Self-Supervised Learning for Pretraining Image Classification Task

Aishwarya Budhkar
asb862@nyu.edu

Diego Casabuena
dlc423@nyu.edu

Lekha Iyengar
li471@nyu.edu

Abstract

Self-supervised pretraining for feature extraction has recently gathered a lot of attention in the computer vision community. We implement and experiment with the Split Brain Auto-Encoder [1] model from Zhang et al. (2017). The method splits the input channels into two and feeds them into disjoint sub-networks trained to reconstruct each other's data channels. We use the pretrained features to fine-tune a 1000-class image classifier using only a few labelled examples per class.

1 Introduction

Deep learning algorithms have shown that, when given large collections of labelled data, they can achieve human-level performance on computer vision tasks. However, for many practical tasks, the availability of data is limited. Self-supervised pretraining is a method of training whereby a network predicts a part of its input using an another unseen part, which acts as the label. The objective is to learn useful representations of the data in order to fine-tune with supervision on downstream tasks such as image classification.

We explore a recent method based on image colorisation: the Split-Brain Autoencoder [1]. This method finds useful global features for classification by solving complementary prediction tasks and therefore utilizing all data in the input. The network is divided into two fully convolutional sub-networks and each is trained to predict one subset of channels of input from the other. For fine-tuning, a classifier is added as the last layer. Using a dataset of 96x96 images, with 512k unlabeled images, 64k labelled training images, and 64k labelled validation images, we perform 1000-class classification.

2 Motivation and Related Work

Autoencoders [2] are one of the de-facto architectures for self-supervised pretraining. They reduce the dimensionality of the input data to create an information bottleneck and use a latent space representation to reconstruct the input. Extensions to this method include adding a denoising task to prevent learning the identity function [3], stacking autoencoders to get a deeper network [4], and making the architecture convolutional [5] to learn hierarchical feature representations and exploit the spatial locality of the input. Unfortunately, all these approaches suffer from a domain gap between the pretraining and the fine-tuning task. In fact, there's evidence to suggest that the features required for image reconstruction and classification are generally uncorrelated. [6]

Successful pretraining implies learning more global context. Research has tried to achieve this by solving puzzles [7], teaching class independent features [8], using image redundancy [9], using different non-linearities [10], exploiting the max and argmax data contained by pooling layers [11], adding adversarial noise generators [12], and relating denoising and inpainting [13] amongst other approaches. We considered impainting, which consists of generating large image regions missing from the input. Along this direction, we found context encoders [14], a convolutional version used for semantic inpainting, and cutout [15], which performs significantly better than [11] on the STL-10

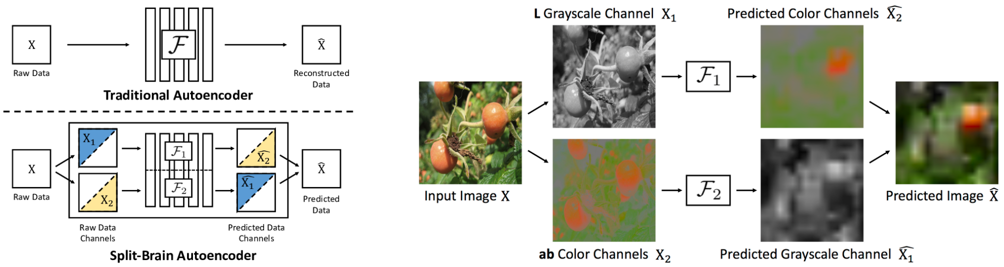


Figure 1: Split Brain Architecture and CIE-LAB Space Visualization

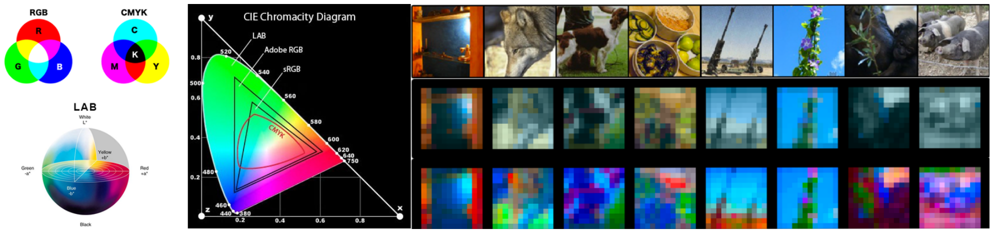


Figure 2: Left to Right: Topology of CIE-Lab, Gamut of CIE-Lab, Examples of Downsampled 12x12 images and their re-scaled LAB counterparts

dataset for image classification. Despite these successes, [14][15] and other inpainting methods suffer from an input handicap: they remove a lot of useful information from the input. Instead of removing spatial data, one can just hide channel data and train the network with maximal information.

Colorization [16][17] is the task of predicting color channels from gray-scale, and [18] showed its application towards self-supervised feature learning. This brought us to Split-Brain Autoencoders [1], a cross-channel prediction method base on colorisation. It exceed the classification performance of the other methods discussed and overcomes both domain gaps and input handicaps.

3 Method

The Split-Brain architecture is depicted in Figure 1 above. The approach consists of splitting an image into two subsets of input channels (2 to 1 for a 3-channel space), preferably using a color space that separates color and luminosity. It then passes each subset through a fully convolutional architecture [19] in order to predict the other subset. To make this prediction, it takes the Cross Entropy loss between the network output and a downsampled, quantized version of the original image (acting as labels). To clarify this with our numbers, the input image has 96x96 input features and each sub-network has 12x12 output features, each of which corresponds to a pixel in a 12x12 downsampled ground truth of the input image. The number of output channels in each sub-network corresponds to the number of classes for each pixel, which is exactly the number of colors into which each channel was quantized into. Fine-tuning consists of adding a classifier on top of the concatenated output of the two sub-networks.

The original paper uses the CIE-Lab color space, which reduces feature extraction to colorization $(L \rightarrow \hat{ab})$ and gray-scale prediction $(ab \rightarrow \hat{L})$. Whereas the use of LAB is effective, it has one drawback. Naturally occurring colors in RGB map to a narrow sub-domain in LAB, meaning that the mapping between these color spaces is non-linear. Therefore, quantizing the transformed images into a reduced number of bins is a very lossy process, as most colors fall into the same bins. To address this, we normalized the range of the bin values over each image. Doing so results in a distorted image, but provides more information to the network and significantly better classification performance.

4 Experiments

We train three architectures in three color spaces, and use a denoising autoencoder as our baseline. Our color spaces include RGB (split R and BG), LAB (split L and ab), and the re-scaled LAB space, where the image is quantized according to the min and max values of the image. We trained three fully convolutional [19] models: AlexNet (140M params) [20] as in [1], ResNet18 (7M params) [21] which performs well on colorization tasks, and SimpleNet (700k params), a shallower 5-layer convnet inspired by [22]. Each model represents a different scale in the number of parameters. Cross Entropy loss is used for pretraining because regression-based losses tend to learn blurry representations.[1].

For hyperparameter tuning, we were conditioned by our compute power. After trying downsized images of size 12x12, 16x16, and 25x25, we stuck with the former. Similarly, we chose a quantization size of 100 for the single channel, and 10 for the dual channel (100 bins over that subset) due to computational restrictions. Nonetheless, the final chosen values are the same as those used in [1]. After a hyperparameter grid search, we determined an optimal batch size of 64, a initial learning rate of 1e-4, and a learning rate decay of 0.1 every epoch of pretraining and fine-tuning.

We report top-1 and top-5 validation accuracy metrics for all three models and all three color spaces described, using the optimal hyperparameters. Further, we report the same metrics for our best model when fine-tuning on different numbers of samples per class. Our baseline is a 5-layer convolutional denoising autoencoder trained with 5% salt-and-pepper noise. We experimentally found that this was the strongest of all the baseline models we trained (including masking, gaussian, and salt-and-pepper noise at 5,10,20, and 50% noise levels).

5 Results

The original paper achieves up to 35% top-1 accuracy on ImageNet Classification, which is similar to our classification task in that the number of classes are the same. However, we trained on less than half the number of unlabeled samples and each image was smaller, a difference of about 72% less pixels per image. Our experiments also trained for much shorter (48 hours). The number of pretraining epochs completed in this time varied widely: FC AlexNet (5), ResNet18 (7), and SimpleNet (11). During finetuning, we applied early stopping, which happened at 5 epochs for all models except AlexNet, which could only complete 2 epochs (limit 12 hours). GPU training was conditioned by the time taken to convert images to Lab, which is currently performed in cpu due to library constraints. Nonetheless, we were able to achieve almost half of the original Split-Brain's top-1 accuracy at 14.04%. Further, the accuracy of the model deteriorates less than linearly as the number of samples per class is decreased.

Architecture	Image Space	FT Epochs	Top 1 Acc.	Top 5 Acc.
FCAlexNet	RGB	2	0.0943	0.2343
FCAlexNet	LAB	2	0.0301	0.0988
FCAlexNet	Re-scaled LAB	2	0.0958	0.2479
ResNet 18	RGB	5	0.1227	0.2711
ResNet 18	LAB	5	0.1351	0.2945
ResNet 18	**Re-scaled LAB**	**5**	**0.1404**	**0.3015**
SimpleNet 18	RGB	5	0.1081	0.2476
SimpleNet 18	LAB	5	0.1093	0.2558
SimpleNet 18	Re-scaled LAB	5	0.1266	0.2799
DAE S&P 5% Noise	RGB	5	0.0653	0.1815

Table 1: Validation Metrics on our Models and Baseline

6 Conclusion

Split-Brain overcomes many of the problems faced by self-supervised image pretraining methods. For this project, we coded the entire method from scratch and made our code publicly available on **Github**. We showed that ResNet18 was able to provide good results much faster than the large FC AlexNet described in the original paper. Our best top-1 accuracy is 14.04% and our best top-5

Samples per class	1	2	4	8	16	32	64
Top-1 Acc.	0.0065	0.0112	0.0199	0.0366	0.0656	0.1012	0.1450
Top-5 Acc.	0.0215	0.0352	0.0574	0.0998	0.1648	0.2374	0.3096

Table 2: Validation Metrics for our Top Model: ResNet18

accuracy is 30.15%. We also show that the method continues to do well even when reducing the number of labelled examples to 4, 2, or 1.

References

[1] Richard Zhang, Phillip Isola, and Alexei A. Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. *2017 IEEE CVPR*, Jul 2017.

[2] G.E. Hinton and R.R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science (New York, N.Y.)*, 313:504–7, 08 2006.

[3] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th ICML*, ICML '08, pages 1096–1103, New York, NY, USA, 2008. ACM.

[4] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.*, 11:3371–3408, December 2010.

[5] Jonathan Masci, Ueli Meier, Dan C. Ciresan, and Jürgen Schmidhuber. Stacked convolutional auto-encoders for hierarchical feature extraction. In *ICANN*, 2011.

[6] R. Ingold M. Alberti, M. Seuret and M. Liwicki. A pitfall of unsupervised pre-training, 2017.

[7] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. *CoRR*, abs/1603.09246, 2016.

[8] A. Dosovitskiy, J. T. Springenberg, M. A. Riedmiller, and T. Brox. Discriminative unsupervised feature learning with convolutional neural networks. *CoRR*, abs/1406.6909, 2014.

[9] Phillip Isola, Daniel Zoran, Dilip Krishnan, and Edward H. Adelson. Learning visual groups from co-occurrences in space and time. *CoRR*, abs/1511.06811, 2015.

[10] Alireza Makhzani and Brendan J. Frey. A winner-take-all method for training sparse convolutional autoencoders. *CoRR*, abs/1409.2752, 2014.

[11] Junbo Jake Zhao, Michaël Mathieu, Ross Goroshin, and Yann LeCun. Stacked what-where auto-encoders. *CoRR*, abs/1506.02351, 2015.

[12] Shixiang Gu and Luca Rigazio. Towards deep neural network architectures robust to adversarial examples, 2014.

[13] Junyuan Xie, Linli Xu, and Enhong Chen. Image denoising and inpainting with deep neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 341–349. Curran Associates, Inc., 2012.

[14] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting. *CoRR*, abs/1604.07379, 2016.

[15] Terrance Devries and Graham W. Taylor. Improved regularization of convolutional neural networks with cutout. *CoRR*, abs/1708.04552, 2017.

[16] R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization. *CoRR*, abs/1603.08511, 2016.

[17] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Colorization as a proxy task for visual understanding. *CoRR*, abs/1703.04044, 2017.

[18] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. *CoRR*, abs/1603.06668, 2016.

[19] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *2015 IEEE CVPR*, pages 3431–3440, June 2015.

[20] 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 NeurIPS 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[22] Seyyed Hossein HasanPour, Mohammad Rouhani, Mohsen Fayyaz, and Mohammad Sabokrou. Lets keep it simple, using simple architectures to outperform deeper and more complex architectures. *CoRR*, abs/1608.06037, 2016.

7 Appendix: Pre-Trained Features by Color Space (11 Epochs)

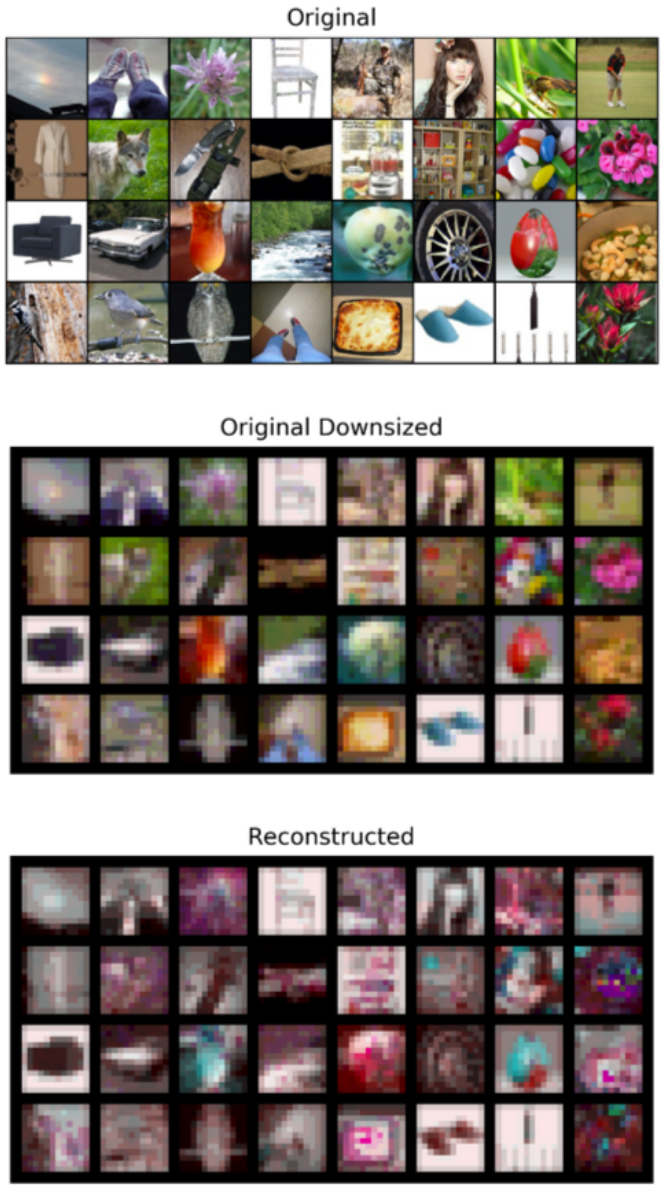
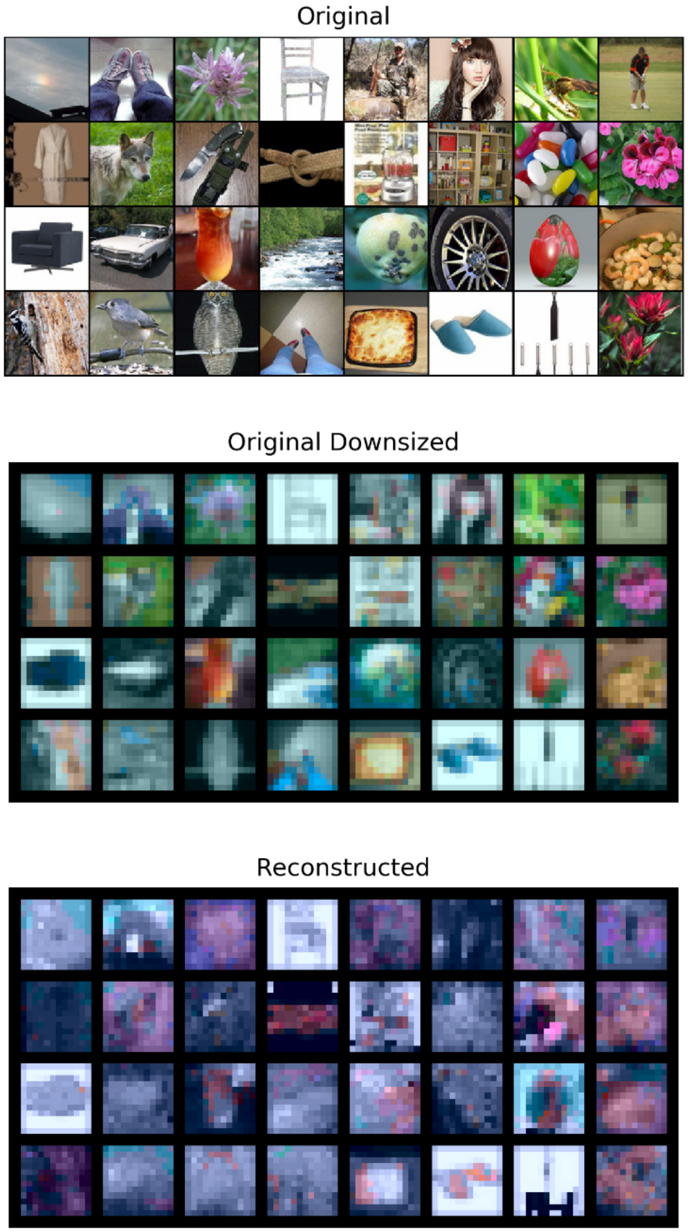


Figure 3: RGB



Original

Original Downsized

Reconstructed

Figure 4: CIE-LAB

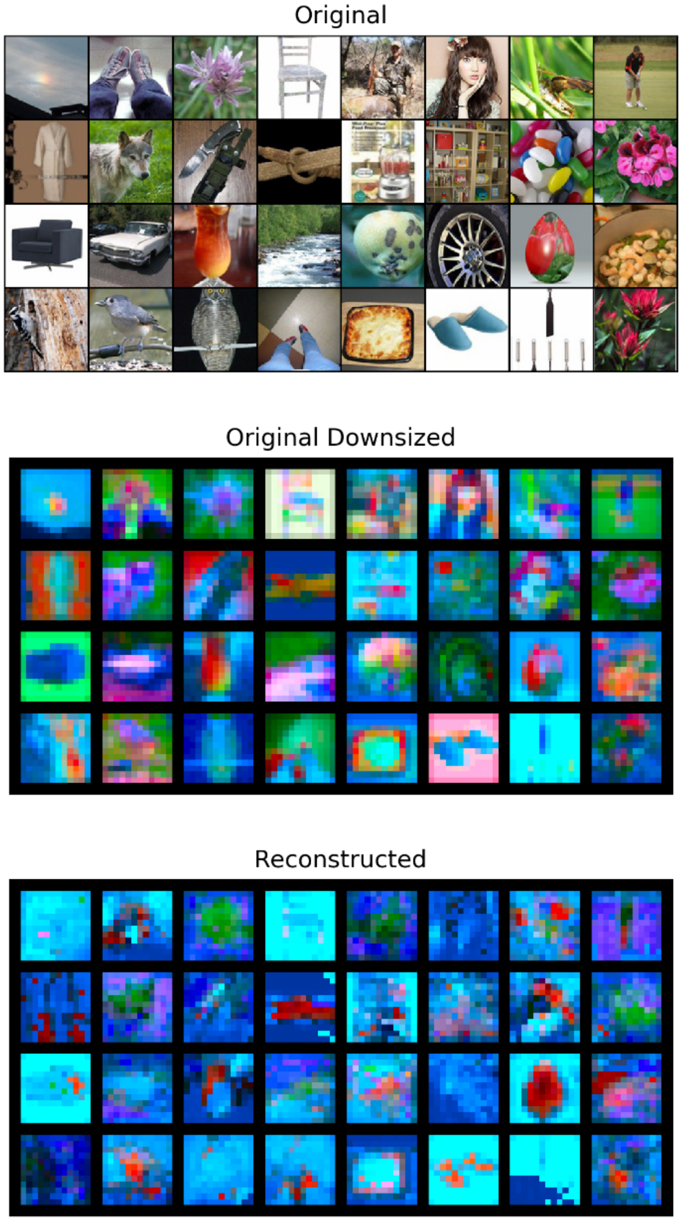


Figure 5: Re-Scaled CIE-LAB