shallower and deeper versions of All-CNN perform much worse when trained with explicit regularization, even when trained without data augmentation. This is another piece of evidence that explicit regularization needs to be used very carefully, it requires a proper tuning of the hyperparameters and is not always benefitial.