

Using Learned Image Compression for Training Accurate and Robust Convolutional Neural Networks

Navneeth Maudgalya



Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2021-98

<http://www2.eecs.berkeley.edu/Pubs/TechRpts/2021/EECS-2021-98.html>

May 14, 2021

Copyright © 2021, by the author(s).
All rights reserved.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

Acknowledgement

I would like to thank my research advisor, Professor Gerald Friedland, and reader, Professor Kannan Ramchandran, for supporting me with their time and feedback during an unpredictable year. Thank you to Spencer Kent and Professor Bruno Olshausen for their invaluable mentorship over the last two years.

Thank you to all my friends back home for giving me the mental strength to make the most of an online school year. Thank you to my grandparents, parents and sister for always allowing and encouraging me to love what I learn and learn what I love.

Finally, thank you UC Berkeley for taking me on a ride whose highs and lows I will always cherish and never forget. Go Bears!

Using Learned Image Compression for Training Accurate and Robust Convolutional Neural
Networks

by

Navneedh Maudgalya

A thesis submitted in partial satisfaction of the

requirements for the degree of

Master of Science

in

Electrical Engineering and Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Gerald Friedland, Chair
Professor Kannan Ramchandran

Spring 2021

**Using Learned Image Compression for Training Accurate and Robust
Convolutional Neural Networks**

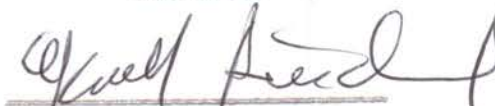
by Navneeth Maudgalya

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences,
University of California at Berkeley, in partial satisfaction of the requirements for the
degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:



Professor Gerald Friedland
Research Advisor

May 14, 2021
(Date)



Professor Kannan Ramchandran
Second Reader

5/8/2021
(Date)

Using Learned Image Compression for Training Accurate and Robust Convolutional Neural
Networks

Copyright 2021
by
Navneedh Maudgalya

Abstract

Using Learned Image Compression for Training Accurate and Robust Convolutional Neural Networks

by

Navneedh Maudgalya

Master of Science in Electrical Engineering and Computer Science

University of California, Berkeley

Professor Gerald Friedland, Chair

Convolutional neural networks (CNNs) for image classification are trained to learn compact and informative representations from high-dimensional images to make accurate predictions. CNNs are burdened with learning to distinguish between information relevant for classification and noise. By training CNNs on perceptually compressed images, we show several benefits to removing irrelevant content from our data prior to training the model. We generate compressed data sets using JPEG and learned compression models at various quality levels to train image classification models. First, we explore these classifiers' ability to maintain high accuracy when trained on compressed data. Next, we compare the performance of classifiers trained on compressed and uncompressed data as the number of trainable parameters are reduced. Finally, we show that compressed data can be used as a form of data augmentation, enabling gains in robustness to high and low frequency image distortions while preserving accuracy on the original data. By thinking more about the quality of information contained in our training data, we can reduce the usage of large, high-quality image data sets, avoid highly parameterized neural networks, and train robust models. We hope these findings will motivate the future use of compressed data for training deep learning models.

To Niyantri, Amma, and Appa - thank you

Contents

Contents	ii
List of Figures	iii
List of Tables	iv
1 Introduction	1
2 Related Works	3
2.1 Data Compression	3
2.2 Data Augmentation	4
3 Image Compression for Classification	6
3.1 Introduction	6
3.2 Methods	9
3.3 Results	11
4 Image Compression for Data Augmentation	18
4.1 Introduction	18
4.2 Methods	18
4.3 Results	19
5 Conclusion	25
6 Future Work	26
6.1 Improving Learned Compression Models for Classification	26
6.2 Interpreting the Benefits of Learned Compression	27
6.3 Alternative Training Paradigms with Compressed Data	27
Bibliography	29

List of Figures

3.1	Procedure for Training Classifiers with Learned Image Compression	10
3.2	Rate-distortion Curves	12
3.3	Examples of Compressed Images	12
3.4	Training Classifiers with Original, BLS, BMJH and JPEG-Compressed Data . .	13
3.5	Distortion vs Classification Accuracy for JPEG, BLS, and BMJH	14
3.6	Training CNNs with Fewer Parameters and Compressed Data	16
4.1	(BDA) Average Accuracy over all Image Deformations	20
4.2	(BDA) Classifier Accuracy on all Image Deformations	21
4.3	(BDA) Comparing Model Accuracy on Corrupt and Clean Data	22
4.4	(BDAD) Average Accuracy over all Image Deformations	23
4.5	(BDADS) Average Accuracy over all Image Deformations	23
4.6	(BDAD) Accuracy on All Image Deformations	23
4.7	Fourier Magnitude Spectrum of Augmented Data	24
6.1	Training with Compressed Images and Human Labels	28
6.2	Contrastive Training with Compressed Images	28

List of Tables

3.1	Bit Rates of Compressed Images	10
3.2	Reduced Parameter Variants of VGG11 Models	15
3.3	Reduced Parameter Variants of Resnet18 Models	15
4.1	Data Augmentations and Quality Parameters	19
4.2	(BDA) Ranked Data Quality Parameters	20
4.3	(BDAD) Optimal Data Quality Parameters	22

Chapter 1

Introduction

Convolutional neural networks (CNNs) have been shown to be successful in tasks such as object detection, image segmentation, and pose estimation [19]. However, to discover and utilize components of the image signal relevant for effective classification, large amounts of data and highly parameterized models are often deemed essential, leading to data storage issues, data labeling costs, and computational burden. Though large machine learning models [4] and data sets are widely used, many have begun to recognize the social and environmental implications of overusing resources for machine learning. Furthermore, these models' performance degrades significantly when evaluated on out-of-distribution data [40][29]. This leads to faulty predictions and decision making when models are deployed in the real world. In this paper, I describe a data preprocessing procedure to address these pertinent issues.

Toneva et al. [41] shows that not all data is equally useful for training CNNs for image classification. By discriminating between informative and non-informative data, models can learn from fewer samples while maintaining accuracy. Non-informative data includes noisy, redundant, or out-of-distribution samples that may hinder learning accurate classification. Adaptive data selection strategies are employed while training to locate and train on informative data. Such strategies include training auxiliary neural networks to estimate the likelihood of data being useful for training [47][10] and formulating data querying strategies to identify uncertain samples [34][27][18]. Such methods require more computing power to train auxiliary models and search for useful data, must balance sampling diverse and informative data and are not effective throughout the model training procedure. For those who wish to train on all available data while reducing the size of their data set, image compression is a worthy alternative. By substituting images in the original training data with compressed versions, the total number of images in the data set remains the same yet the size of each image file is smaller and the quality of each image is partially degraded as per the image compression method. We show that CNNs maintain decent classification accuracy on held-out CIFAR-10 and Tiny ImageNet data and the potential to reduce the number of model parameters when training classifiers with various levels of compressed data.

When evaluated on images with visually perceivable deformities [31] or imperceptible added noise [13], CNNs frequently misclassify images that humans classify with relative

ease. Despite being trained on a large corpus of data, image classification models are often not robust to natural deformations such as blur, additive noise, contrast and scale though these corruptions preserve the semantics of the images. Traditional CNN architectures employ translational invariance and other image recognition models incorporate relevant inductive biases to improve robustness to different corruptions [28][17][39]. As an alternative to modifying the model's architecture, training data can be augmented to include certain perturbations which the classifier learns to be invariant to. However, these data augmentation strategies often remain ineffective in dealing with different classes of corruptions as models tend to overfit to perturbations encountered during training. On the other hand, training on a large variety of perturbations may burden the model to learn more invariances resulting in underfitting [12]. We suggest using image compression as a form of data augmentation. If compressed images retain perceptually relevant information useful for classification, models can be trained on this distilled input. By supplying the sufficient input content necessary for correct classification, we hope the model learns invariant representations robust to various perturbations, offering a more general and less compute-intensive data augmentation option. We investigate improvements in model robustness afforded by image compression compared to other data augmentation techniques.

This paper explores the effect of using image compression for training data on the classification model's architecture, generalization performance, and robustness. In Chapter 2, we present previous research on relevant image compression methods, the use of image compression in deep learning and data augmentation. In Chapter 3, we observe how compressed training data affects classification accuracy and the number of sufficient trainable parameters for the classifier. In Chapter 4, we use image compression to augment our training data and observe the model's robustness to various perturbations.

Chapter 2

Related Works

2.1 Data Compression

Preprocessing with lossy image compression degrades images' perceptual quality. However, the extent of degradation can often be adjusted to influence the training and testing of downstream CNNs. Dodge et al. [8] explores the resiliency of CNNs when tested on JPEG-compressed images. Compared to distortions such as Gaussian noise and blur, they find that several CNNs are fairly resilient to artifacts introduced by JPEG above quality levels of 20. Friedland et al. [11] uses Helmholtz free energy to estimate the number of bits of noise contained in images and shows that using JPEG at the appropriate quality levels can remove this noise, enabling CNNs to maintain accuracy with fewer parameters. Pistono et al. [24] trains DNNs on partially JPEG-compressed images, observing the impact of quality level and extent of compression on a CNN's classification accuracy for CIFAR-10 data. JPEG2000, a compression standard extending JPEG, is used to compress histopathological whole-slide images to train CNNs for cancer detection, reducing data storage costs while performing well on both low and high quality test data [48].

Despite the prevalence of JPEG and its ability to compress images without sharply degrading quality, it frequently introduces ringing artifacts near edges due to the removal of high frequency components and blocking artifacts caused by independently compressing 8x8 sized blocks in the image. Alternatively, deep learning methods have been shown to more effectively learn image compression, preserving higher perceptual quality in compressed images. The neural network outlined in [1] consists of a convolutional encoder, a uniform quantizer, and a convolutional decoder trained to minimize the entropy of the encoder's compressed representation and a pixel-level reconstruction loss. [3] suggests a similar approach but incorporates a prior on the latent compressed representation. Several other learned image compression models have shown further improvement utilizing a 3D-CNN model that incorporates context [25], hierarchical latent variable models [20], and generative adversarial networks [26]. Compared to JPEG, learned image compression is a relatively new area of research, but these models' ability to remove undesired perceptual artifacts introduced at

low bit rates and flexible architectures may impact the future of image compression and data storage. An in-depth review of the compression techniques used are discussed in Chapter 3.

In [38], the neural network from [1] is trained to minimize a classification loss as opposed to the pixel-level reconstruction loss, learning compressed representations via classification. Though this paper connects learned compression with classification, their model is trained using the images' class labels. However, our objective is to understand the effectiveness of using learned compression to generate compressed images while remaining agnostic to the downstream learning task.

2.2 Data Augmentation

Though several data augmentation strategies have been proposed utilizing style transfer, random cropping, translation, rotation, color space and geometric transformations [35], models trained using these approaches are often not robust to different types of image corruptions since their performance is biased by the properties of the augmented training data. [46] shows that models trained on low-pass filtered images are skewed towards using low frequency information for classification, increasing robustness to a subset of corruptions including Gaussian noise and blur. Similarly, standard CNNs trained on certain types of blurred images do not generalize to other forms of blur [43].

However, certain data augmentation techniques effectively improve the quality of the training signal to learn invariance to different corruptions. Lopes et al. [23] describes a simple augmentation technique that applies Gaussian noise to random patches in the original image. The authors propose that this augmentation allows the model to be invariant to high frequency noise yet sensitive to high frequency information during classification. Models trained in this manner have high accuracy on both corrupt and clean data, which is useful in the real world.

Beyond using simple image transformations for data augmentation, other techniques learn augmentation strategies using auxiliary models and feedback from the classifier. [32] jointly trains a classifier and adversarial noise generator network to learn noise distributions that maximally confuse the classifier when added to images which boosts classification efficacy. For a given data set, [6] uses a learned search algorithm to find the types, order, and magnitude of image transformations that maximize performance on a noisy validation set. This method achieves state-of-the-art accuracy on perturbed versions of CIFAR-10, CIFAR-100, and ImageNet.

Since training data inherently biases how models classify data, data augmentation strategies must enable neural networks to learn the appropriate amount of invariance to a broad range of deformations by selectively preserving high and low frequency information and removing high and low frequency noise. Moreover, it is beneficial in practice for the data augmentation approach to be agnostic to the data set and lightweight to deploy.

JPEG compression has been used to preprocess data to effectively prevent misclassification when input images are adversarially perturbed [7][9]. Adversarial images are created by

adding imperceptible noise to an image to prevent the model from classifying correctly. To the best of our knowledge, image compression has not been used to augment data to improve robustness to images with perceivable defects.

Chapter 3

Image Compression for Classification

3.1 Introduction

During the training of deep neural networks (DNNs) for classification, [36] proposes that DNNs learn intermediate representations that compress the content in the input such that relevant information for correct classification is preserved. Given access to these intermediate representations, we could use these classification-relevant features as input to train a classifier, minimizing training time and model size. However, access to such compressed representations are not available prior to training so we alternatively explore the effects of detaching compression from classification. We first compress images and then train a CNN for classification using these compressed images and their corresponding class labels. In this chapter, we ask the following two questions:

- Does compressing the input images outside the context of the classification model retain semantic content useful for classification?
- Does using image compression as a data preprocessing technique impact the model's generalization performance and ability to train with fewer parameters?

Image Compression

To train on all images while reducing the total size of training data, the size (measured in bits) of each image file must be reduced. Image compression techniques reduce the number of bits per pixel, or bit rate, of an image while reducing its perceptual quality, commonly characterized by the rate-distortion trade-off. As an image is compressed, the more it visually differs from the original, intuitively making it difficult for a human or CNN to accurately perceive it. However, the resulting visual artifacts and quality of the compressed images for a given bit rate depend on the type of image compression used. The three compression techniques, JPEG, BLS, and BMHJ, and descriptions of relevant distortion metrics are described below. Using these compression methods, images can be compressed to various bit

rates with different perceptual qualities, influencing the performance of downstream data-driven learning tasks.

JPEG

The JPEG compression standard is a discrete cosine transform-based method usable for lossy compression of colored or gray-scale images [17]. This approach typically consists of the following steps:

1. Colored images are converted from RGB to the YCbCr color space where the three channels correspond to brightness, blue chrominance and red chrominance.
2. The image is split into 8x8 blocks and each block is transformed from the spatial domain to the frequency domain using the Discrete Cosine Transform.
3. Based on the chosen quality level, a 8x8 quantization table is constructed. The DCT coefficients are divided by the quantization table and rounded to the nearest integer. At lower quality levels, quantization preserves fewer of the high spatial frequency components in each 8x8 block.
4. Entropy encoding arranges the quantized DCT coefficients in zigzag order and uses Huffman coding to produce the final condensed representation of the image.

To decompress the image, the inverse of each of the previously mentioned steps is performed for which the previously used quantization table and Huffman table must be accessible. Although JPEG is widely used, it is a linear transform and is not learned from the statistics of the images.

Learned Lossy Image Compression

The data-driven training paradigm and non-linearity provided by DNNs make them effective for image compression. Here we describe two such DNNs outfitted for image compression and used in the rest of the paper.

The BLS model [1] is trained on a corpus of images to jointly optimize rate and distortion. Similar to the architecture of an autoencoder, BLS contains an upsampling and downsampling convolutional neural network whose parameters are jointly optimized. The encoder, which performs the analysis transform, takes the image as input and passes it through three layers of convolution, downsampling and divisive normalization. The encoder's continuous vector output is uniformly quantized and entropy coded assuming a fully-factorized code space to produce the final compressed representation. Since the gradients of the quantization function will be zero, rendering stochastic gradient descent useless for training, uniform noise is independently added to the quantized output to provide a continuous approximation. The decoder, which performs the synthesis transform, passes this continuous approximation through approximately inverse operations to reconstruct the image. As opposed to training

this model with commonly used batch normalization, a biologically-inspired trainable generalized divisive normalization function is employed. The objective function to minimize is of the form $R + \lambda D$ where R represents rate, or the entropy of the compressed encoding, D represents distortion, or the error in reconstructing the input image and λ is a hyperparameter. The metric used for distortion is mean squared error, though other semantically relevant metrics can be used. The rate-distortion curves generated by this model on natural images are better than those produced by JPEG and JPEG2000.

The BMHJ model [3] extends the BLS model by learning a hyperprior that characterizes the spatial dependencies of the image in the latent representation. The hyperprior controls the amount and content of additional information passed from the encoder to the decoder that if provided improve the expected entropy coding. An encoder and decoder similar to those used in the analysis and synthesis transforms of BLS are stacked on top of the BLS autoencoder, allowing relevant side-information to pass through and modify the compressed representation. This stacked autoencoder is jointly trained so that the hyperprior and compression models are optimized together to minimize $R + \lambda D$. Experimental rate-distortion curves on natural images from the Kodak data set show improvements over JPEG, BLS, and other neural network based compression models. Compared to using a factorized prior as found in BLS, the hyperprior does not remove visual artifacts introduced by compression but rather preserves more detail at lower bit rates. Hence, the learned non-linear transformations and vector quantization found in BLS remain the significant contributors to the success of end-to-end learned compression models.

Distortion Metrics

Mean squared error (MSE), peak signal-to-noise ratio (PSNR) and structural similarity index measure (SSIM) are used in this paper to quantify the error in reconstructing the original image from a compressed representation. Even though an image’s perceptual quality may be subjective, these metrics are widely used and allow us to quantify perceptual differences between images. Assume \hat{X} is the reconstructed image and X is the original image.

MSE computes the pixel-wise squared differences between \hat{X} and X . Although MSE possesses many properties useful for optimization such as convexity and differentiability, it is sensitive to the intensity of the image and invariant to significant perceptual distortions in images [44].

$$MSE(\hat{X}, X) = \frac{1}{I \cdot J} \sum_{i=1}^I \sum_{j=1}^J (\hat{X}_{(i,j)} - X_{(i,j)})^2$$

PSNR scales the MSE according to the range of pixel intensities (L) and therefore only provides more information than MSE if images with different pixel ranges are compared.

$$PSNR(\hat{X}, X) = 10 \log_{10} \frac{L^2}{MSE(\hat{X}, X)}$$

SSIM [49] draws inspiration from the human visual system to identify and compare structural content of images. SSIM extracts the luminance (μ), contrast (σ), and structure (γ) for both images and defines a comparison function for each of the three features ($l(X, \hat{X}), c(X, \hat{X}), s(X, \hat{X})$). The SSIM score is given by combining the the results of these comparisons. Typically, SSIM is computed between corresponding local regions of the images and averaged [45]. Compared to PSNR, SSIM is more correlated with human visual perception [44].

$$\mu_X = \frac{1}{I \cdot J} \sum_{i=1}^I \sum_{j=1}^J X_{(i,j)}$$

$$\sigma_X = \left(\frac{1}{I \cdot J - 1} \sum_{i=1}^I \sum_{j=1}^J (X_{(i,j)} - \mu_X)^2 \right)^2$$

$$\gamma_X = \frac{(X - \mu_X)}{\sigma_X}$$

$$SSIM(\hat{X}, X) = l(\hat{X}, X) \cdot c(\hat{X}, X) \cdot s(\hat{X}, X)$$

3.2 Methods

In this section, I provide an overview of the experimental procedure and describe the datasets, classification model architectures and training details used.

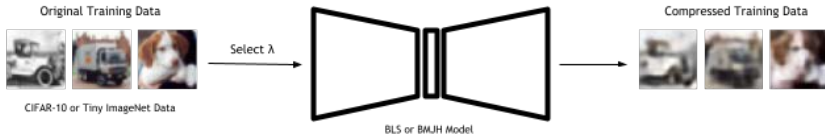
Procedure Overview

The BLS model was trained on CIFAR-10 training images and the BMHJ model was trained on Tiny ImageNet training images at various values of λ [2]. By varying this hyperparameter, BLS and BMHJ are trained to compress images at different bit rates and distortion. Using the trained BLS and BMHJ models, CIFAR-10 and Tiny ImageNet training data sets are compressed at different levels. JPEG compression was used at six quality levels to generate compressed training data for both datasets. Two CNN image classification models were trained on these compressed datasets and are evaluated on held-out original test data. For parameter reduction experiments, classification models were evaluated on held-out compressed test data. See Figure 3.1 for a diagram of our procedure.

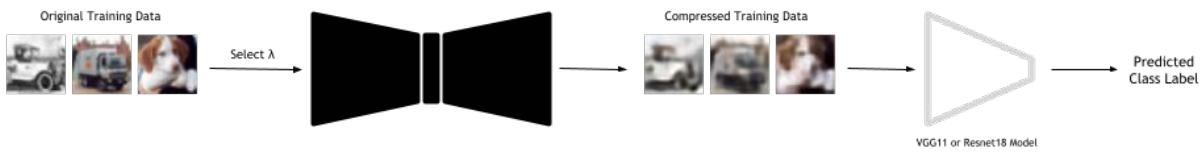
Data

We used two popular data sets containing natural images: CIFAR-10 [21] and Tiny ImageNet. CIFAR-10 contains 60,000 32x32 images of which 50,000 are training images and 10,000 are test images. Each image belongs to one of 10 classes. Tiny ImageNet is a smaller version of the well-known ImageNet data set [33] and contains 100,000 training images and 10,000 test images. Each image is size 64x64 and belongs to one of 200 classes.

(1) Compression Model Training



(2) Classification Model Training



(3) Classification Model Testing



Figure 3.1: Procedure for Training Classifiers with Learned Image Compression

Dataset	Learned Compression (BLS/BMHJ)	JPEG
CIFAR-10	0.26, 0.52, 0.69, 1.08, 1.49, 1.63, 1.91	0.36, 0.48, 0.75, 1.0, 1.5, 1.91
Tiny ImageNet	0.22, 0.6, 0.93, 1.38, 1.99, 2.18, 2.64	0.27, 0.38, 0.6, 0.94, 1.4, 2.77

Table 3.1: Bit Rates of Compressed Images

JPEG compression was used at six quality levels (1, 5, 10, 20, 40, 60) on CIFAR-10 and Tiny ImageNet images. To use learned image compression, BLS and BMHJ were trained at various λ (0.001, 0.003, 0.005, 0.012, 0.033, 0.05, 0.1) and used to compress CIFAR-10 and Tiny ImageNet images respectively. To generate these compressed data sets, the original images were fed into the learned compression model and the reconstructed images output by the decoder were added to the corresponding compressed data set. To reduce data storage in practice, the encoder’s compact vector outputs should be saved and the trained decoder should be used downstream to reconstruct images prior to image classification. Although one can use the encoder’s output to train classification models, we are interested in understanding how compression-induced changes in an image’s quality affects classification so we focus our work on images.

For each image in a dataset, the number of bits were counted and averaged to calculate average bit rate per dataset. The average bit rates for all compressed images can be found

in Table 3.1. To ensure a fair comparison between the average bit rate measurements for different compression techniques, only the bits representative of image-specific content were accounted for, not those used by decoding mechanisms or header segments. For JPEG compression, bytes corresponding to the Huffman table and quantization tables were not used. Similarly, the size of the BLS and BMHJ decoders were not taken into account.

Data augmentation was applied after generating compressed data sets and prior to training the classification model. For CIFAR-10 images, random cropping and random horizontal flip were used. For Tiny ImageNet images, random rotation and random horizontal flip were used.

Image Classification Models

The two image classification models used in experiments were VGG11 and Resnet18. Both models were primarily based on the original architectures specified in [22] and [15] but slightly modified to account for the different spatial resolution of images in both datasets. Models trained on CIFAR-10 used a batch size of 128 and 30 training epochs. For Tiny ImageNet data, the Resnet18 model used a batch size of 200 and 30 training epochs while the VGG11 model used a batch size of 300 and 30 training epochs. All models were trained using the stochastic gradient descent with a learning rate of 0.01, a momentum factor of 0.9, and a L2 weight regularization penalty of $5e-4$. For a given dataset, each model was trained three times from scratch to account for fluctuations in model behavior due to random weight initializations and shuffling of training data.

3.3 Results

Comparing Rate-Distortion Curves

To compare how different compression methods trade-off bit rate and distortion, we observe the rate-distortion curves on CIFAR-10 (left) and Tiny ImageNet (right) using SSIM (top) and PSNR (bottom) as the distortion metrics in Figure 3.2. Error bars represent distortion values within one standard deviation of the average across training images. Examples of compressed images are also shown (Figure 3.3). Each column contains images with approximately the same bit rate and for each image, the corresponding versions in the top row were compressed using the appropriate learned compression method and the versions in the second row were compressed using JPEG. At all bit rates, both distortion metrics suggest that BLS and BMHJ-compressed images are on average more perceptually similar to the original images when compared to JPEG-compressed images. At lower bit rates, the difference between the quality of JPEG and BLS-compressed images is larger which can partly be attributed to the blocking artifacts created by JPEG. Reducing precision in JPEG reveals the linear basis functions which do not naturally characterize the structure of the original image. Learned image compression methods use non-linear transformations to simplify the image

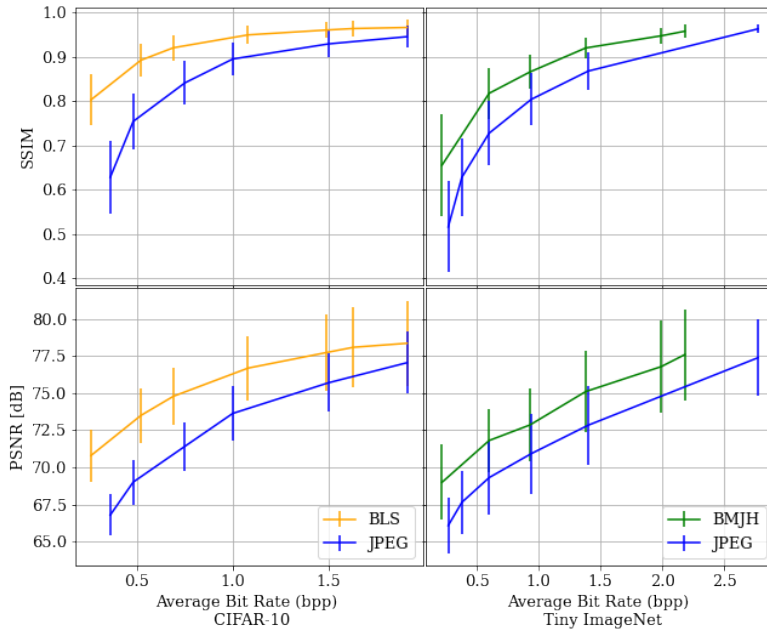


Figure 3.2: Rate-distortion Curves

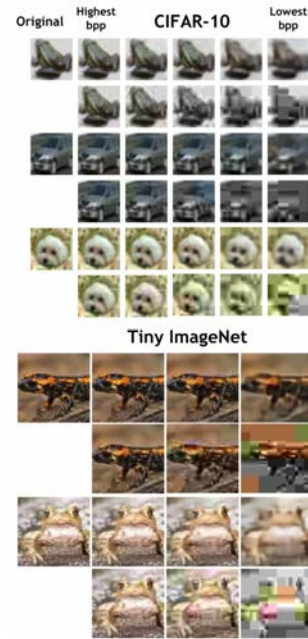


Figure 3.3: Examples of Compressed Images

structure so fewer bits are necessary for faithful representation. Though average distortion is a useful metric, variation in distortion indicated by standard deviation characterizes the overall quality of the compressed data set.

Training with Compressed Data

In this section, we explore the impact of data compression on training classification models. Using the procedure outlined in Figure 3.1, we compress data using JPEG, BLS and BMJH to train VGG11 and Resnet18 convolutional neural networks.

Figure 3.4 shows the maximum test classification accuracy achieved by neural networks trained on original data (black line) and varying levels of compressed data. Results using VGG11 (top) and Resnet18 (bottom) for CIFAR-10 (left) and Tiny ImageNet (right) are shown. For models trained on Tiny ImageNet data, we report the top-5 accuracy which labels a prediction as correct if the correct class label is among the model’s top 5 predictions.

Training models on BLS and BMHJ-compressed data provides higher accuracy at all levels of compression for both data sets. At lower bit rates for compressed CIFAR-10 images there is a larger gap in performance, similar to the corresponding rate-distortion curve suggesting the importance of avoiding visual artifacts for classification. However, results

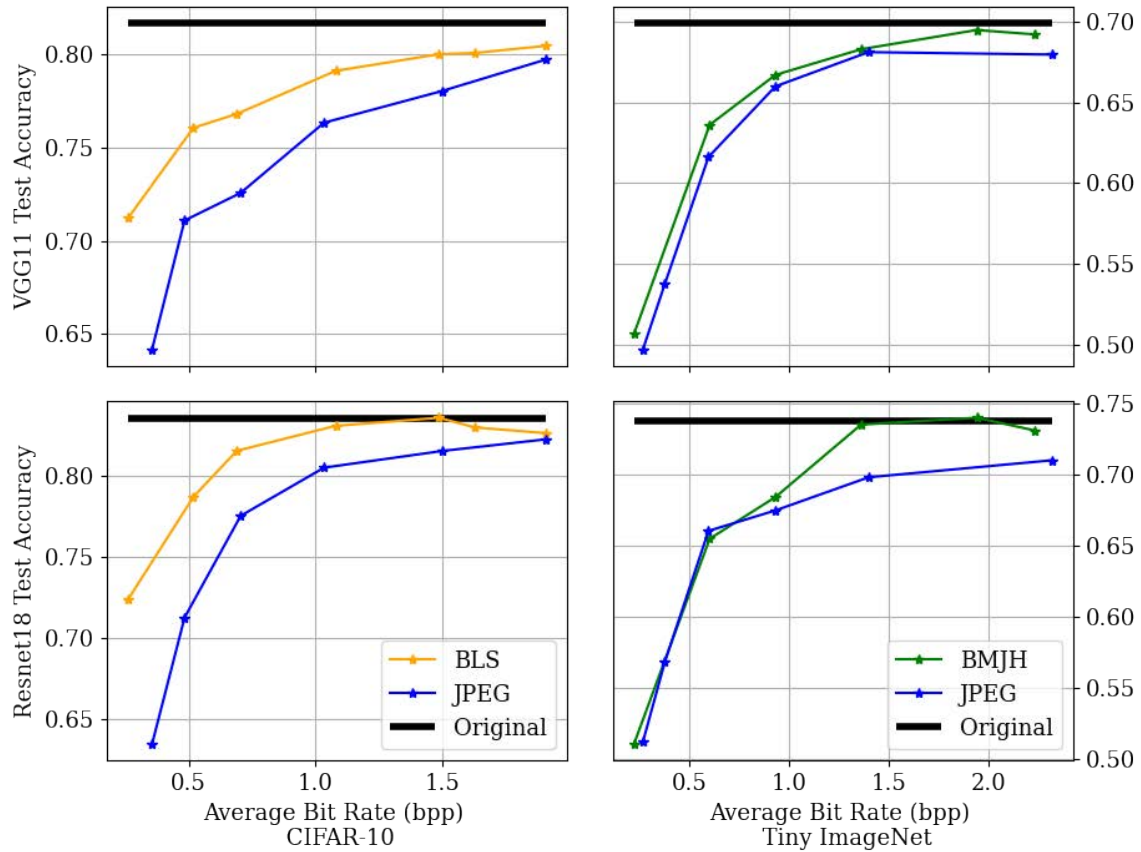


Figure 3.4: Training Classifiers with Original, BLS, BMJH and JPEG-Compressed Data

using BMJH on Tiny ImageNet are marginally better than results using JPEG especially at lower bit rates. Since Tiny ImageNet images belong to one of 200 classes, applying the same models (VGG11 and Resnet18) used on CIFAR-10 might not be effective. Especially, since at lower bit rates, Tiny ImageNet images are heavily distorted ($SSIM \approx 0.5 - 0.7$) for JPEG and BMJH, the type of image compression may have an insignificant effect on the classifier's performance and its architecture may be the limiting factor. On both data sets, the Resnet18 models perform better than the VGG11 models, achieving accuracies on compressed data equal to or greater than the model trained on the original data. However, test accuracy does not monotonically increase with respect to the average bit rate. It is difficult to conclude whether this is due to the stochasticity in model training or due to removal of deleterious content at lower bit rates which may hurt accuracy if unremoved at higher bit rates. The

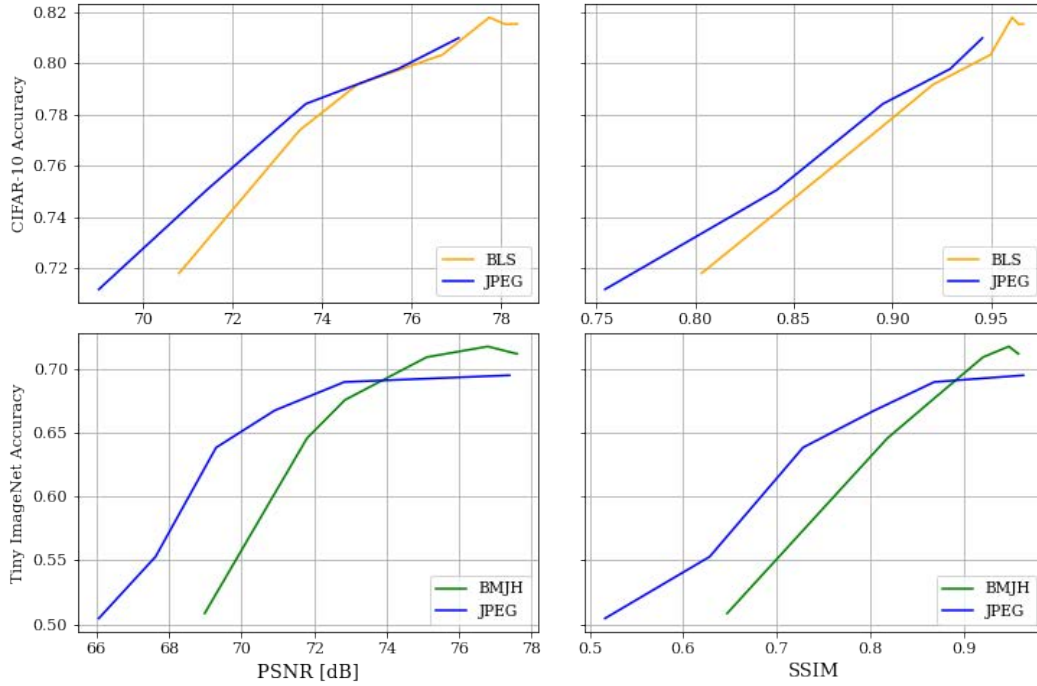


Figure 3.5: Distortion vs Classification Accuracy for JPEG, BLS, and BMJH

trend in accuracy by all compression models corroborate the findings of [11] that the model’s classification accuracy is a logarithmic function of compression level. The point at which accuracy sharply drops off suggests the compression level at which relevant semantic content begins to be removed as opposed to noise.

Figure 3.5 shows the relationship between training data quality, quantified by PSNR (left) and SSIM (right), and the accuracy of classification models trained on these data sets. As expected, the classification accuracy is positively correlated with images’ perceptual quality. However, for a given low quality level, JPEG-compressed images yield models with higher classification accuracy. At higher quality levels, accuracy for a given distortion level becomes similar, with BLS and BMHJ-compressed data eventually producing better classification models. The varying effects of each compression method on classification suggests that these distortion metrics are not accurate predictors of model accuracy, despite the two quantities being correlated. At lower quality levels, JPEG preserves certain classification-relevant information while discarding redundant or unnecessary content as opposed to the learned compression methods. Further work is required to investigate the differences in se-

semantic content preserved by both compression methods. Though JPEG often seems more effective for a given level of distortion, learned non-linear compression methods can compress data with high fidelity at lower bit rates, allowing us to effectively train models with highly compressed data.

Training with Compressed Data and Reduced Model Parameters

Using compressed training data reduces data storage costs and may also reduce the number of model parameters necessary without significantly degrading accuracy. If learned compression preserves classification-relevant content while removing noise, fewer parameters are needed to effectively train the classifier as shown in [11].

Five versions of the VGG11 model and four versions of the Resnet18 model with different number of parameters were created. For VGG11-based models, a convolutional layer module (conv-module) consists of a convolutional layer, batch normalization, and ReLU activation. A Resnet18 conv-module consists of a convolutional layer, batch normalization, ReLU activation, convolutional layer, batch normalization, summation with skip connection, and ReLU activation. To define conv-modules for both networks, the number of output channels shared across all the convolutional layers in a conv-module is specified. The model names, network architectures and number of trainable parameters are listed in Table 3.2 and Table 3.3.

Each model was trained using the training paradigm described in Figure 3.1 but was tested on held-out data compressed to the same extent as the training data. Training on

Model Name	Network Architecture	Number of Parameters
VGG11 (Original)	[64, 128, 256, 512, 512, 512, 512]	9,231,114
VGG11-1	[64, 128, 256, 256, 256]	1,555,466
VGG11-2	[64, 128, 128, 128, 256]	669,962
VGG11-3	[64, 64, 64, 128, 128]	300,554
VGG11-4	[32, 32, 32, 64, 128]	113,610
VGG11-5	[16, 16, 16, 32, 128]	48,426

Table 3.2: Reduced Parameter Variants of VGG11 Models

Model Name	Network Architecture	Number of Parameters
RES18 (Original)	[64, 64, 128, 128, 256, 256, 512, 512]	11,173,962
RES18-1	[64, 128, 256, 512]	4,903,242
RES18-2	[64, 128, 256]	1,235,274
RES18-3	[64, 128]	326,374
RES18-4	[64]	116,810

Table 3.3: Reduced Parameter Variants of Resnet18 Models

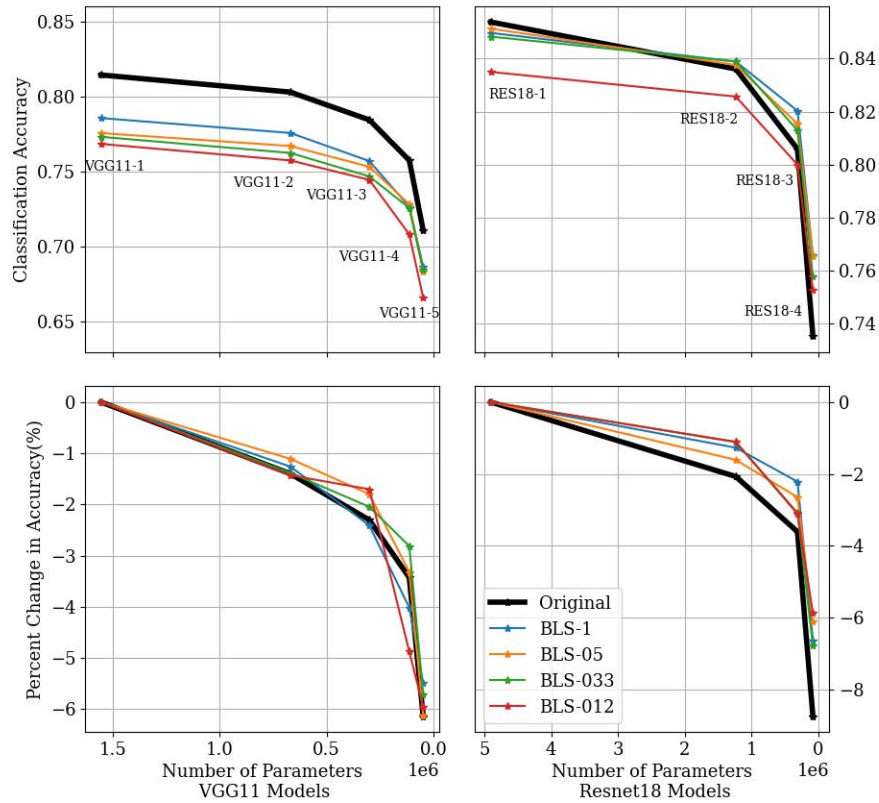


Figure 3.6: Training CNNs with Fewer Parameters and Compressed Data

compressed data and testing on original data requires larger models. If models are evaluated on original data containing both relevant content and irrelevant noise, more parameters are required for correct classification so parameter reduction is not effective. In practice, generating compressed training and test data also saves more space.

We selected four compressed CIFAR-10 data sets ($\lambda = 0.012, 0.033, 0.05, 0.1$) whose average bit rates were larger than the point at which accuracy sharply drops off in Figure 3.4. This ensures that the accuracy does not significantly drop when testing on compressed data.

In Figure 3.6, we present accuracy and percent change in accuracy for VGG11 (left) and Resnet18 (right) classifiers as model and data are compressed. Percent change in accuracy provides a fair way to compare models' performance degradation.

As opposed to the Resnet18 models, all VGG11 models report a higher accuracy when

trained and tested on the original data. As parameters are removed from the Resnet18 model, the accuracy of models trained on original data degrades more rapidly compared to the models trained on compressed data. However, we do not observe the same clear trend for VGG11 models, though certain parameter reductions result in a larger percent decrease in accuracy when using original data compared to compressed data.

In general, the percent decrease in accuracy when using compressed data is comparable, if not less than when using original data, suggesting irrelevant noise is being removed by compression enabling efficient classification. However, results are dependent on the chosen classifier's architecture and the parameter reduction technique. Further work is required to understand the interplay between model architecture and data set quality when reducing model size for training lightweight machine learning models.

Chapter 4

Image Compression for Data Augmentation

4.1 Introduction

In this section, we provide empirical results highlighting the effectiveness of training on compressed data to increase robustness to common image corruptions. Compared to JPEG and other simple image transformations, we first show that training on BLS-compressed data improves classification accuracy on a diverse set of perceptual deformations. We then use the 2D Discrete Fourier Transform to investigate the differences in robustness afforded by different augmentations.

4.2 Methods

We transform images with BLS, JPEG, Gaussian noise and Gaussian blur to compare their benefits as data augmentation strategies. Adding independent Gaussian noise to each pixel introduces extraneous high frequency content to the image. Gaussian blur serves as a low-pass filter, removing high frequency signals from the image by convolving the images with a Gaussian function. Using CIFAR-10 training data, we replace the original images with those transformed using BLS, JPEG, Gaussian noise, and Gaussian blur. Details about the quality parameters for each augmentation strategy are presented in Table 4.1. These adjustable parameters allow us to control the extent to which training images are modified. Upon transformation, all pixel values are clipped to be between 0 and 1.

Per augmented data set, a VGG11 model was trained for three independent iterations and the model with the highest accuracy on CIFAR-10 test data was selected. This CNN was evaluated on test data from CIFAR-10C [16]. CIFAR-10C provides a standardized benchmark for evaluating a model’s robustness to common corruptions on CIFAR-10 images at varying levels of severity. The ten corruptions used in our experiments include speckle

Augmentation	Data Quality Parameters
BLS	$\lambda = [0.001, 0.003, 0.005, 0.012, 0.033, 0.05, 0.1]$
JPEG	quality = [1, 5, 10, 20, 40, 60, 80]
Gaussian noise	$\mu = 0$ and $\sigma = [1, 0.8, 0.3, 0.1, 0.05, 0.01]$
Gaussian blur	kernel size = [9x9, 7x7, 5x5, 3x3]

Table 4.1: Data Augmentations and Quality Parameters

noise (SN), contrast (C), impulse noise (IN), saturation (S), brightness (B), fog (F), frost (Fr), snow (Sn), Gaussian noise (GN) and Gaussian blur (GB) at five levels of severity.

4.3 Results

Since we experiment with four data augmentation strategies that each have variable amounts of parameters, we introduce three different quality parameter selection procedures. For each type of data augmentation, we do the following:

- **Best Data Augmentation (BDA)** - Select the data quality parameter that provides the maximum average test accuracy on all ten deformations at all levels of severity. This is the most selective yet resourceful method for selecting data since only one set of training data is used for all deformations and severity levels. Additionally, this quality parameter is more likely to generalize to new unseen corruptions.
- **Best Data Augmentation per Deformation (BDAD)** - For each of the ten CIFAR10-C image deformations, use the data quality parameter that maximizes average test accuracy across all levels of severity. Since each deformation introduces different defects in the images, this method provides the flexibility to select data quality parameters that maximize performance for each deformation. However, making use of this flexibility requires training on multiple data sets per augmentation strategy.
- **Best Data Augmentation per Deformation and Severity (BDADS)** - For each of the ten deformations and each severity level, select the data quality parameter that maximizes performance. This is the most flexible method yet potentially requires using the most training data.

In practice, the improvements afforded by BDAD and BDADS do not always outweigh the cost of using a large amount of training data, so the simpler and practical BDA is often preferred.

Augmentation	Ranked Data Quality Parameters
BLS	0.012, 0.1, 0.05, 0.033, 0.003, 0.001, 0.005
JPEG	80, 20, 60, 10, 40, 5, 1
Gaussian noise	0.01, 0.05, 0.1, 0.3, 1, 0.8
Gaussian blur	3x3, 5x5, 7x7, 9x9

Table 4.2: (BDA) Ranked Data Quality Parameters

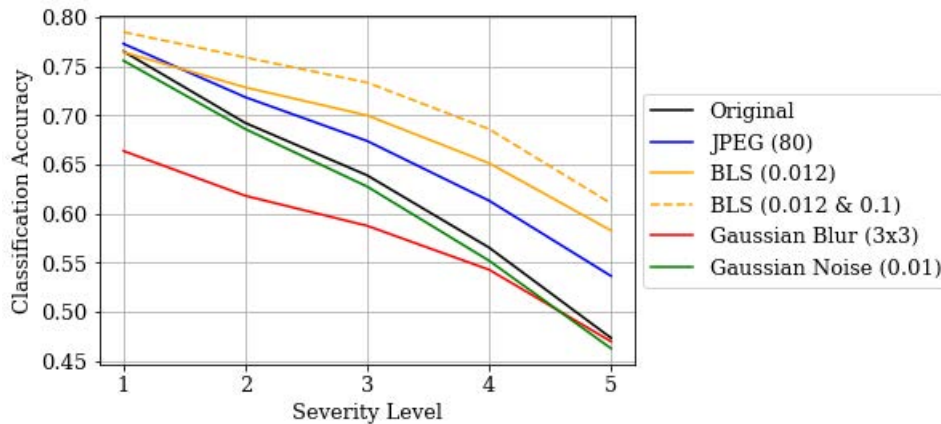


Figure 4.1: (BDA) Average Accuracy over all Image Deformations

BDA

In this section, we use the BDA approach to select data quality parameters for each augmentation strategy. Table 4.2 ranks the data quality parameters according to their effect on average accuracy over all deformations and quality levels. For Gaussian noise and blur augmentations, the parameters are predominantly ordered such that decreasing perceptual quality in training data corresponds to decreasing robustness to corruptions. However, the ordering for BLS and JPEG differs as both high and low quality data enable significant robustness gains. We train our CNN on these four data sets using the top-ranked parameter as well as the the original training data and show the average accuracy of these models across all deformations in Figure 4.1.

Besides the lowest severity level and particularly at higher severity levels, BLS outperforms all other augmentation strategies. JPEG also shows significant improvement over using original training data and the other augmentations. It is interesting to note that the optimal JPEG (quality = 80) and BLS ($\lambda = 0.012$) training data have similar average SSIM and PSNR values, suggesting that training data exhibiting certain amounts of distortion may bear robust models. We join the top-2 ranked BLS data sets ($\lambda = 0.012$ and 0.1) and

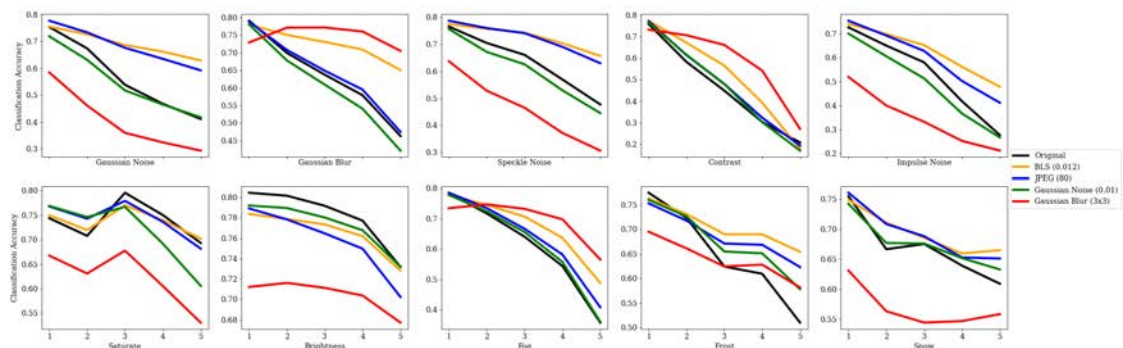


Figure 4.2: (BDA) Classifier Accuracy on all Image Deformations

train the CNN on this combined training data set (yellow dotted line). Though the size of the training data is two times larger, the addition of the higher quality ($\lambda = 0.1$) compressed data improves the average accuracy across all deformations and all severity levels.

Since we are averaging results over all types of deformations, the optimal Gaussian noise and blur augmentation strategies are not very promising as their effectiveness varies across deformations as seen in Figure 4.2. For example, Gaussian blur clearly outperforms other strategies on certain deformations (Gaussian blur, contrast, and fog) but suffers substantially on others. We can partially account for such discrepancies by selecting the optimal parameter per deformation using BDAD or BDADS.

As noted in [23], it is valuable to design data augmentation procedures that yield models with high accuracy on both clean data and corrupt data. In Figure 4.3, we show how different augmentations compare in terms of their clean and corrupt accuracy. Clean accuracy is calculated by evaluating the models on original CIFAR-10 test data and corrupt accuracy is calculated by averaging the accuracy of models across all deformation and severity levels. As expected, training on original data produces a model with the highest clean accuracy. Despite having the highest corrupt accuracy, BLS ($\lambda = 0.012$) has a lower clean accuracy compared to the other strategies. However, training on the top-2 BLS compressed data sets ($\lambda = 0.012$ and 0.1), boosts both corrupt and clean accuracy. Training on additional higher quality data ($\lambda = 0.1$) allows the model to retain pertinent visual details to correctly classify clean data while improving on the robustness afforded by the lower quality compressed data ($\lambda = 0.012$). BLS-compressed training data does not merely restrict the model to learning invariance to distortions introduced by compression. It remains unclear how the combination of the two data sets may influence the representations learned by the model, but training on multiple levels of compressed data seems advantageous and is revisited in Chapter 6.

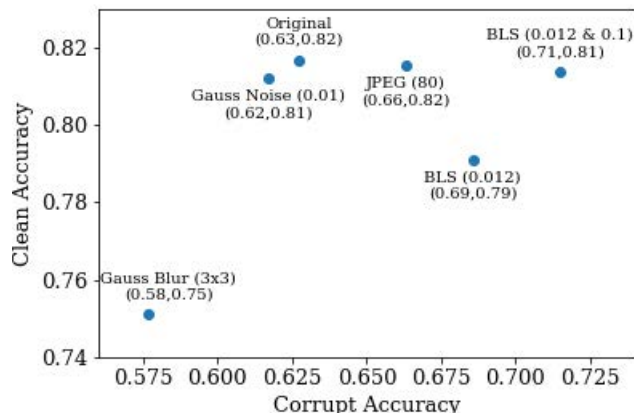


Figure 4.3: (BDA) Comparing Model Accuracy on Corrupt and Clean Data

Augmentation	GN	GB	SN	C	IN	S	B	F	Fr	Sn
BLS	0.05	0.1	0.012	0.1	0.05	0.033	0.033	0.012	0.1	0.033
JPEG	20	5	20	60	20	80	80	80	80	80
Gaussian noise	0.05	0.01	0.05	0.01	0.1	0.01	0.01	0.01	0.01	0.01
Gaussian blur	3x3	3x3	3x3	3x3	3x3	3x3	3x3	3x3	3x3	3x3

Table 4.3: (BDAD) Optimal Data Quality Parameters

BDAD

As seen in the previous section, training our classification models on the same data may be too brittle when evaluating our model on different image deformations. In this section, we use the BDAD approach to select quality parameters. Table 4.3 specifies the optimal quality parameter per augmentation and deformation. To train a model with a certain augmentation strategy for d different deformations, BDAD requires n training samples at the minimum and $d * n$ training samples at the maximum where n is the number of training images. Certain augmentations such as Gaussian noise and blur show little to no variability in parameters across all deformations compared to BLS and JPEG. The broader range of parameters selected for BLS and JPEG may indicate that compressed data can be better tailored to maximize robustness for specific deformations. However, BLS and JPEG do not always use similar levels of compressed data for the same deformations as seen by the optimal parameters used for the Gaussian blur deformation. This reinforces the notion that different information is retained by these two compression methods which contributes to differences in model accuracy and robustness.

In Figure 4.4 we show the average classification accuracy across all deformations using

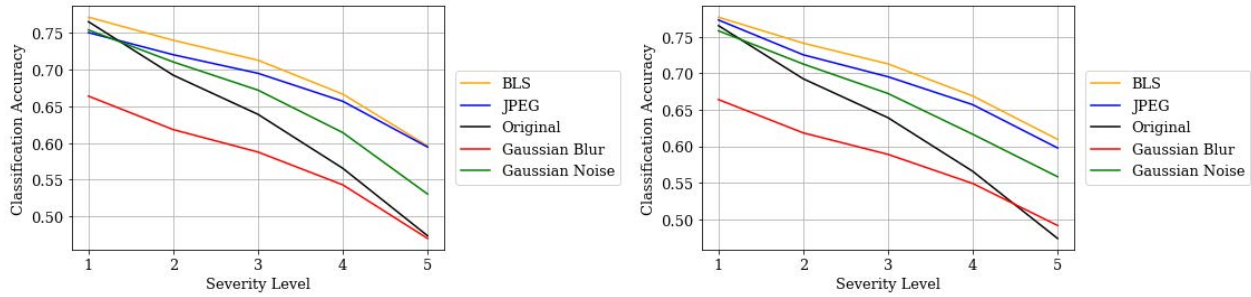


Figure 4.4: (BDAD) Average Accuracy over all Image Deformations

Figure 4.5: (BDADS) Average Accuracy over all Image Deformations

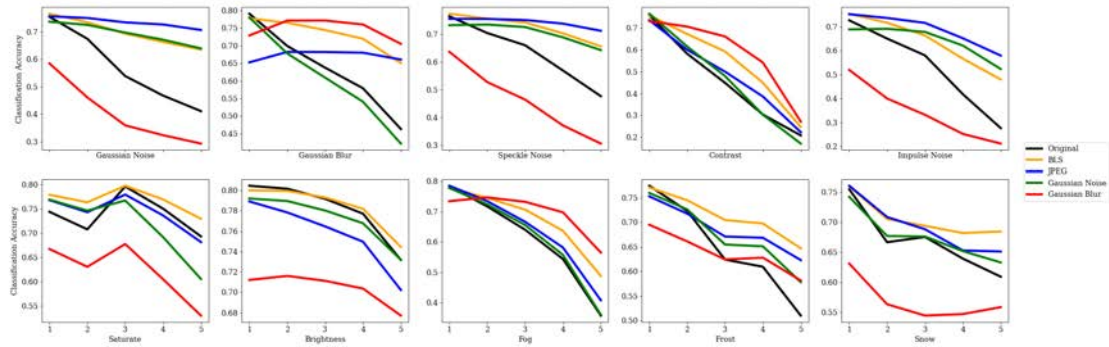


Figure 4.6: (BDAD) Accuracy on All Image Deformations

BDAD parameter selection. Most notably, JPEG closes the performance gap with BLS at higher severity levels and Gaussian noise proves to be more effective than using original data.

The accuracy of classifiers per deformation is shown in Figure 4.6. Compared to BDA, using BDAD for parameter selection increases robustness for a select few deformations and augmentations, rather than a uniform gain in robustness across all corruptions. Though the optimal data quality parameter may depend on the type of image deformation, one should consider the amount of training data required and the difficulty in stratifying test data by corruption to apply different trained classifiers when deciding between BDA and BDAD.

BDADS

In Figure 4.5, we use BDADS and show the average accuracy across all deformations. Selecting parameters optimized over both quality level and type of deformation produces results very similar to those achieved using BDAD.

Frequency Analysis of Data Augmentations

In this section, we present a preliminary attempt in understanding why certain augmentations are more effective for training robust models by inspecting the Fourier spectrum of the transformed images in the augmented data sets. For each augmentation strategy, we select the optimal data quality parameter using BDA and compute the 2D Discrete Fourier Transform (DFT) on each of the modified images. In Figure 4.7, we display the average normalized log magnitude of the DFT output over all images for each augmentation. The DC component is displayed in the center of each image. All displayed magnitude spectrums are similar to empirically observed spectral signatures of natural objects [42]. Gaussian noise and JPEG produce spectrums very similar to the original data where the contribution of higher frequency components quickly diminishes. Despite preserving more low and medium frequency signals, Gaussian blur serves its purpose as a low pass filter and removes most high frequency information as well. The higher frequency signals do not exhibit the same falloff for BLS-compressed images. Compared to original data and the other image transformations, BLS uniformly preserves a wider range of frequencies, providing classifiers with images containing richer spectral signatures that enable correct classification despite corruptions.

As mentioned previously, the most effective data augmentation strategies do not simply remove all high frequency information from the images as some of this content may be useful. Unlike JPEG and Gaussian blur which are primarily intended to remove high frequency signals indiscriminately, the objective of BLS is to preserve perceptual information in the image during compression. As a result, BLS may conserve a wider range of frequencies to preserve compressed images' perceptual acuity.

Nevertheless, how the differences in augmentations affect model training and robustness remains an open question. Frequency analysis of transformed images offers one useful perspective but other approaches mentioned in Chapter 6 may also be beneficial.

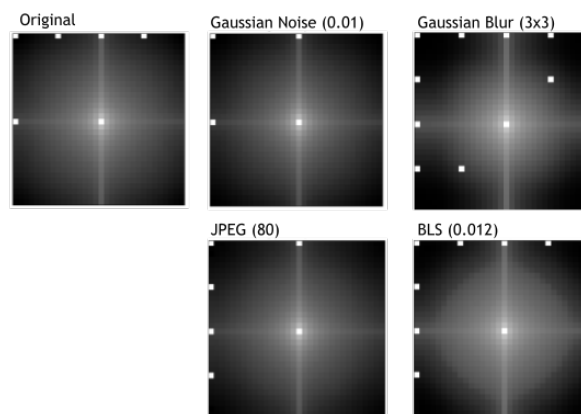


Figure 4.7: Fourier Magnitude Spectrum of Augmented Data

Chapter 5

Conclusion

In this report, we describe the process of using learned image compression to preprocess data for image classification models and explore its effect on model accuracy and robustness. In Chapter 3, we show that learned compression enables higher fidelity image compression at lower bit rates compared to JPEG, resulting in classifiers that maintain test accuracy with less training data. Image quality as quantified by PSNR and SSIM is positively correlated with classification accuracy yet the differences in classification-relevant content preserved by JPEG and learned image compression remain unclear and should be further explored. Finally, reducing the number of trainable parameters and training on compressed data results in similar or better performing classifiers depending on the classifier architecture.

In Chapter 4, we show that training on BLS-compressed data offers robustness to a diverse set of image perturbations while preserving accuracy on clean data. This indicates that effectively removing irrelevant content from images may provide a cleaner training signal for the classifier to learn from. Furthermore, training with multiple levels of compressed data improves accuracy on both corrupt and clean data. Finally, Fourier analysis of compressed images indicates that learned compression better preserves high frequency information which may be useful for robust classification.

The empirical findings summarized above would benefit from explanations grounded in deep learning, data compression and signal processing theory. Potential theoretical explorations include quantifying the content being preserved and removed by the learned image compression model, using the information bottleneck theory [36] to compare compressed representations learned during classification with compressed representations learned outside the context of classification, and comparing the theoretical limitations in information preservable by linear compression models and learned image compression. Although many questions remain unanswered and theoretical explanations are needed, we hope our findings motivate further exploration and the use of compressed data for training deep learning models.

Chapter 6

Future Work

The results presented in this report are a preliminary attempt in exploring the intersection of learned data compression and supervised machine learning yet many questions remain unanswered or waiting to be asked.

Though results in Chapter 3 and 4 suggest several benefits for training with compressed data, our experiments should be performed on different CNN architectures and image datasets. This will help ascertain whether gains in robustness, model compression, and generalization are actually attributable to compressed data. It will also be useful to perform similar experiments for other supervised learning tasks such as object detection or image segmentation to more broadly understand the effect of data compression on supervised learning.

6.1 Improving Learned Compression Models for Classification

Although learned image compression models such as BLS and BM3D have shown tremendous success in preserving visual acuity of compressed images at low bit rates, it is worthwhile exploring whether and to what extent semantic information can be explicitly conserved during compression by modifying the loss functions used by the compression models. Training compression models using MS-SSIM [45], contrastive learning [14] or combinations of classification and reconstruction losses will better preserve images' structural content, potentially improving performance on downstream learning tasks. Using computer vision algorithms such as Histogram of Oriented Gradients (HOG), we can detect and compress background pixels, leaving the object in the foreground as is. This would prevent the image classifier from 'cheating' by using background features for classification and allow image compression without degrading the quality of the object. By compressing images using the methods outlined above, we can uncover differences between images compressed using different objectives.

6.2 Interpreting the Benefits of Learned Compression

Although learned compression enables higher accuracy and robustness in downstream classifiers compared to JPEG and other baselines, the beneficial features of compressed images remain unknown. In Section 4.3, the average magnitude spectrum of BLS-compressed images does not shed light on what is learned by the classifier from this data. Using the approach from [46], noise aligned with different Fourier basis vectors can be added to the images and the trained classifier’s sensitivity to these perturbations at different layers can be recorded to probe the model’s robustness to varying frequencies. As an alternative to the global image statistics provided by the 2D DFT, a histogram of Gabor filter responses may highlight the differences between augmentation strategies. Finally, saliency maps in image space can be constructed using backpropagation on trained classifiers [37] to visually compare what regions of compressed and original images are used for classification.

6.3 Alternative Training Paradigms with Compressed Data

As observed in Section 4.3, training on different levels of compressed data can boost robustness to different corruptions and increase accuracy on clean data. Data compressed to various extremities may be useful to varying extents so they must be used accordingly during training. One potential method would be to create a training data set containing different levels of compressed images and weight the contribution of a given image for training such that highly compressed images affect model training less.

As images are compressed, they are more easily confused as belonging to an incorrect class. By collecting human perceptual judgements on compressed training images, we can create a probability distribution over classes for each image. During training, we compare the classifier’s softmax output with the human labels which reflect the uncertainty in judging compressed images (Figure 6.1). This procedure was used with original CIFAR-10 data and showed improvements in classifiers’ generalization and robustness [30].

Contrastive learning is used for learning embeddings of high-dimensional images by pulling together similar images and pushing apart different images in a lower-dimensional embedding space [14]. Varying image augmentations such as cropping, rotation, and Gaussian blur have been used with contrastive learning to learn invariant representations [5].

Since relevant information for classification may be preserved across different levels of compression for a given image, learning a single representation that must incorporate this diverse information can be useful. As shown in Figure 6.2, a pair of weight-sharing encoders are jointly trained using a contrastive loss that pulls the same image compressed to different extents and pushes apart different images. Once trained, the encoder’s output can be used to train a DNN classifier.

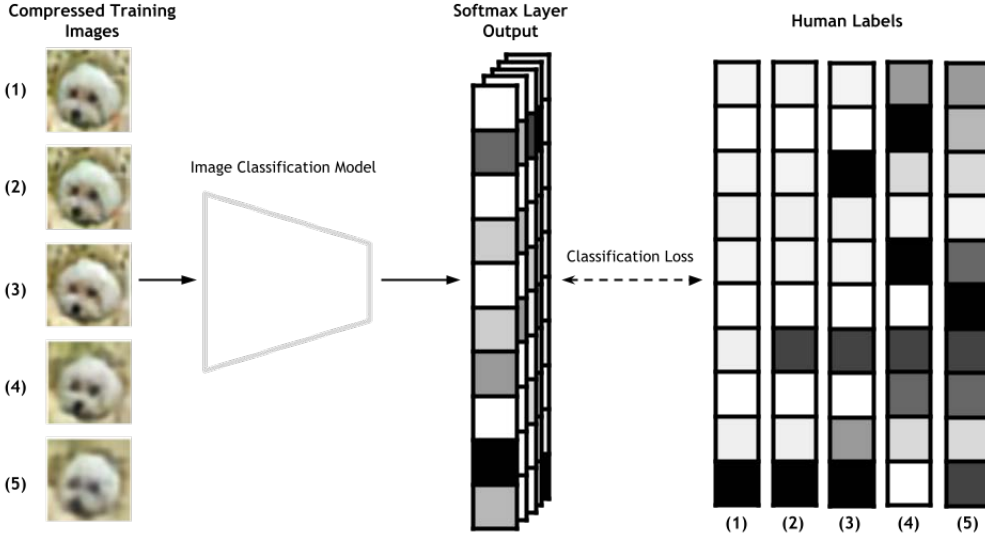


Figure 6.1: Training with Compressed Images and Human Labels

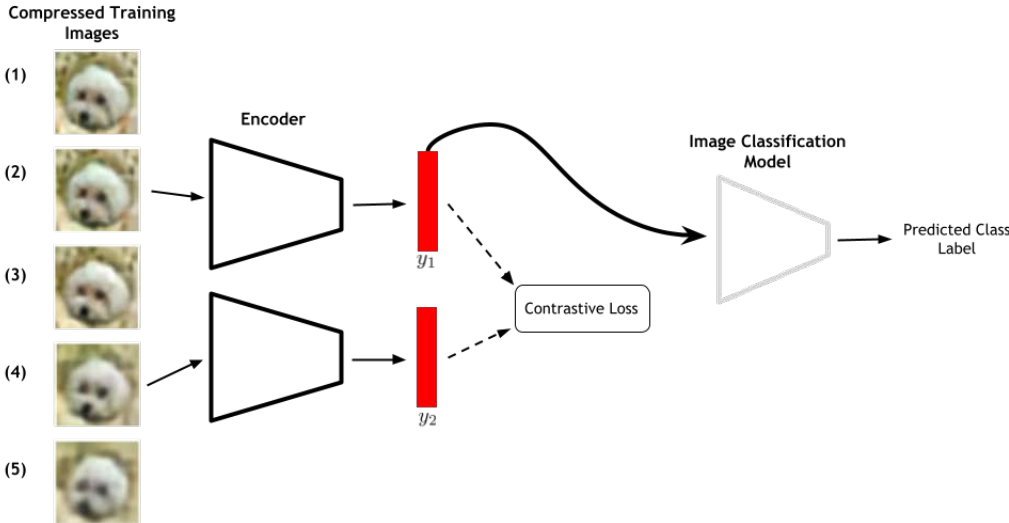


Figure 6.2: Contrastive Training with Compressed Images

Bibliography

- [1] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. “End-to-end Optimized Image Compression”. In: *Proceedings of the 5th International Conference on Learning Representations* (2016).
- [2] Johannes Ballé et al. *Tensorflow Compression*. 2017. URL: www.github.com/tensorflow/compression.
- [3] Johannes Ballé et al. *Variational image compression with a scale hyperprior*. 2018. arXiv: 1802.01436 [eess.IV].
- [4] Tom B. Brown et al. “Language Models are Few-Shot Learners”. In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901.
- [5] Ting Chen et al. “A Simple Framework for Contrastive Learning of Visual Representations”. In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, 13–18 Jul 2020, pp. 1597–1607.
- [6] Ekin D. Cubuk et al. *AutoAugment: Learning Augmentation Policies from Data*. 2019. arXiv: 1805.09501 [cs.CV].
- [7] Nilaksh Das et al. *Keeping the Bad Guys Out: Protecting and Vaccinating Deep Learning with JPEG Compression*. 2017. arXiv: 1705.02900 [cs.CV].
- [8] Samuel Dodge and Lina Karam. “Understanding how image quality affects deep neural networks”. In: *2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX)*. 2016, pp. 1–6.
- [9] Gintare Karolina Dziugaite, Zoubin Ghahramani, and Daniel M. Roy. *A study of the effect of JPG compression on adversarial images*. 2016. arXiv: 1608.00853 [cs.CV].
- [10] Yang Fan et al. “Learning What Data to Learn”. In: *CoRR* abs/1702.08635 (2017). arXiv: 1702.08635.
- [11] Gerald Friedland et al. “On the Impact of Perceptual Compression on Deep Learning”. In: *2020 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*. 2020, pp. 219–224.

- [12] Robert Geirhos et al. “Generalisation in humans and deep neural networks”. In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc., 2018.
- [13] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. “Explaining and Harnessing Adversarial Examples”. In: *International Conference on Learning Representations*. 2015.
- [14] Raia Hadsell, Sumit Chopra, and Yann LeCun. “Dimensionality Reduction by Learning an Invariant Mapping”. In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06)*. Vol. 2. 2006, pp. 1735–1742. DOI: 10.1109/CVPR.2006.100.
- [15] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90.
- [16] Dan Hendrycks and Thomas Dietterich. “Benchmarking Neural Network Robustness to Common Corruptions and Perturbations”. In: *Proceedings of the International Conference on Learning Representations (2019)*.
- [17] Md Tahmid Hossain et al. *Robust Image Classification Using A Low-Pass Activation Function and DCT Augmentation*. 2020. arXiv: 2007.09453 [cs.CV].
- [18] Ajay J. Joshi, Fatih Porikli, and Nikolaos Papanikolopoulos. “Multi-class active learning for image classification”. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 2372–2379. DOI: 10.1109/CVPR.2009.5206627.
- [19] Asifullah Khan et al. “A Survey of the Recent Architectures of Deep Convolutional Neural Networks”. In: *CoRR* abs/1901.06032 (2019). arXiv: 1901.06032.
- [20] Friso H. Kingma, Pieter Abbeel, and Jonathan Ho. “Bit-Swap: Recursive Bits-Back Coding for Lossless Compression with Hierarchical Latent Variables”. In: *Proceedings of the 36th International Conference on Machine Learning*. Vol. 97. Proceedings of Machine Learning Research. PMLR, Sept. 2019, pp. 3408–3417.
- [21] Alex Krizhevsky. *Learning multiple layers of features from tiny images*. Tech. rep. 2009.
- [22] Shuying Liu and Weihong Deng. “Very deep convolutional neural network based image classification using small training sample size”. In: *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*. 2015, pp. 730–734. DOI: 10.1109/ACPR.2015.7486599.
- [23] Raphael G. Lopes et al. *Improving Robustness Without Sacrificing Accuracy with Patch Gaussian Augmentation*. 2019. arXiv: 1906.02611 [cs.LG].
- [24] Gouenou Coatrieux Maxime Pistono, Jean-Claude Nunes, and Michel Cozic. “Training Machine Learning on JPEG Compressed Images”. In: *2020 Data Compression Conference (DCC)*. 2020, pp. 388–388.

- [25] Fabian Mentzer et al. “Conditional Probability Models for Deep Image Compression”. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4394–4402.
- [26] Fabian Mentzer et al. *High-Fidelity Generative Image Compression*. 2020. arXiv: 2006.09965 [eess.IV].
- [27] Mingkun Li and I. K. Sethi. “Confidence-based active learning”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.8 (2006), pp. 1251–1261. DOI: 10.1109/TPAMI.2006.156.
- [28] Dylan M. Paiton et al. “Selectivity and robustness of sparse coding networks”. In: *Journal of Vision* 20.12 (Nov. 2020), pp. 10–10.
- [29] Yanting Pei et al. “Effects of Image Degradation and Degradation Removal to CNN-Based Image Classification”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.4 (2021), pp. 1239–1253. DOI: 10.1109/TPAMI.2019.2950923.
- [30] Joshua C. Peterson et al. “Human Uncertainty Makes Classification More Robust”. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 9616–9625.
- [31] Prasun Roy et al. “Effects of Degradations on Deep Neural Network Architectures”. In: (2018). arXiv: 1807.10108 [cs.CV].
- [32] Evgenia Rusak et al. *A simple way to make neural networks robust against diverse image corruptions*. 2020. arXiv: 2001.06057 [cs.CV].
- [33] Olga Russakovsky et al. “ImageNet Large Scale Visual Recognition Challenge”. In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252. DOI: 10.1007/s11263-015-0816-y.
- [34] Burr Settles. *Active Learning Literature Survey*. Computer Sciences Technical Report 1648. University of Wisconsin–Madison, 2009.
- [35] Connor Shorten and Taghi M. Khoshgofaar. “A survey on Image Data Augmentation for Deep Learning”. In: *Journal of Big Data* 6.60 (2019).
- [36] Ravid Shwartz-Ziv and Naftali Tishby. “Opening the Black Box of Deep Neural Networks via Information”. In: *CoRR* abs/1703.00810 (2017). arXiv: 1703.00810.
- [37] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. *Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps*. 2014. arXiv: 1312.6034 [cs.CV].
- [38] Saurabh Singh et al. *End-to-end Learning of Compressible Features*. 2020. arXiv: 2007.11797 [cs.CV].
- [39] Nicola Strisciuglio, Manuel Lopez-Antequera, and Nicolai Petkov. *A Push-Pull Layer Improves Robustness of Convolutional Neural Networks*. 2019. arXiv: 1901.10208 [cs.CV].

- [40] Christian Szegedy et al. *Intriguing properties of neural networks*. 2014. arXiv: 1312.6199 [cs.CV].
- [41] Mariya Toneva et al. “An Empirical Study of Example Forgetting during Deep Neural Network Learning”. In: *ICLR*. 2019.
- [42] Antonio Torralba and Aude Oliva. “Statistics of natural image categories”. In: *Network: Computation in Neural Systems* 14.3 (2003), pp. 391–412.
- [43] Igor Vasiljevic, Ayan Chakrabarti, and Gregory Shakhnarovich. *Examining the Impact of Blur on Recognition by Convolutional Networks*. 2017. arXiv: 1611.05760 [cs.CV].
- [44] Zhou Wang and Alan C. Bovik. “Mean squared error: Love it or leave it? A new look at Signal Fidelity Measures”. In: *IEEE Signal Processing Magazine* 26.1 (2009), pp. 98–117.
- [45] Zhou Wang, Eero P. Simoncelli, and Alan C. Bovik. “Multiscale structural similarity for image quality assessment”. In: *The Thrity-Seventh Asilomar Conference on Signals, Systems Computers, 2003*. Vol. 2. 2003, pp. 1398–1402.
- [46] Dong Yin et al. “A Fourier Perspective on Model Robustness in Computer Vision”. In: *Neural Information Processing Systems (NeurIPS)*. 2019.
- [47] Jinsung Yoon, Serkan O. Arik, and Tomas Pfister. “Data Valuation using Reinforcement Learning”. In: *Proceedings of the 37th International Conference on Machine Learning*. Vol. 119. Proceedings of Machine Learning Research. PMLR, 13–18 Jul 2020, pp. 10842–10851.
- [48] Farhad G. Zanjani et al. “Impact of JPEG 2000 compression on deep convolutional neural networks for metastatic cancer detection in histopathological images”. In: *Journal of Medical Imaging* 6.2 (2019), pp. 1–9.
- [49] Zhou Wang et al. “Image quality assessment: from error visibility to structural similarity”. In: *IEEE Transactions on Image Processing* 13.4 (2004), pp. 600–612. DOI: 10.1109/TIP.2003.819861.