8-12-2023

# Understanding Image Quality for Deep Learning-Based Computer Vision

Austin Bergstrom
acb6595@rit.edu

# Understanding Image Quality for Deep Learning-Based Computer Vision

by

Austin Bergstrom

A dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy
in the Chester F. Carlson Center for Imaging Science
College of Science
Rochester Institute of Technology

August 12, 2023

Signature of the Author _____

Accepted by _____
Coordinator, Ph.D. Degree Program Date

CHESTER F. CARLSON CENTER FOR IMAGING SCIENCE
COLLEGE OF SCIENCE
ROCHESTER INSTITUTE OF TECHNOLOGY
ROCHESTER, NEW YORK

CERTIFICATE OF APPROVAL

---

Ph.D. DEGREE DISSERTATION

---

The Ph.D. Degree Dissertation of Austin Bergstrom
has been examined and approved by the
dissertation committee as satisfactory for the
dissertation required for the
Ph.D. degree in Imaging Science

Dr. David Messinger, Dissertation Advisor

Dr. George Thurston, External Chair

Dr. Carl Salvaggio

Dr. Michael Gartley

Date

# Understanding Image Quality for Deep Learning-Based Computer Vision

by

Austin Bergstrom

Submitted to the
Chester F. Carlson Center for Imaging Science
in partial fulfillment of the requirements
for the Doctor of Philosophy Degree
at the Rochester Institute of Technology

## Abstract

Extensive research has gone into optimizing convolutional neural network (CNN) architectures for tasks such as image classification and object detection, but research to date on the relationship between input image quality and CNN prediction performance has been relatively limited. Additionally, while CNN generalization against out-of-distribution image distortions persists as a significant challenge and a focus of substantial research, a range of studies have suggested that CNNs can be be made robust to low visual quality images when the distortions are predictable. In this research, we systematically study the relationships between image quality and CNN performance on image classification and detection tasks. We find that while generalization remains a significant challenge for CNNs faced with out-of-distribution image distortions, CNN performance against low visual quality images remains strong with appropriate training, indicating the potential to expand the design trade space for sensors providing data to computer vision systems. We find that the functional form of the GIQE can predict CNN performance as a function of image degradation, but we observe that the legacy form of the GIQE does a better job of modeling the impact of blur/relative edge response in some scenarios. Additionally, we evaluate other image quality models that lack the pedigree of the GIQE and find that they generally work as well or better than the functional form of the GIQE in modeling computer vision performance on distorted images. We observe that object detector performance is qualitatively very similar to image classifier performance in the presence of image distortion. Finally, we observe that computer vision performance tends to exhibit relatively smooth, monotonic variation with blur and noise, but we find that performance is relatively insensitive to resolution under a range of conditions.

## Disclaimer

The views expressed in this dissertation are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U.S. Government.

## Acknowledgements

My advisor Dr. David Messinger was an immensely helpful guide throughout this research, particularly in helping to prioritize the questions to answer first and in deciding which unkonwns to leave unknown. I am also grateful to the other members of my committee– Dr. Michael Gartley, Dr. Carl Salvaggio, and Dr. George Thurston–for their helpful comments and questions. Much of what I found (and many of the mistakes in my work that I caught) came as a result of thinking about and working through their helpful questions and suggestions.

David Conran offered significant assistance in several ways. Materially, he provided some very helpful code that I was able to use in studying the extent to which optical point spread functions and their Gaussian approximations yield similar or different relative edge response values in image simulations. More importantly, our many conversations were terrific fodder throughout my research.

I am grateful for the Center for Imaging Science faculty who helped lay the foundation for understanding the image chain from end to end. And while all of the courses I took proved helpful, I would be especially remiss without highlighting the contributions of Dr. Roger Easton. The world looks different after his Fourier course; I am genuinely grateful to see and understand the beauty of the linear shift invariant system.

I am also grateful and indebted to Mr. Bruce Murdock, high school physics teacher for the ages. Almost 20 years after taking his class, his explanations of kinematics and Newton's laws are still fresh. His influence is hard to estimate.

Finally, I am grateful most of all to my wife Holly for her love and the sacrifices that she made as I pursued this degree. Without a doubt, she gave up more than I did to make this degree a reality. She is a treasure.

*To Holly, Jack, Nate, Lucy and Reese.*

*And Mr. Murdock.*

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Understanding of image quality for computer vision applications lags understanding of image quality for human vision. Designers of cameras and remote sensing systems have historically sought to optimize image quality for the human visual system, because historically remote sensing imagery (panchromatic and / or true color) has been analysed visually. Research on the problem of designing cameras and remote sensing systems for the human visual system dates back to the earliest use of telescopes and accelerated in the late $20^{\text{th}}$ and early $21^{\text{st}}$ centuries with the proliferation of electro-optical imaging [1, 2, 3, 4, 5, 6, 7, 8]. More recently, at the back end of the image chain (*i.e.*, processing of the imagery after collection), significant research has explored the problem of optimizing convolutional neural networks (CNNs) for classifying images and performing tasks such as object detection and image segmentation, particularly since 2012 when Krizhevsky *et al.* decisively demonstrated the capability of CNNs for image classification [9]. And while these CNNs have found widespread use, and despite their strong performance against relatively high quality images, generalization remains a significant challenge, and CNNs struggle with images of objects in unusual contexts or viewed from sub-optimal geometry [10, 11, 12].

Largely used in the remote sensing field for images to be analyzed manually, the National Image Interpretability Rating Scale (NIIRS) uses the General Image Quality Equation (GIQE) shown in equation 1.1 to map three fundamental sensing parameters–ground sample distance (GSD), relative edge response (RER), and signal to noise ratio (SNR)–to a numerical score which predicts the types of objects an analyst can identify in an image, as shown in Table 1.2 [13]. The GIQE is non-linear, with an independent term for each variable and a cross-term intended to capture the impact of coupling between RER and

SNR, taking the form

$$\text{NIIRS} = A_0 + A_1 \log_{10}(\text{GSD}) + A_2 \left(1 - \exp\frac{A_3}{\text{SNR}}\right) \log_{10} \text{RER} + A_4 (\log_{10} \text{RER})^4 + \frac{A_5}{\text{SNR}}, \quad (1.1)$$

with the coefficients $\{A_i\}$ taking on the values in Table 1.1 [14]. To build a sensor that will collect images for human interpretation, designers work within a well defined trade space.

Table 1.1: GIQE version 5 coefficients relating imaging system parameters to image NIIRS scores

| $A_0$ | $A_1$ | $A_2$ | $A_3$ | $A_4$ | $A_5$ |
|---|---|---|---|---|---|
| 9.57 | -3.32 | 3.32 | -1.9 | -2 | -1.8 |

Table 1.2: Example analysis tasks possible at each NIIRS level

| **NIIRS Rating** | **Example analysis tasks enabled** |
|---|---|
| 0 | Interpretability of the imagery is precluded by obscuration, degradation, or very poor resolution. |
| 1 | Detect a medium-sized port facility andyor distinguish be-tween taxiways and runways at a large airfield. |
| 2 | Detect large hangars at airfields. Detect large buildings e.g., hospitals, factories |
| 3 | Detect the presence/absence of support vehicles at a mobile missile base. |
| 4 | Identify individual tracks, rail pairs, control towers, switch-ing points in rail yards. |
| 5 | Identify individual rail cars by type (e.g., gondola, flat, box) and/or locomotive by type (e.g., steam, diesel). |
| 6 | Identify automobiles as sedans or station wagons. |
| 7 | Identify ports, ladders, vents on electronics vans. |
| 8 | Identify windshield wipers on a vehicle. |
| 9 | Differentiate cross-slot from single slot heads on aircraft skin panel fasteners. |

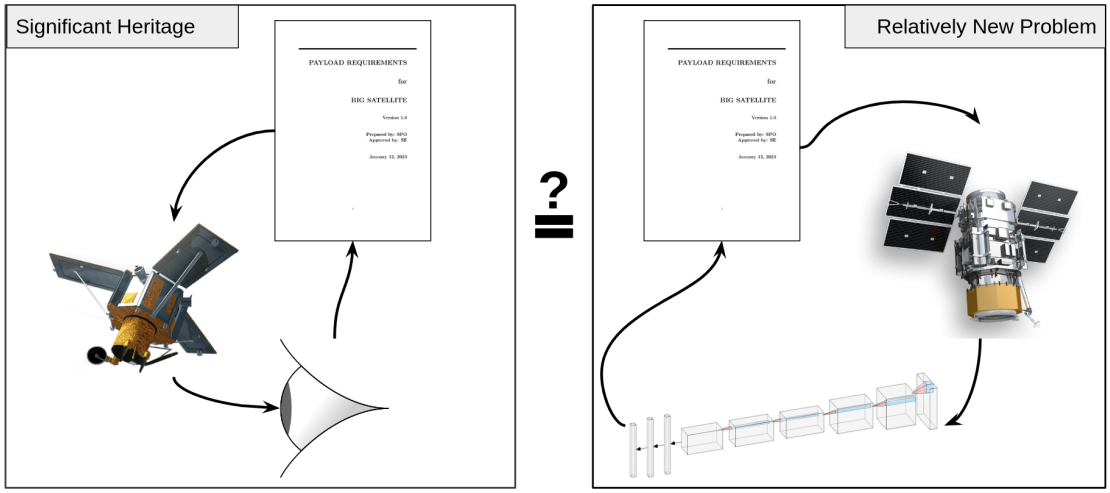



Figure 1.1: Our approach to designing and developing imaging systems evolved with the primary goal of collecting images for human viewing and analysis. Here, we begin to systematically study image quality for computer vision with the goal of understanding the extent to which traditional imaging system requirements will or will not transfer to systems that primarily provide data for machine analysis.

At the intersection of imaging and computer vision lies the challenge of understanding image quality as it relates to the performance of computer vision tools, particularly CNNs. For many computer vision tasks, CNNs will continue to analyze images captured primarily for human use; for these tasks, optimizing the CNN with respect to its input data is the pertinent problem. For other tasks, however, such as those relating to autonomy and remote sensing, CNNs will analyze images captured primarily for machine use. In the latter case, understanding image quality as it relates to CNN performance represents a significant question. As computer vision plays more and more central roles in analyzing image data, it becomes increasingly important to understand the extent to which image quality for computer vision mirrors or diverges from image quality for human vision (Fig. 1.1).

Images in remote sensing and autonomous operations have the propensity to stress the CNN weaknesses identified in the literature [10, 11, 12]. Relative to ImageNet [15] data, overhead imagery has lower resolution, often contains more noise, and (relative to everyday ImageNet images) is collected from atypical viewing geometry. Similarly, images collected by autonomous vehicle cameras will at times experience degradations such as blur and poor illumination. Given the growing range of applications in which computer vision

systems will be required to analyze lower visual quality images from sensors that may provide few or no images to human viewers, it is important to understand the relationship between image quality factors and CNN performance, particularly if those factors differ between human and computer interpreters. This understanding could lead to sensors that collect data better tailored to machine interpretation and could open up new trade space for imaging system designs, even if the collected imagery are of "low" visual quality.

Additionally, we highlight that in many computer vision tasks, scene content and clutter will vary from scene to scene and particularly from dataset to dataset. These types of variations tend to stress the *generalization* capabilities of CNNs discussed in 2.4, representing an axis in the problem of computer vision that we believe to be largely *orthogonal* to that of image quality. Both axes warrant attention, and generalization is arguably the primary focus of computer vision research today; we intend to address the question of image quality in this effort.

# Chapter 2

# Background

## 2.1 Image Quality

Historically, we can identify three main branches of image quality study. First, in the imaging hardware regime, the primary task is to minimize the differences between image and scene; the field of optics largely centers on how to create a tight point spread function (PSF) and maximizes a system's optical transfer function (OTF), and the field of sensor design largely centers on maximizing signal and minimizing noise [8, 16, 13, 17]. Second, in the image processing regime, the primary task is to minimize the difference between an initial image and a final compressed image as quantified by any number of functions [18, 6, 19], where often the functions are driven by empirical research on human visual perception. Finally, in the regime of remote sensing systems, the primary task is to maximize some utility metric (whether quantitative or qualitative) defined by the task at hand. To consider how these image quality branches are interrelated, we will consider some of the basic relationship affecting optical designs and how these relationships ultimately drive system designs and trades.

### 2.1.1 Linear Shift-Invariant Systems

To understand image quality, it is helpful to begin by approximating optical systems as linear and shift invariant (LSI), allowing us to use the tool set developed for the study of *LSI systems*. As encapsulated by Easton [20], a linear system is one whose output consists of a weighted sum of its inputs. Formally, in one dimension, we can treat such as system as an operator $\Omega$ acting on functions $f_1$ and $f_2$ that satisfies

$$\Omega\left\{\alpha f_1\left(x\right)+\beta f_2\left(x\right)\right\}=\alpha\Omega\left\{f_1\left(x\right)\right\}+\beta\Omega\left\{f_2\left(x\right)\right\}, \tag{2.1}$$

where $\alpha$ and $\beta$ are (possibly complex) constants. A shift invariant system is one whose action on an input is independent of the input's position. Formally, in one dimension, we can treat such a system as an operator $\Omega$ operating on $f_1$ such that if

$$\Omega\left\{f\left(x\right)\right\} = g\left(x\right), \tag{2.2}$$

then

$$\Omega\left\{f\left(x - x_0\right)\right\} = g\left(x - x_0\right) \tag{2.3}$$

for all $x_0$.

Two factors make the linear shift-invariant (LSI) system a particularly useful approximation. First, the eigenfunctions of any linear shift invariant system are complex exponentials of form $f\left(x\right) = \alpha e^{ikx}$. In other words, an LSI-system with a complex exponential input will *always* output a scaled version of this same complex exponential. Importantly, these complex exponentials form an ortho-normal set of basis functions into which any function defined over the real number line $\mathbb{R}$ can be decomposed. We can therefore rewrite the input to any LSI system as an infinite superposition of complex exponentials, and the output will be a superposition of the same complex exponentials scaled by the *transfer function* of the system. Using the common notation of Fourier analysis, we can express any function $f\left(x\right)$ as the sum of its complex exponential Fourier components, or

$$f\left(x\right) = \int_{-\infty}^{\infty} F\left(\xi\right) e^{i2\pi\xi x} d\xi, \tag{2.4}$$

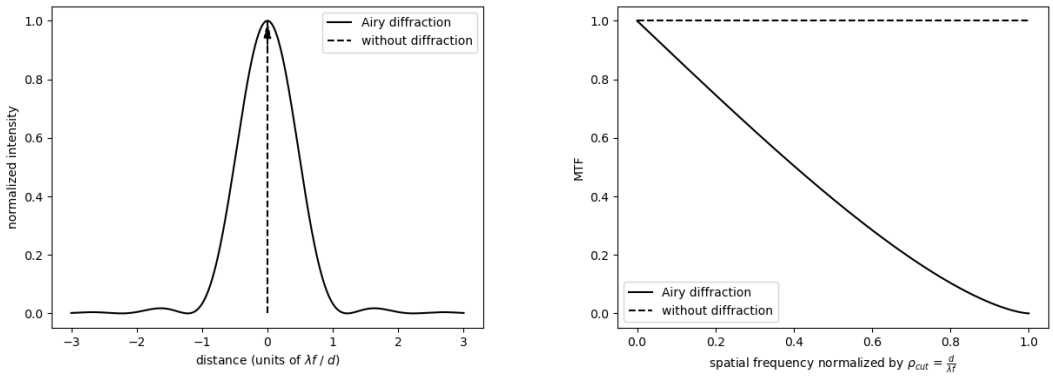
where $F\left(\xi\right)$ is the superposition weighting of the complex exponential of frequency $\xi$. The function $F\left(\xi\right)$ is generally known as the Fourier representation of $f\left(x\right)$, and it is calculated by taking the projection of $f\left(x\right)$ onto the set of complex exponentials, which we compute with the Fourier transform,

$$F\left(\xi\right) = \int_{-\infty}^{\infty} f\left(x\right) e^{-i2\pi\xi x} dx \equiv \mathfrak{F}\left\{f\left(x\right)\right\}, \tag{2.5}$$

denoted here as the operator $\mathfrak{F}$.

Second, any LSI system can be fully described with a convolution operation. Specifically, the output $g\left(x\right)$ of any LSI system is the convolution of the input $f\left(x\right)$ with the system's impulse response function $h\left(x\right)$ (known as point spread function for optical applications), formally expressed by the integral

$$g\left(x\right) = f\left(x\right) * h\left(x\right) = \int_{-\infty}^{\infty} h\left(\alpha\right) f\left(x - \alpha\right) d\alpha. \tag{2.6}$$




(a) PSF (b) MTF

Figure 2.1: PSF and MTF with and without Airy diffraction from a circular aperture with no obscuration

Importantly, the *transfer function* $H(\xi)$ is given by the Fourier transform of the PSF, $\mathfrak{F}\{h(x)\}$. As noted above, the transfer function is a representation of the system's eigenvalues; it describes the action of the system by quantifying how the system scales its eigenfunctions. Accordingly, we can find the output by representing the input $f(x)$ in the system's eigen-basis (i.e. by taking the Fourier transform $F(\xi) = \mathfrak{F}\{f(x)\}$) and scaling by the system transfer function. Formally, we have

$$G(\xi) = F(\xi) H(\xi), \tag{2.7}$$

where $G(\xi)$ is the Fourier representation of the LSI output. From the Fourier representation $G$ we can compute the spatial representation $g(x)$ using equation 2.4, also known as the inverse Fourier transform, leaving us with

$$g(x) = \mathfrak{F}^{-1}\{H(\xi) F(\xi)\} = \int_{-\infty}^{\infty} H(\xi) F(\xi) e^{i2\pi\xi x} d\xi. \tag{2.8}$$

### 2.1.2 Image Quality and Imaging Systems Trades

#### Optics

An ideal optical system would map a point source at infinity to an infinitesimal point in the focal plane. Such a system would have a PSF defined by the Dirac delta function $\delta(x)$, and a transfer function of 1 at all spatial frequencies. The physics of wave optics

do not allow this ideal system, however. Because of diffraction, the best possible optical system with a circular aperture of diameter $d_{ap}$ and focal length $f$ maps a point source at infinity of wavelength $\lambda$ and unit radiance to the Airy pattern shown in 2.1 and given by

$$I(r) = \left[ \frac{d_{ap}}{4} \frac{J_1\left(\frac{\pi d_{ap} r}{\lambda f}\right)}{r} \right]^2, \tag{2.9}$$

where $J_1$ is the first order Bessel function of the first kind and $r$ is the spatial coordinate in the focal plane [21]. The first zero of $J_1$ occurs when the argument evaluates to $1.22\pi$, leading to the well known Airy radius

$$r_{Airy} = 1.22\frac{\lambda f}{d_{ap}} = 1.22\lambda F, \tag{2.10}$$

where $F$ represents $f/d_{ap}$, typically known as f-number or numerical aperture. Using the small angle approximation $\theta \approx \frac{r}{f}$, we reach our angular resolution $\theta_{min}$, where

$$\theta_{min} = \frac{1.22\lambda}{d_{ap}}. \tag{2.11}$$

The transfer function of this ideal optical system is zero above the optical cutoff frequency $\rho_{cut}$, where

$$\rho_{cut} = \frac{d_{ap}}{\lambda f}. \tag{2.12}$$

To first order then, we can see by inverting equation 2.11 that to resolve features of angular extent as low as $\theta_{min}$, an optical system needs a diameter $d_{ap} \geq \frac{1.22\lambda}{\theta_{min}}$. Optical aberrations, however, are inevitable for any realistic system, increasing this limiting spot size and decreasing angular resolution commensurately. Accordingly, the basic task of an optical designer is to ensure sufficient system resolution, first through choice of an appropriate aperture and second through the more difficult task of minimizing aberrations.

### Noise

Having established the minimum aperture diameter necessary to achieve angular resolution of $\theta_{min}$ or better, we must next consider the the necessity of capturing the image, historically with film and today with a charge coupled device (CCD) or complementary metal-oxide-semiconductor (CMOS) sensor. If the basic task of an optical designer is to ensure sufficient system resolution, the basic task of a sensor designer is to deliver a high signal to noise ratio (SNR) from pixels spaced closely enough to preserve the resolution

of the optical system. In practice, this means maximizing quantum efficiency, minimizing sensor noise contributors, and minimizing pixel size.

CCDs and CMOS sensors convert impinging photons to photoelectrons in a semiconductor (usually silicon for visible applications) which are usually stored in a capacitor before being read out (counted) by sensor electronics. The ideal sensor would convert each impinging photon to a photoelectron and count all photo-electrons noiselessly. Physics and statistical mechanics, however, prevent such a sensor.

First, no sensor converts all impinging photons to photo-electrons; some photons are reflected, some pass through the sensor unabsorbed, and some electron-hole pairs recombine before a photo-electron can be re-absorbed. Quantum efficiency, $\eta$, summarizes these effects, with

$$\eta = \frac{\text{total photoelectrons captured}}{\text{total photon arrivals}}. \tag{2.13}$$

Although no detector can achieve 100% quantum efficiency, it is possible in many applications to find $QE$ values well above 90% in the relevant spectral band. For each pixel of quantum efficiency $\eta$, we therefore have a photon signal $n_p$ given by

$$n_p = \eta \Phi_p t_{int} p^2, \tag{2.14}$$

where $\phi_p$ is our photon flux in in units of photons per unit area, $t_{int}$ is integration time, $p$ is pixel pitch, and $n_p$ denotes *total photoelectrons captured*.

Next, we must consider the various noise sources that affect any sensor. First, the statistics of random photon arrivals means that *all* signal photons and captured photoelectrons carry intrinsic noise according to the Poisson distribution (see figure 2.2), given by the probability density function

$$P\left(N = n; \lambda\right) = \frac{\lambda^n e^{-\lambda}}{n!}, \tag{2.15}$$

where $P\left(N = n; \lambda\right)$ gives the probability that $n$ photoelectrons will be captured during some arbitrary period of time in which $\lambda$ photoelectrons should arrive on average. The Poisson distribution has variance equal to its mean $\lambda$, and therefore standard deviation $\sqrt{\lambda}$, meaning that photon noise $\sigma_p$ (also known as shot noise) is given by the square root of average photoelectrons captured $\sqrt{\bar{n}_p}$ and photoelectron variance is simply

$$\sigma_p^2 = \bar{n}_p. \tag{2.16}$$

After the noise inherent in the randomness in photon arrivals, we must consider the primary noise sources intrinsic to the sensor itself, namely dark current noise and read noise. Dark current is primarily due to thermally generated electron-hole pairs in the

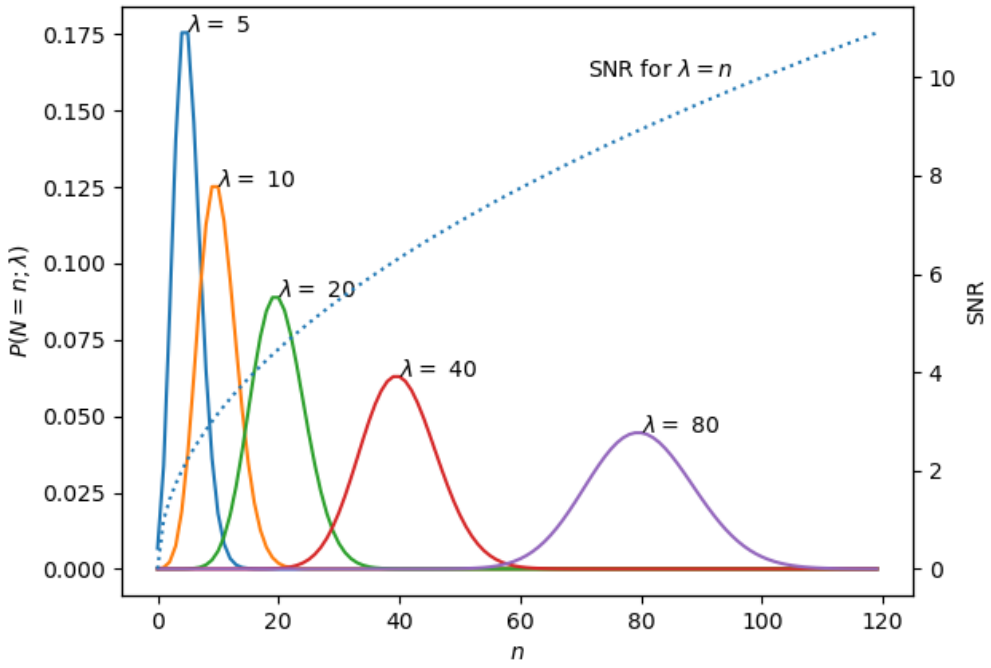



Figure 2.2: Poisson probability distributions and SNR for various $\lambda$ values

sensor itself [22]. While it is possible to remove the DC component of dark current via calibration, since dark current rates are predictable, the generation of dark electrons also follows a Poisson distribution. Accordingly, for an expected dark electron count $\bar{n}_d$, we have

$$\bar{n}_d = i_d t_{int}, \tag{2.17}$$

where $i_d$ is average dark current for each pixel in electrons per second and $t_{int}$ is integration time, leading to dark current variance

$$\sigma_d^2 = \bar{n}_d = i_d t_{int}, \tag{2.18}$$

with dark current noise $\sqrt{\bar{n}_d}$. Photon noise and dark constitute the intrinsic noise in the electrons that are collected and later counted.

After photoelectron generation and its associated noise terms, read noise constitutes the final noise term inherent in any sensor. Read noise is signal-independent and occurs

primarily due to thermal processes in sensor readout electronics and can be affected by a variety of factors, such as the size (capacitance) of the electron well that stores photoelectrons after generation, the readout rate, and the temperature of the sensor [23, 24]. Here, we will model read noise as a zero-mean Gaussian distribution of standard deviation $\sigma_r$, given by

$$P\left(N = n; \sigma_r\right) = \frac{1}{\sigma_r\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{n}{\sigma_r}\right)^2}. \tag{2.19}$$

Finally, most images are recorded with a bit depth insufficient to capture the exact number of electrons read out by the sensor electrons, driving sampling noise, the last term that we will consider here. For instance, many sensors have well depths between 10,000 and 100,000 electrons, while virtually all consumer images are stored in an 8-bit format capable of recording only 256 signal levels per pixel. If we had sensor a well depth of 25,600 electrons being read out in an 8-bit format, each output digital number (DN) would correspond to a range of 100 signal electrons. Accordingly, sampling noise follows a discrete uniform distribution given by

$$P\left(N = n; n_l, n_h\right) = \frac{1}{n_h - n_l + 1} = \frac{1}{k}, \tag{2.20}$$

where $n_l$ and $n_h$ are the lowest and highest electron counts corresponding to a particular output digital number and $k$ is the total number of discrete values that $N$ can take on. The discrete uniform probability distribution has variance

$$\sigma_u^2 = \frac{k^2 - 1}{12} \approx \frac{k^2}{12} = \frac{\left(n_h - n_l\right)^2}{12}, \tag{2.21}$$

for large ranges $n_h - n_l$, which is equivalent to the variance of the more familiar continuous uniform distribution. Using the approximation of a uniform distribution for a system with a well depth of $d_w$ electrons and a bit depth $b$, we have sampling noise variance given by

$$\sigma_s^2 = \frac{1}{12}\left(\frac{d_w}{2^b}\right)^2, \tag{2.22}$$

where the quantity $d_w/2^b$ is the number of electrons per digital number.

Since these noise sources are all independent, we can add them in quadrature to find the total noise for an image. Specifically,

$$\sigma_{total}^2 = \sum_i \sigma_i^2, \tag{2.23}$$

where $\sigma_{total}$ is total noise of an image with independent noise contributors of variance $\sigma_i^2$. Here, with the noise contributors discussed above, we have

$$\sigma_{total} = \sqrt{\sigma_p^2 + \sigma_d^2 + \sigma_r^2 + \sigma_s^2}, \tag{2.24}$$

where $\sigma_p$ is photon (shot) noise, $\sigma_d$ is dark current noise, $\sigma_r$ is read noise, and $\sigma_s$ is sampling or quantization noise. For each pixel, then, we have an SNR given by

$$\text{SNR} = \frac{n_p}{\sqrt{\sigma_p^2 + \sigma_d^2 + \sigma_r^2 + \sigma_s^2}} = \frac{n_p}{\sqrt{n_p + \sigma_d^2 + \sigma_r^2 + \sigma_s^2}}. \tag{2.25}$$

We can see from equation 2.25 that for high signal conditions, photon (shot) noise will be our dominant noise source, with SNR approaching the $\sqrt{n}$ curve shown in figure 2.2. For low signal conditions, conversely, the remaining noise terms will dominate overall SNR.

### Integration and Optimization

Having considered the main concerns of an optical design and a sensor designer, resolution and SNR respectively, we turn to the basic task of an optical designer: trading resolution and SNR. Thus far we have considered resolution and SNR in isolation, but the two are highly coupled.

To capture all of the spatial information passed by our system optics, small, closely space pixels maximize system resolution but do so at the expense of SNR. The optical quality factor, typically called $Q$, encapsulates this trade by quantifying the ratio of Airy radius to pixel spacing, with

$$Q = \frac{\lambda F}{p} \tag{2.26}$$

for pixel pitch $p$ and f-number $F$ [25, 8]. To faithfully capture an image without aliasing or contrast inversion, a sensor needs at least two pixels per Airy radius $r_{Airy}$, or $Q \geq 2$. While we can grasp intuitively that we need multiple pixels within the PSF to capture the information contained, we can return to the concept of the system transfer function to understand the important of pixel spacing more rigorously.

Here, we will begin by considering the transfer function of a single detector element in a line scanning system, ignoring for now the effects of periodic sampling. In one dimension, we can approximate the process of capturing an image at a focal plane as the convolution of a rectangular detector element with the image. Extending the nomenclature of 2.6, we will call our pre-sampling optical image $g_o$ and our final sampled image $g_f$, with

$$g_o(x) = f(x) * h_o(x), \tag{2.27}$$

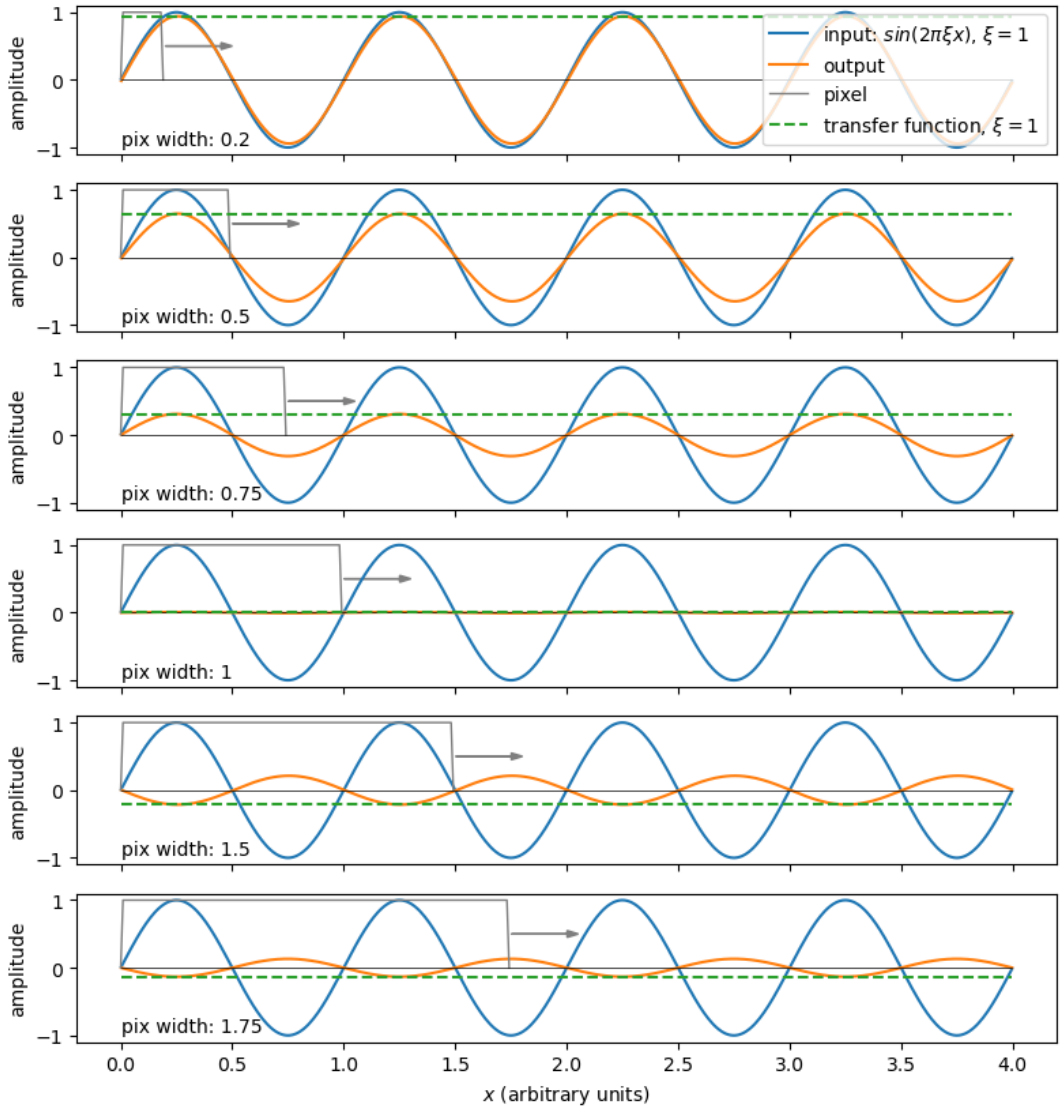



Figure 2.3: Example to illustrate the impact of pixel width on sensor transfer function. Here, we start with a a pure sine function of unit frequency/period and convolve varied width RECT (window) functions. Output signal is attenuated as window width approaches the period of the sine function. The pixel's transfer function becomes negative for pixel widths $\in (1, 2)$ sine period, causing an inversion of the output phase. (Note: pixel window functions depicted are not been normalized as shown. The output shown, however, result from convolution of the input signal with a *unit area* window function.)

where $f$ is our scene and $h_o$ is our optical PSF/impulse response. Our final image $g_f$ is given by convolution of the optical image with our detector impulse response $h_d$, meaning that our final image $g_f$ is given by

$$g_f(x) = g_o(x) * h_d(x) = f(x) * h_o(x) * h_d(x). \tag{2.28}$$

Furthermore, using the properties of transfer functions, we can take the Fourier transform of equation 2.28 and find that

$$G_f(\xi) = F(\xi) H_o(\xi) H_d(\xi), \tag{2.29}$$

which yields a system transfer function that is the product of the optical and sensor transfer functions, or $H_s = H_o H_d$.

Figure 2.3 illustrates the the impact of the detector's transfer function on a unit amplitude, unit frequency sinusoid. Our signal is fully attenuated as the pixel pitch is equal the input period (the inverse of the input spatial frequency), and we observe a contrast inversion (or phase shift of $\pi$ radians) when the pixel pitch exceeds the input period. If we incorporate periodic sampling at intervals of $d_s$, rather than simply modeling our the effect of the sensor through convolution with a rectangular detector element, we will also observe aliasing for spatial frequencies $\xi > 1/(2 \times d_s)$, commonly known as the Nyquist frequency.

To explore the impact of periodic sampling, we will consider a sensor with rectangular pixels of size (pitch) $p$ and spacing $d_s = p$ (i.e. a detector with a 100% fill factor). As described by Easton [20], we can model the action of such a sensor by convolution with the window function followed by multiplication by a normalized comb function, which consists of Dirac delta functions evenly spaced at $d_s$, resulting in a final sampled image $g_s$, with

$$g_s(x; d_s) = g_f(x) \frac{1}{d_s} \mathrm{COMB}\left(\frac{x}{d_s}\right), \tag{2.30}$$

where $g_f$ is given by equation 2.28. Taking the Fourier transform and applying the modulation theorem (see Easton chapter 9 for a proof [20]), we find that the frequency space representation of the image can be expressed

$$G_s(\xi; d_s) = \mathfrak{F}\{g_f\} \mathfrak{F}\left\{\frac{1}{d_s} \mathrm{COMB}\left(\frac{x}{d_s}\right)\right\} = G_f(\xi) * \mathrm{COMB}(d_x \xi), \tag{2.31}$$

where $G_f$ is given by equation 2.29. Convolution with the comb function is equivalent to convolution with an infinite series of Dirac delta functions spaced at frequency intervals $\Delta\xi = \frac{1}{d_s}$. Convolution with a delta function of non-zero phase (e.g. $\delta(\xi + \xi_0)$ creates a shifted copy the input signal, or

$$F(\xi) * \delta(\xi + \xi_0) = F(\xi + \xi_0) \tag{2.32}$$

for an arbitrary function $F(\xi)$. Because of this phase shifting, convolution with COMB $(d_s\xi)$ creates frequency space "echos" of the input function spaced at intervals $\frac{1}{d_s}$. Accordingly, if the original input signal has spatial frequency content above $\frac{1}{2d_s}$, frequency-shifted copies of the original signal will overlap with one another, and higher frequencies will be aliased to lower frequencies.

Figure 2.4 illustrates this frequency overlap when $d_s > \frac{1}{2|\xi|_{max}}$. Here, we use the band limited sinc function, with $\mathfrak{F}\{\text{sinc}(x)\} = \text{RECT}(\xi)$, where

$$\text{sinc}(x) \equiv \frac{\sin(\pi x)}{\pi x} \tag{2.33}$$

and

$$\text{RECT}(\xi) \equiv \begin{cases} 1 & \text{when } -\frac{1}{2} < \xi < \frac{1}{2} \\ \frac{1}{2} & \text{when } \xi = \pm\frac{1}{2} \\ 0 & \text{otherwise} \end{cases}. \tag{2.34}$$

When $d_s < \frac{1}{2|\xi|_{max}}$, the periodic repetitions (sampling artifacts) of our signal's frequency content do not overlap with its un-sampled, true frequency content. If we discard all of the frequencies larger than $|\xi| > \frac{1}{2d_s}$, we can reconstruct a representation of our original signal without aliasing. Conversely, when $d_s > \frac{1}{2|\xi|_{max}}$, frequencies above $\frac{1}{2d_s}$ will be aliased to lower frequencies in the band that will be used for reconstruction, as illustrated by the bottom row in figure 2.4 where the sampling interval $d_s = 1.25$ (arbitrary length units). For a sensor with a 100% fill factor and rectangular pixels, this aliasing begins as the same spatial frequency where phase inversion due to convolution with a rectangular pixel element occurs (illustrated in figure 2.3). Consequently, if our only concern were faithful sampling of our optical image, we would always design sensors with sampling rates of at least twice the highest spatial frequency passed by the system's optics, or

$$d_s \leq \frac{1}{2\xi_{max}} = \frac{1}{2\rho_{cut}} = \frac{d}{2\lambda f} \tag{2.35}$$

where $d_s$ is our pixel spacing and $\rho_{cut}$ is the optical cutoff frequency of our system, given by equation 2.12 for a system with aperture diameter $d$ and focal length $f$ imaging wavelength $\lambda$. For a focal plane with a 100% fill factor, where pixel pitch $p$ equals sampling period $d_s$, it is this result that explains why only systems with $Q \geq 2$ avoid aliasing and retain the spatial information passed by the system's optics.

Maintaining $Q \geq 2$, or pixel pitch $p \leq \frac{\lambda F}{2}$, however, decreases SNR by spreading the optical signal over more pixels. The well known "camera equation" quantifies this effect by relating detector irradiance $E_{detector}$ to aperture radiance $L_{aperture}$ through the quantity

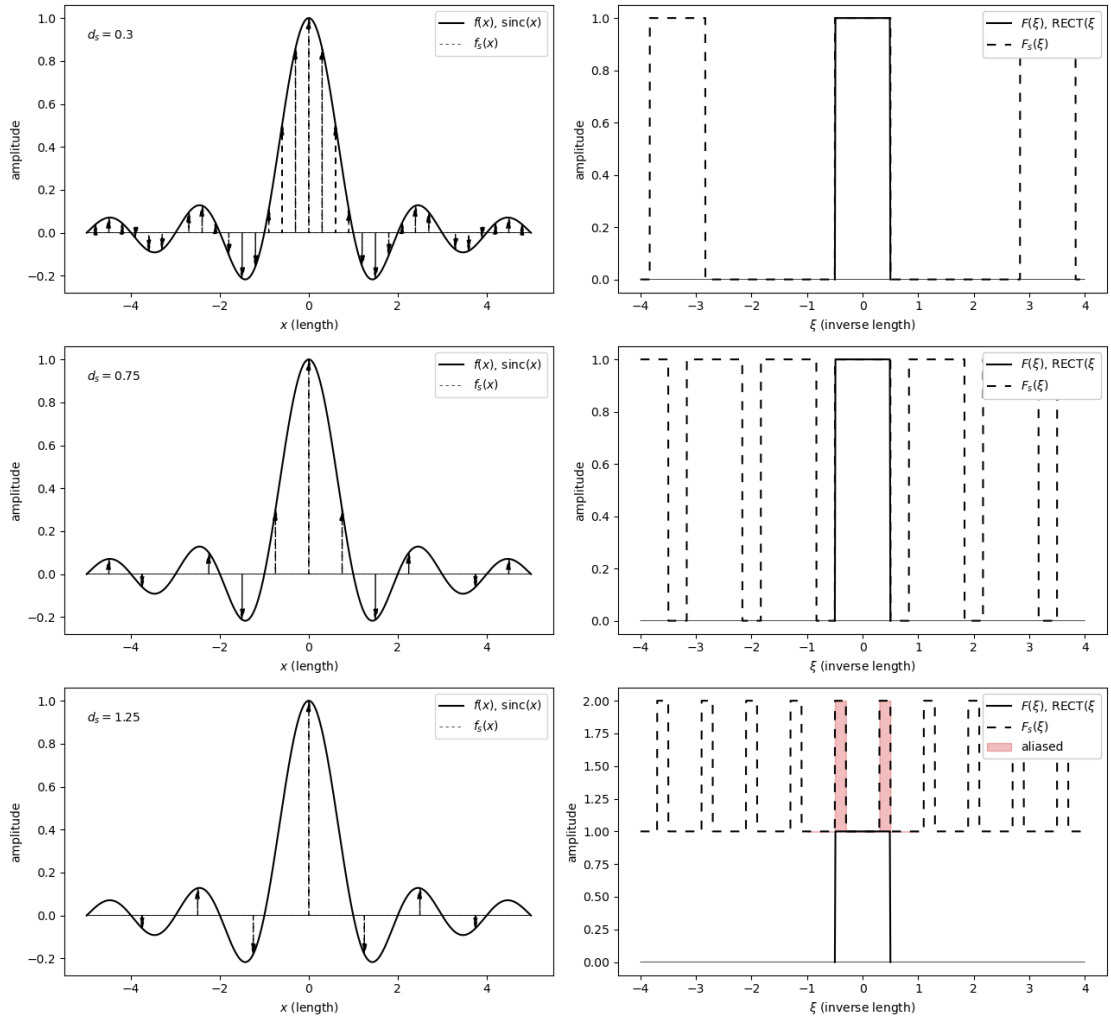



Figure 2.4: Example to illustrate the frequency-domain impact of spatial sampling periods. For a sampling period of $d_s$, frequency-domain copies of the original signal are created every frequency step $\Delta\xi = \frac{1}{d_s}$. The sinc function offers a convenient example; $\text{sinc}\,(x)$ contains an equal weighting of all spatial frequencies $\in \left[-\frac{1}{2}, \frac{1}{2}\right]$ (often denoted by the window or RECT function), with maximum spatial frequency $|\xi|_{max} = \frac{1}{2}$. When we sample at a spatial frequency $\xi_s > |\xi|_{max} = \frac{1}{2}$, where $\xi_s = \frac{1}{d_s}$, we do not cause aliasing. When we sample at a spatial frequency below $\xi_s < |\xi|_{max}$, the frequency-domain copies of the orginal signal overlap and we observe aliasing.

$G\#$, with

$$E_{detector} = \frac{L_{aperture}}{G\#} = \frac{\pi\tau}{1+4F^2} L_{aperture} \tag{2.36}$$

for optical transmission $\tau$ and f-number $F$ [26]. For reasonably large $F$, we can approximate $G\#$ with

$$G\# \approx \frac{4F^2}{\tau\pi} \tag{2.37}$$

and see that detector irradiance is roughly proportion to the inverse square of f-number.

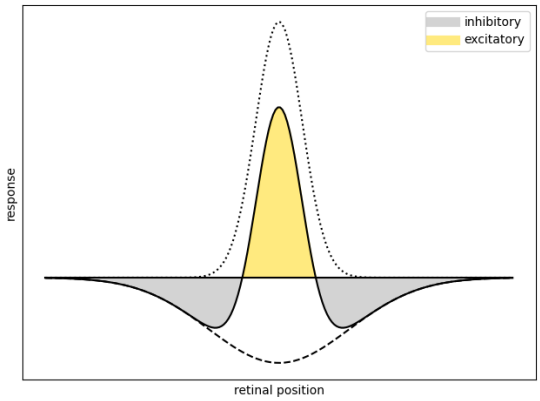
At spatial frequencies approaching $\rho_{cut}$, our optical transfer function is often approaching zero, particularly for apertures with a large fill factor. With the system transfer function already low at a high spatial frequencies, finer pixel spacing begins to offer diminishing returns. And from a practical perspective, decreasing pixel size is not always practical due to device fabrication constraints, and increasing focal length is not always practical due to system size and weight constraints. Accordingly, the basic task of a *sensor* designer is to maximize sensitivity while minimizing noise, while the basic task of the *system* designer is to optimize the trade between resolution and SNR.

For remote sensing applications, the GIQE (Eqn. 1.1) illustrates the trade offs inherent in balancing SNR and resolution. The GIQE was designed to predict the utility of overhead images, assigning a rating based on the GSD, SNR, and RER that could predict what an image analyst could do with an image [14]. The imaging conditions themselves are a major driver, with Fiete *et al.* providing an excellent summary in [8]. For high-SNR conditions, Cochrane *et al.* showed that increasing optical $Q$ does not negatively impact image quality, but the benefits even in high SNR conditions may be offset by the commensurate loss on field-of-view that comes with finer sampling [27, 25].

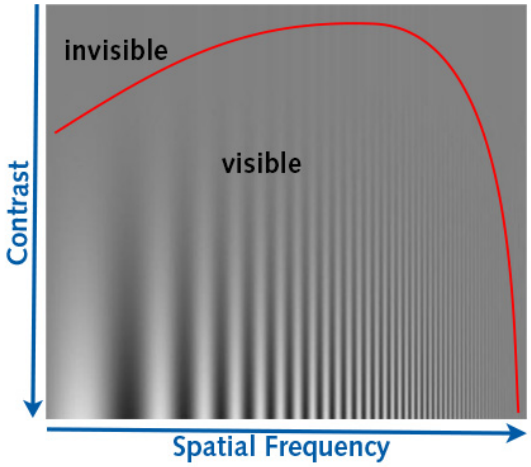
## 2.2 Human Visual System

Since all imaging prior to the (relatively recent) advent of machine vision terminated with the human visual system, it is helpful to frame our understanding of image quality and computer vision algorithms with an understanding of the human visual system.

While the optical working of the eye has been understood since Kepler [28], understanding of how the brain translates photons reaching the retina into visual *information* came much later. Perhaps most importantly for the our understanding of image quality, researchers began to model retinal responses using the difference of Gaussians. In this model depicted in figure 2.5a, a narrow center Gaussian exhibits an either excitatory or inhibitory response, a wider surround Gaussian exhibits the opposite response, and the associated neuron's response is proportional to the difference between the center and




(a) Center-surround retinal response (based on graphic in [29]



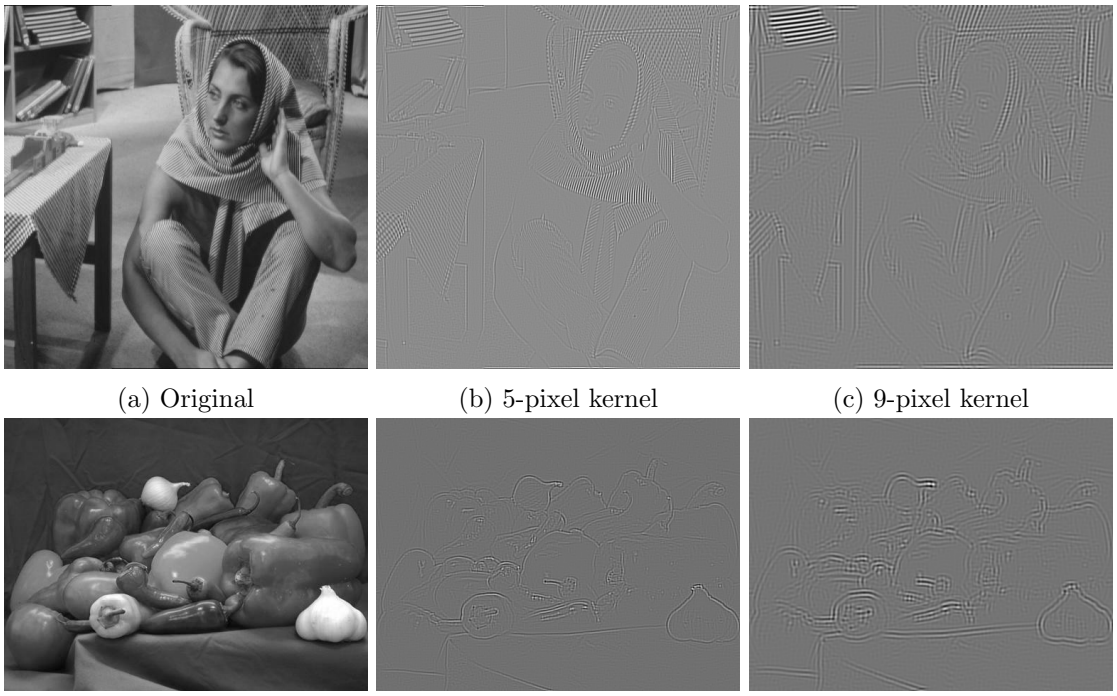

(b) Contrast sensitivity function, taken from [30]

Figure 2.5: Center surround retinal response (left) and depiction of contrast sensitivity function (right). For center-surround depicted on the left, a light spot overlapping the positive region of the resultant curve surrounded by a dark spot overlapping the negative region would maximize net response.

surround stimulus levels [29]. The size of these difference-of-Gaussian receptive fields determines the spatial frequencies to which they are most sensitive, qualitatively explaining the contrast sensitivity function depicted in figure 2.5b.

Figure 2.6 simulates the output of Gaussian center-surround receptive field to illustrate the output of center-surround receptive fields. Convolving the original image with 5-pixel and 9-pixel difference-of-Gaussian kernels, where the the sum of kernel elements $\sum_i k_i = 0$, results in figures 2.6b and 2.6c. In these figures, we observe the *contrast* selectivity of difference filters. Bright but smooth areas of the original image dark dark in the filtered images, whereas areas with contrast in the pass-band of the difference filters are bright in the filtered images. Both kernels summed to zero, making them insensitive to the constant, "DC" component of the image. Here, we observe that the light and dark regions of the filtered images are driven by the spatial frequency content of the original, with the 5-pixel kernel amplifying the high spatial frequency patterns in the scarf and chair and the 9-pixel kernel amplifying the stack of books.

Hubel and Weisel significantly extended the insights gained from the center-surround model of simple visual receptive fields. Building on work showing that optic neuron receptive fields often contained clear excitatory and inhibitory regions [31], Hubel and Wiesel



(a) Original (b) 5-pixel kernel (c) 9-pixel kernel

(d) Original (e) 5-pixel kernel (f) 9-pixel kernel

Figure 2.6: "Barbara.jpg" and "peppers.jpg" image through five- and nine-pixel difference of Gaussian filters depicted in 2.5a. For the five-pixel filter, the excitatory center and inhibitory surround Gaussians had standard deviation of 1 and 2 pixels respectively. For the 9-pixel kernel, the center and surround standard deviations were 2 and 4 respectively. All images have been contrast stretched to fill an 8-bit dynamic range.

mapped the activity in visual cortex neurons of anaesthetized cats in response to a range of visual stimuli [32]. Significantly, they identified and explored the behavior of two types of receptive fields they termed as simple and complex. Simple receptive fields typically integrated both excitatory and inhibitory retinal regions that exhibited mutual antagonism– illumination of the inhibitory region negated illumination of the excitatory region. Some of these fields took the form of light or dark spot detectors, and others behaved as edge detectors. Importantly, simple receptive fields were localizable within the retina. Complex receptive fields, conversely, were not fully localizable within the retina. Complex receptive fields exhibited responses sensitive to characteristics such as the size and orientation of visual stimuli. Broadly, Hubel and Wiesel observed receptive fields acting as *feature de-*

*tectors*, with contrast rather than raw input radiance driving output. Generally speaking, the human visual system is devoted to contrast detection.

## 2.3 Convolutional Neural Networks

Neural networks are descendants of the perceptron, which demonstrated rudimentary machine vision in the 1950s [33, 34]. The perceptron, a single-node neural network, worked by taking input vector $\mathbf{x}$ of length $n$, multiplying by weight vector $\mathbf{w}$ of length $n$, and setting the result to $\pm 1$ depending on whether the result was greater or less than zero, formally expressed

$$\hat{y} = \varphi_{activation}\left(\sum_{i=1}^{n} w_i x_i + b\right), \tag{2.38}$$

where $\varphi_{activation}$ is the activation function performing our thresholding operation, with

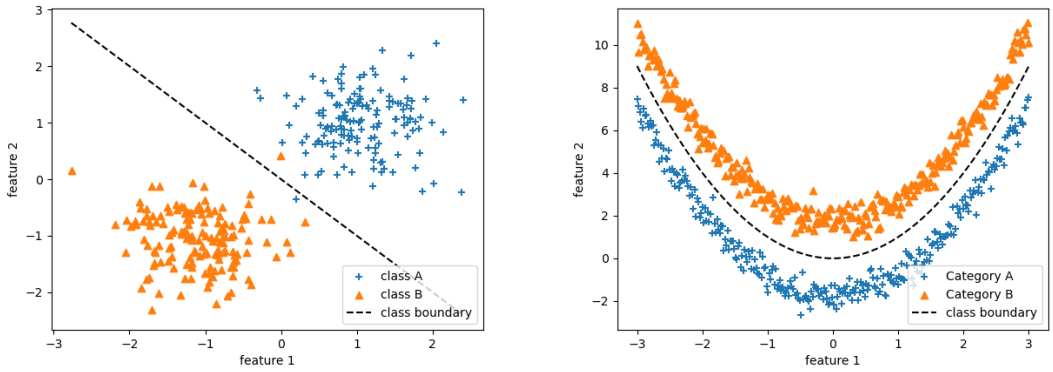
$$\varphi_{activation}(x) = \begin{cases} 1 & \text{when } x > 0 \\ -1 & \text{when } x \leq 0 \end{cases}. \tag{2.39}$$

Given a misclassified classified training training example $x_t$ with label $y_t$, weights would be updated according to the rule

$$\mathbf{w} \leftarrow \mathbf{w} + y_t \mathbf{x_t}. \tag{2.40}$$

Conceptually, we can imagine the vector $\mathbf{w}$ pointing from the mean of the examples belonging to class 1 to the mean of the examples belonging to class 2; a positive dot product between this weight vector and the input results in a prediction of class 2, a negative dot product a prediction of class 1. Fundamentally, this approach by the perceptron meant that it could only perform class separation with a *linear* hyperplane, leaving it unable to classify data that was not linearly separable (see Fig. 2.7. While this approach generated some interest, the inability to learn non-linear class boundaries largely relegated the perceptron and its immediate offspring to being curiosities at the time.

A 1986 paper by Rumelhart *et al.* reinvigorated the study of neural networks with the introduction of back-propagation [37]. We can conceptualize their work by picturing a fully connected neural network (Fig. 2.8). At a high level, a neural network maps inputs to outputs, with a loss function acting as a distance metric to quantify the difference between the network's prediction and the correct output. Rumelhart *et al.* demonstrated that we could apply the chain rule from basic calculus to calculate the contribution from each weight in the network to the final loss value for a given training example as long as the network used differentiable activation functions. Formally, for loss $L$, we can calculate



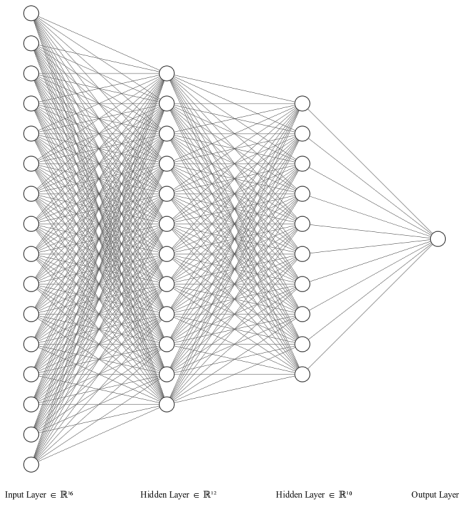

(a) Linearly separable (b) Nonlinearly separable

Figure 2.7: A 2D example of linearly and nonlinearly separable classes

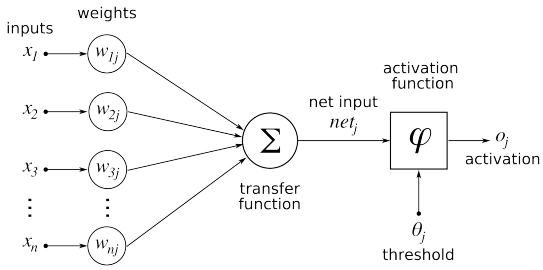
$\frac{\partial L}{\partial w_i}$, effectively tracing the loss flow through the network. While there are numerous optimization approaches to update the weights in a neural network [38, 39, 40, 41, 42, 43, 44], all amount to gradient descent algorithms designed to move in the $-\frac{\partial L}{\partial \mathbf{w}}$-direction toward the global loss minimum in the (often very large) space spanned by the neural network weights $\mathbf{w}$. Back-propagation enabled larger networks with non-linear activation functions, which in combination enabled models to learn non-linear class boundaries and handle problems too complex for the original perceptron.

These fully connected networks, however, proved inefficient for most tasks, particularly for computer vision, in large part because the features learned in an input layer were not shared globally. A training image with useful features in the upper left corner might change the input weights associated with this corner, but this "knowledge" would not be shared with the input weights associated with the upper right corner, even if the feature were equally relevant when present in either location. The convolutional neural network, originally developed by Yan LeCun and collaborators starting in the late 1980s [45, 46, 47] and brought to widespread prominence by Krizheski *et al.* in 2012 [9], overcomes this limitation by learning convolutional filters, which are applied across the input data (see Fig. 2.9).

Convolutional neural networks exhibit interesting parallels to the human visual system, learning convolutional filters which extract features analogous to those extracted by the various receptive fields in the human visual system discussed in Sec. 2.2. Figure 2.10 depicts the 64 first-layer filters learned by a relatively small, six-layer neural network trained on CIFAR-10 [48]. Although none of these filters is circularly symmetric, we can




(a) Fully connected neural network [35]




(b) Artificial neuron [36]

Figure 2.8: A fully connected neural network and an artificial neuron. All of outputs from the neurons in a layer are fed to each neuron in the subsequent layer. Each of these neurons multiplies the inputs by its own set of weights and applies its activation function and the results are passed to the next layer.

readily perceive the similarities to the center-surround, difference of Gaussian receptive field depicted in Fig. 2.5a. We observe that some of these filters are primarily color-focused while others are almost chromatically uniform and tuned to varied spatial frequencies and edges of varying orientations. These filters act as initial feature extractors, with their outputs fed to subsequent layers which will build up increasingly abstract representations of the input image.

Figure 2.11 depicts the filter outputs of a ResNet-18 model with "peppers.jpg" (also used in Fig. 2.6) as its input. The selected layer 1 and layer 5 filter outputs exhibit similarities with the 5- and 9-pixel difference of Guassian filter outputs in figures 2.6e and 2.6f, while the layer 8 filter output shown has become too abstract to be meaningful to a human viewer.

Convolutional networks have been demonstrated to perform exceedingly well in a wide range of computer vision tasks. Despite the wide ranging successes of CNNs since 2012, with CNNs often performing better than humans when trained and tested on i.i.d. datasets, CNNs struggle with generalization against unseen perturbations and distortions [50, 51, 52].

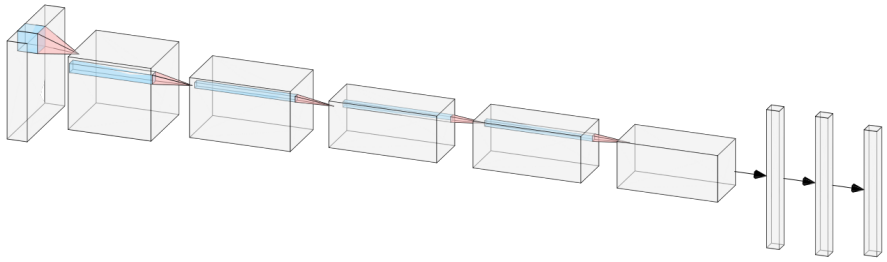


Figure 2.9: A convolutional neural network [35]. Outputs of convolutional filters (most not depicted here) are fed to subsequent layers containing additional convolutional filters, typically culminating in one or more fully connected layers similar to those in fully connected networks.

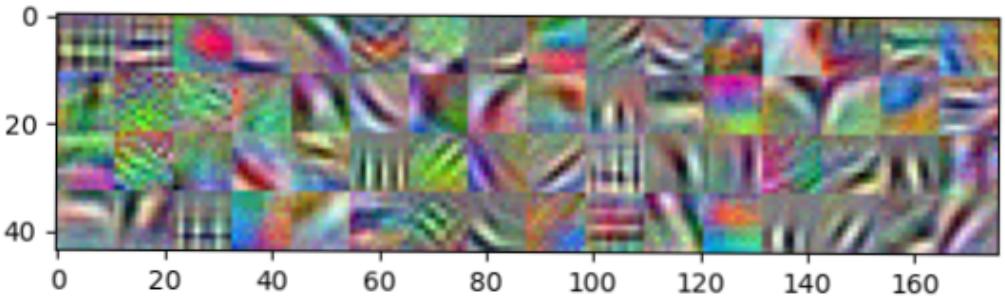
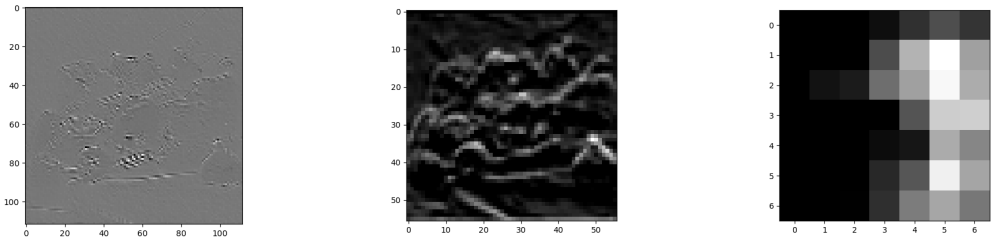



Figure 2.10: $11 \times 11 \times 3$ filters learned in the first layer of small neural network trained on CIFAR-10. These filters have been normalized and contrast stretched to fill the range $[0, 1]$. [49]



(a) Layer 1, filter 25 (b) Layer 4, filter 27 (c) Layer 8, filter 512

Figure 2.11: Intermediate representations of "peppers.jpg" after convolutional filters in an ImageNet-trained ResNet=18 model. Image representations become increasingly compressed and abstract as they pass through the network. [49]

### 2.3.1 Image Quality in Traditional Image Processing

Since digital imaging became ubiquitous, a substantial body of research has explored image quality in the context of image processing and lossy image compression. A major focus on this research has been to develop image quality metrics that correlate strongly with subjective image quality assessment by human image viewers [18, 6, 19, 53].

Some of simplest methods of error measurement rely on simple pixel difference measurements such as $L_p$ norms or Minkowski metrics, where

$$\varepsilon^{\gamma} = \left[ \frac{1}{N^2} \sum_{i,j=0}^{N-1} \left| I_0(i) - I_1(i) \right|^{\gamma} \right]^{\frac{1}{\gamma}} \tag{2.41}$$

for a single channel $N \times N$ image, with $\gamma = 2$ corresponding to the mean square error metric used widely across disciplines[19]. These pixel distance based metrics, however, generally do not correlate well with perceptual image quality; we can imagine, for instance, a large DC offset between two images that would lead to a significant $L_p$ error without significantly affecting image quality. In a successful effort to overcome some of the deficiencies with of distance-based metrics while using a relatively simple model, Wang and Bovik proposed a metric $Q \in [-1, 1]$ designed to capture changes in correlation, luminance, and contrast, with

$$Q = \frac{4\sigma_{xy}\bar{x}\bar{y}}{\left(\sigma_x^2 + \sigma_y^2 \left[ (\bar{x})^2 + (\bar{y})^2 \right]\right)} = \frac{\sigma_{xy}}{\sigma_x \sigma_y} \frac{2\bar{x}\bar{y}}{(\bar{x})^2 + (\bar{y})^2} \frac{2\sigma_x \sigma_y}{\sigma_x^2 + \sigma_y^2}, \tag{2.42}$$

where the first term captures the correlation between the images, the second term measures the DC offset between the two images, and the third captures the contrast similarities [54]. Wang and Bovik note elsewhere that many error metrics fail to capture information loss as determined by human observers and fail to capture structural distortion between images [55]. (Note that this quantity is unrelated to the concept of $Q$ described above in Sec. 2.1.2)

Other metrics used correlation values between initial and final images, image histogram similarity, Laplacian mean square error, color-angle similarity, and edge stability metrics with varying levels of success; significantly, the most effective metrics have tended to be contrast-focused, centering on attributes such as the stability of Canny edge detections and changes in spatial frequency content[18, 19]. One of the more interesting and successful methods surveyed by Avcibas applied the transfer function from a rudimentary model of the human visual system (HVS). In this method, the transfer function of the HVS is applied to the discrete cosine transform (DCT) of the original and compressed image, with the resulting spectrum converted back to the spatial domain via an inverse DCT and the error calculated via the $L_p$ norm of the HVS-weighted image [19].

A central difficulty in image quality assessment is measuring the information content of an image. The number of possible single channel, 8-bit, $28 \times 28$ images is $10^{1888}$, almost 2000 *orders of magnitude* greater than the oft cited $10^{78}$ to $10^{83}$ atoms in the observable universe. We know that the number of plausible natural images is much smaller, but quantifying the size of this smaller figure has proven challenging. Phrased differently, the actual information content of real images is vastly lower than their maximum theoretical information capacity. Measuring the information content of images has proven challenging [56, 57, 58, 59].

Claude Shannon effectively founded the field of information theory with a 1948 paper exploring the information content of text strings [60]. Shannon borrowed the concept of entropy from statistical mechanics and showed that the information content of a message in units of bits per character is given in Equation 2.43,

$$H = -\sum p_i \log_2 p_i, \tag{2.43}$$

where $H$ is the entropy of the message and $p_i$ is the probability of occurrence for the $i^{th}$ character in the message. Shannon showed that the optimal compression scheme maximizes the entropy $H$ of transmitted messages. The Shannon entropy metric has informed a range of image quality studies [61, 62, 63, 64].

Though often correlated to image quality and information content, Shannon entropy suffers from a key weakness as a metric for the information content of an image, particularly if applied naively to raw pixel values. Specifically, Shannon entropy does not account for the two-dimensional spatial relationship between pixels. Some images with no real information content can score maximal Shannon entropy values; a smooth ramp from 0 to 255 in an eight-bit image would have a Shannon entropy of eight bits per pixel. To address this problem, several studies consider methods to use image gradients to quantify image information content, primarily as a means of understanding image compressibility [59, 57]. Despite these efforts, however, no single metric has proven capable of definitely quantifying the information content of a two dimensional image.

A significant and perhaps primary element in the challenge of quantifying the information content of an image is the challenging of identifying an appropriate set of basis functions. As described above, the identity basis (i.e. raw pixel values) is prone to pathologically overestimating information content. In a 2001 paper, Wainwright et al proposed an approach based on multi-scale wavelet decomposition trees which they posited as suitable for describing the statistics of natural images [65]. Sheikh *et al.* applied these results in a series of subsequent papers, using the image statistic models developed by Wainwright to estimate the mutual information between initial and final images[63, 53]. In a broad survey of image quality metrics, Sheikh *et al.* found that their information theoretic approach outperformed other information quality metrics in predicting the perceptual image

quality rated by human observers [66].

## 2.4 Image Quality and Convolutional Neural Networks

While extensive effort has gone toward optimizing and advancing CNN capabilities [67, 68, 69], and significant research has considered how to make CNNs robust to subtle adversarial image modifications [70, 71, 72, 73, 74], less work has considered the impact of image quality on the performance of the CNNs in question. In an effort to address one of the more common image quality degradations, Zanjani *et al.* studied the impact of JPEG 2000 compression on the performance of CNNs trained to detect metastatic cancer in histopathological images, observing that JPEG compression can degrade performance but finding that CNN performance recovers and remains strong up to relatively high compression ratios when trained on similarly distorted images [75]. Several other studies have considered ways to optimize JPEG compression tables for computer vision applications [76, 77, 78, 79]. JPEG compression works by performing a 2-dimensional discrete cosine transform (DCT) on each $8 \times 8$ patch in an image, resulting in a total of 64 coefficients to losslessly represent each patch. These coefficients are then divided by values in a set of tables, with the specific table selected based on the degree of compression sought. The quotients are then rounded to the nearest integer, typically leading to a substantial fraction rounding to zero. The tables used to divide the DCT coefficients have been developed empirically to maximize image quality as perceived by the HVS. Duan and Chao both optimized compression tables for images to be used in pre-deep learning computer vision tasks [76, 77], while Li and Lui both optimized compression tables for images to be used by CNN-based classifiers, realizing measurable but modest gains (1-2 % classification accuracy improvements) through table adjustments [78, 79].

A range of other studies have considered CNN performance as a function of deliberate image degradations aside from those introduced as artifacts of image compression. Dodge and Karam studied performance of pre-trained model on images distorted with Gaussian blur and additive Gaussian noise, as well distortion due to JPEG and JPEG 2000 compression; they found that models trained on undistorted imagery are particularly sensitive to noise and particularly blur, hypothesizing that the blur vulnerability stems from the propensity of many CNNs to rely on high spatial frequencies and textures in making classification decisions [80]. In a follow up study, the same authors tuned their models on the image distortions in question and compared their CNN test accuracy to the accuracy of human test subject on the same images degraded images. They found that fine tuning the models on distorted imagery improved CNN performance against images subjected to blur and noise, but performance still lagged in comparison to human image classifiers against degraded images. Perhaps of greatest interest, the classification errors made by

CNNs and humans on the most degraded images were almost completely uncorrelated, strongly suggesting *differing classification strategies* [81].

In a pair of similar studies [50, 51], Geirhos *et al.* compared human and CNN performance across a range of image distortions, with a focus on understanding CNN generalization when trained on one particular distortion type and tested on another. In the first [50], Geirhos *et al.* found that the performance of CNNs trained on undistorted images dropped to chance level against images with moderate amounts of random noise added. When trained against independently and identically distributed (i.i.d.) image distortions, however, these CNNs often outperformed humans in classifying distorted images, while cross-distortion testing (*i.e.*, training on one type of distortion and testing against another) produced mixed results. For some distortion pairs, training against the first modestly improved test performance against the second, while in other cases training against the first distortion reduced test performance against the second. In the second study [51], Geirhos *et al.* examined the extent to which CNNs rely on texture vs. shape information, finding that CNNs trained on normal quality, undistorted images tend to rely extensively on texture in making classification decisions. Training CNNs on a synthetic database dubbed Stylized-ImageNet, which preserves object shape while largely removing texture, however, forced models to use shape information, producing strong results against both the unmodified and stylized versions of ImageNet as well as against the distorted images used in the initial study [50]. Conversely, models trained against the standard ImageNet dataset performed poorly against Stylized-ImageNet. The pair of studies showed that forcing a shape bias on CNNs generally improved robustness but found that no single image augmentation strategy proved universally optimal for classifying distorted images.

A trio of papers by Hendrycks and various collaborators focused particularly on model robustness to out of distribution (non-i.i.d.) distortions [82, 83, 84]. They began by developing two benchmarking datasets, ImageNet-C and ImageNet-P, with ImageNet-C designed to assess error rates over varied distortions and severity levels and ImageNet-P designed to assess *instability* of model predictions (i.e. probability of prediction flips) over gradually increasing perturbation levels applied to an individual image. After assessments of common models and demonstration of various robustness enhancements, the paper found that robustness against corruptions and perturbations remains a significant problem for virtually all CNN architectures [82]. Next, Hendrycks *et al.* demonstrated a unique technique for robustness enhancement that combined a stochastic mixing of image augmentations (termed AugMix) applied to individual images followed by a loss function that penalized prediction divergence between an original undistorted image and two versions of the image independently distorted using the AugMix algorithm. They found that the combination of AugMix and their new loss function achieved state of the art corruption and perturbation robustness [83]. Subsequently, however, Hendrycks *et al.* tested

a wide range of models and robustness enhancement approaches and found that while various techniques achieved reasonable robustness under certain conditions, no single approach succeeded in yielding model robustness across the full range of corruptions and perturbations considered [84].

A range of other studies have explored mechanisms and strategies for image augmentation to build robustness, usually to non-i.i.d. training and testing data [85, 86, 11, 87, 12, 88, 89]. In a study focused on robustness to blur, Vasiljevic *et al.* [85] observed that image blur degraded CNN performance as expected and found that training against blurred images resulted in better performance recovery than sharpening the images to reverse the effects of blur. In a review of CNNs for image classification, Rawat and Wang find that invariance to scaling, translation, and rotation continued to prove a challenge across architectures [86]. In comparing robustness to object occlusions, Zhu *et al.* found unsurprisingly that CNNs significantly trailed humans [11]. Wang *et al.* proposed a method to improve object detection perfromance in the presence of occlusions but observed that models tended to generate false positives based on context and correlations between objects and their typical backgrounds [12]. Schneider *et al.* found that a model can improve performance by inferring distortion statistics from multiple unlabeled examples shown during testing using batch normalization, which largely amounts to mean subtraction and scaling by the standard deviation *between CNN layers* [88]. In an effort to overcome the limitations inherent in manually crafted image augmentation strategies, Cubuk *et al.* used a recurrent neural network (RNN) trained via reinforcement learning to search for the best image augmentation policies over a predefined search space, where the reward signal to the reinforcement learning algorithm was based on the validation loss after mini-batch training with a particular image augmentation policy [89]. This effort was focused on image augmentation for *undistorted* image classification, but their success in decreasing error rates suggests that a similar approach could be useful in training models for distorted images.

In a particularly elegant set of experiments, Yin *et al.* studied model robustness using Fourier analysis tools [90]. Specifically, they began with three models – one "naturally" trained on undistorted images, another trained on images distorted with Gaussian white noise, and a third trained adversarially [1]–and tested them first on images distorted with a single 2D Fourier basis vector and next with filtered Gaussian white noise. The naturally trained model performed best against narrow bandwidth, low-pass filtered Gaussian noise, while the Gaussian white noise trained model performed best for all bandwidths of high-pass filtered Gaussian noise, with the adversarially trained model performance generally falling in the middle. Interestingly, in a subsequent experiment, this team found that train-

[1] Adversarial training generally refers to training using inputs that have been modified (often subtly) in ways engineered to cause errant neural network outputs, e.g. incorrect labels in a classification task.

ing against a low-frequency "fog" distortion actually *decreased* test performance against the same distortion while increasing robustness to distortion at a single low-frequency. Notably, this team trained an additional model using the AutoAugment algorithm introduced in [89] and tested the three original models plus the AutoAugment trained model on the CIFAR-10-C corruption robustness benchmark dataset. They found that the AutoAugment trained model performed best overall, with the Gaussian white noise trained performing better on a significant fraction the corruption cases. Yin *et al.* noted that the strong performance of AutoAugment was promising, particularly given that it was developed for *undistorted images*, but noted in agreement with others that no single augmentation approach proved universally successful.

While some of these studies identified mechanisms for improving performance against bespoke problems, and many developed mechanisms to improve robustness in a range of situations, we find no single image augmentation or training strategy optimal for all cases. While the lack of a single strategy to achieve robustness against all possible distortions represents a challenge to those interested in *artificial general intelligence*, the ability of these studies to identify strategies for *specific* problems represents a flip side strength for CNNs. If we know the types of distortions our images will experience, we can co-optimize our image collection and image analysis architectures, taking advantage of strong CNN performance when trained on appropriately distorted images.

A 2019 study offers an example of how to perform such a co-optimization [91]. In this effort, Jaffe *et al.* used the Functional Map of the World dataset as a starting point to simulate overhead imagery from payloads with varied design parameters. Of note, Jaffe *et al.* found that for a retrieval task, where CNN embeddings against test images are compared to embedding examples from known classes, pre-trained models performed best against images from simulated payloads with parameters near optimal for collecting imagery for human visual interpretation. Conversely, models fine tuned for the simulated payloads performed best against against images from shorter focal length payloads non-optimal for image quality for the human visual system. This same relationship between payload design parameters and CNN performance held for the study's classification task as well, although the direct comparison between pre-trained and fine tuned CNNs was not possible since the models needed to be trained for the classes in question. Another study on image pre-processing architectures similarly observed that the optimal solution for CNN performance did not match the optimal solution for visually pleasing images [92]. Broadly speaking, the research to date on image quality and the effects of image distortions as they relate to deep learning algorithm performance suggests that image quality as defined by the human visual system may not be synonymous with image quality as defined by computer vision systems.

# Chapter 3

# Relative Edge Response Approximations

Relative edge response (RER) represents a convenient image quality metric summarizing the sharpness of an image by quantifying the spatial derivative of an image in the direction normal to an edge. Of particular importance here, RER is one of the three parameters used in the General Image Quality Equation used by the remote sensing community to quantify the utility of overhead images [14]. Because of the Gaussian distribution's mathematical simplicity, its ubiquity in image processing libraries [93, 94, 95], and its qualitative similarity to optical point spread functions, it is often convenient to use Gaussian kernels to blur images.

In this research, we use Gaussian kernels for all image blurring, and we model CNN performance using the GIQE. In order to map our blur parameters to the appropriate GIQE variables for this modeling, here we derive and evaluate closed form functions that approximate the relationship between Gaussian blur and RER. Additionally, we evaluate the extent to which we can approximate optical point spread functions with Gaussians for the purpose of quickly mapping system point spread functions to RER using the Gaussian RER relationships that we have derived. We observe that simple Gaussian approximations do not directly predict the RER that results from convolving an ideal edge with a realistic system point spread function.

Here, we make the following contributions:

- We derive simple, closed form relationships between RER and Gaussian blur.
- We verify our RER relationships using synthetic edge images blurred with Gaussian kernels.

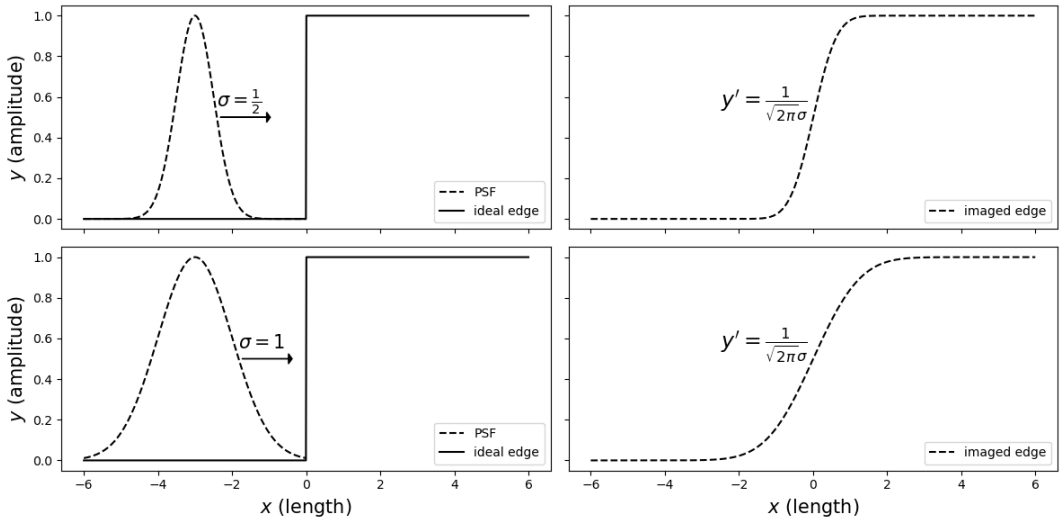



Figure 3.1: Illustration of convolution and the effect of PSF width on relative edge response (RER). To first order, RER is given by the slope of the image at the edge location.

- We show that Gaussian approximations of optical point spread functions yield images with RER that differs from the RER produced by the optical point spread functions themselves.

We have reported the findings of this chapter via arXiv [96].

## 3.1 Derivations

### 3.1.1 First order approximation

We begin by noting that we can approximate an imager as a linear shift invariant (LSI) system, allowing us to model the action of the system as a convolution of an input scene with the system's point spread function, where the convolution operation in given by

$$g(x) = f(x) * h(x) = \int_{-\infty}^{\infty} h(\alpha)\, f(x-\alpha)\, d\alpha = \int_{-\infty}^{\infty} h(x-\alpha)\, f(\alpha)\, d\alpha. \tag{2.6}$$

To derive the relationship between Gaussian blur and RER, we consider a 1-dimensional image $g(x)$ formed by convolution of the edge object $f(x)$ with the the system point spread

function $h(x)$. The edge object is described by the unit step function

$$f(x) = \text{STEP}(x) = \begin{cases} 0 & \text{where } x < 0 \\ \frac{1}{2} & \text{where } x = 0 \\ 1 & \text{where } x > 0 \end{cases}, \tag{3.1}$$

and we approximate our point spread function $h(x)$ as the normalized Gaussian

$$h(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-x^2}{2\sigma^2}\right). \tag{3.2}$$

Applying the definition of a convolution, we see that we can express our 1-dimensional image $g(x)$ with the integral

$$g(x) = \int_{-\infty}^{\infty} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-\alpha^2}{2\sigma^2}\right) \text{STEP}(x-\alpha)\, d\alpha. \tag{3.3}$$

Exploiting the properties of a step function and reversing the limits of integration for convenience, which is allowable since relative edge response is sign independent, we reach the expression for our edge image

$$g(x) = \int_{-\infty}^{x} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-\alpha^2}{2\sigma^2}\right) d\alpha \tag{3.4}$$

RER describes the sharpness of an image based on the slope of its edge response function; RER is a first order approximation of the spatial derivative of an image at the edge location [97]. In our 1D example, therefore, we have

$$\text{RER} \approx \frac{d}{dx} \int_{-\infty}^{x} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-\alpha^2}{2\sigma^2}\right) d\alpha \Big|_{x=0} = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-x^2}{2\sigma^2}\right) \Big|_{x=0} = \frac{1}{\sigma\sqrt{2\pi}}, \tag{3.5}$$

which is illustrated by Fig. 3.1.

### 3.1.2 Refined approximation

While RER is an approximation of the derivative of an image at the location of an edge, it is by definition a discrete approximation of this slope, measured by interpolating the edge spread function at $\pm\frac{1}{2}$ pixel [97]. Without accounting for the discrete sampling inherent in measurement, RER could approach infinity for sufficiently narrow point spread functions. To account for this sampling, we start wth edge image

$$g(x) = \int_{-\infty}^{x} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-\alpha^2}{2\sigma^2}\right) d\alpha \tag{3.6}$$

formed by a system with a Gaussian PSF of standard deviation $\sigma$. Applying the definition of RER that accounts for discrete measurement, we have

$$\mathrm{RER} = g\left(0.5\right) - g\left(-0.5\right), \tag{3.7}$$

which is much less than $\frac{1}{\sigma\sqrt{2\pi}}$ as $\sigma \to 0$. To account for the effects of discrete sampling at $x = \pm 0.5$, we note that there is a convenient closed form relationship for $g\left(0.5\right) - g\left(-0.5\right)$. If we re-write $g\left(x\right)$ as the sum of two integrals, with

$$g\left(x\right) = \int_{-\infty}^{0} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-\alpha^2}{2\sigma^2}\right) d\alpha + \int_{0}^{x} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-\alpha^2}{2\sigma^2}\right) d\alpha, \tag{3.8}$$

we can discard first term since it is constant and falls out in subtraction, finding that

$$\mathrm{RER} = \int_{0}^{\frac{1}{2}} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-\alpha^2}{2\sigma^2}\right) d\alpha - \int_{0}^{-\frac{1}{2}} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-\alpha^2}{2\sigma^2}\right) d\alpha. \tag{3.9}$$

In this form, we can see that RER is given by the difference of two scaled error functions, where the error function $\mathrm{erf}\left(x\right)$ is given by

$$\mathrm{erf}\left(x\right) = \frac{2}{\sqrt{\pi}} \int_{0}^{x} e^{-t^2} dt. \tag{3.10}$$

Using the change of variables

$$t = f\left(\alpha\right) = \frac{1}{\sqrt{2}\sigma}\alpha, \tag{3.11}$$

we arrive at the expression

$$\mathrm{RER} = \frac{1}{\sigma\sqrt{2\pi}} \left( \int_{0}^{\frac{1}{2\sqrt{2}\sigma}} e^{-\alpha^2} \sqrt{2}\sigma d\alpha - \int_{0}^{\frac{-1}{2\sqrt{2}\sigma}} e^{-\alpha^2} \sqrt{2}\sigma d\alpha \right), \tag{3.12}$$

which simplifies to

$$\mathrm{RER} = \frac{1}{2}\left(\mathrm{erf}\left(\frac{1}{2\sqrt{2}\sigma}\right) - \mathrm{erf}\left(\frac{-1}{2\sqrt{2}\sigma}\right)\right) = \mathrm{erf}\left(\frac{1}{2\sqrt{2}\sigma}\right) \tag{3.13}$$

### 3.1.3 Extension to multiple blur stages

In studying image quality, we are likely to be concerned with estimating the RER of images that have two distinct blur contributions, first by their system PSF and second in post-processing. For a real image $g\left(x\right)$, therefore, we have

$$g\left(x\right) = f\left(x\right) * h_0\left(x\right) * h_1\left(x\right) \tag{3.14}$$

where $f$ is the object imaged, $h_0$ is the system PSF, and $h_1$ is the Gaussian blur kernel applied in post-processing. If we approximate the optical psf $h_0$ as a Gaussian of standard deviation $\sigma_0$, then

$$h_0(x) = \frac{1}{\sigma_0\sqrt{2\pi}} \exp\left(\frac{-x^2}{2\sigma_0^2}\right), \tag{3.15}$$

$$h_1(x) = \frac{1}{\sigma_1\sqrt{2\pi}} \exp\left(\frac{-x^2}{2\sigma_1^2}\right), \tag{3.16}$$

and

$$g(x) = f(x) * h_{effective}(x), \tag{3.17}$$

where

$$h_{effective}(x) = h_0(x) * h_1(x). \tag{3.18}$$

Here, we can apply the filter theorem (discussed in detail in [20]) and note that the Fourier transform of our image $\mathfrak{F}\{g(x)\} = G(\xi)$ is given by

$$G(\xi) = \mathfrak{F}\{f(x)\} \cdot \mathfrak{F}\{h_{effective}(x)\} = F(\xi) \cdot H_{effective}(\xi), \tag{3.19}$$

where $H_{effective}$ represents the effective optical transfer function and is given by

$$H_{effective}(\xi) = H_0(\xi) \cdot H_1(\xi). \tag{3.20}$$

Using the Fourier properties of a Gaussian, we have

$$H_0(\xi) = \exp\left(2\pi^2\sigma_0^2\xi^2\right), \tag{3.21}$$

$$H_1(\xi) = \exp\left(2\pi^2\sigma_1^2\xi^2\right), \tag{3.22}$$

and

$$H_{effective}(\xi) = \exp\left(2\pi^2\left(\sigma_0^2 + \sigma_1^2\right)\xi^2\right). \tag{3.23}$$

Having found the effective transfer function, we can find the effective point spread function by taking the inverse Fourier transform of the effective transfer function according to

$$h_{effective}(x) = \mathfrak{F}^{-1}\{H_0(\xi) \cdot H_1(\xi)\}. \tag{3.24}$$

Here,

$$\begin{aligned} h_{effective}(x) &= \mathfrak{F}^{-1}\left\{\exp\left(2\pi^2\left(\sigma_0^2 + \sigma_1^2\right)\xi^2\right)\right\} \\ &= \frac{1}{\sqrt{2\pi\left(\sigma_0^2 + \sigma_1^2\right)}} \exp\left(\frac{x^2}{2\left(\sigma_0^2 + \sigma_1^2\right)}\right) \end{aligned} \tag{3.25}$$

(a) $\sigma = 1$ (b) $\sigma = 2$ (c) $\sigma = 4$ (d) $\sigma = 8$

Figure 3.2: Synthetic edge images with varied Gaussian blur.

$$= \frac{1}{\sqrt{2\pi}\sigma_{effective}} \exp\left(\frac{x^2}{2\sigma_{effective}^2}\right),$$

where $\sigma_{effective} = \sqrt{\sigma_0^2 + \sigma_1^2}$.

Accordingly, for two stages of Gaussian blur, we can model the effective point spread function as a Gaussian of standard deviation $\sigma_{effective}$, where

$$\sigma_{effective} = \sqrt{\sigma_0^2 + \sigma_1^2}. \tag{3.26}$$

## 3.2 Method

To evaluate the RER models derived above, we generated synthetic edge image chips (Fig. 3.2) and measured RER using the slanted edge method outlined in [97], with our code available at [98]. Specifically, we generated ideal slanted edges by defining the location of a near-vertical in an $xy$-plane onto which to superposed our pixel grid. We set pixels to the left of the edge equal to our dark value and pixels on the right side of the edge to our light value, and we assigned values to the border pixels according to the fraction of each on the light and dark side of the edge. Next, applied varying levels of Gaussian blur to our ideal edge image to generate edge images of varying RER. Last, we down-sampled a subset of our edge images using integer pixel binning. Table 3.1 shows the blur parameters used for image chips without downsampling, and Tab. 3.2 displays the parameters used for image chips that were down-sampled after blurring.

We performed this down sampling in order to approximate the process of applying optical blur in the analog domain and then down-sampling with a focal plane array. We used *integer pixel binning* in order to avoid the effects of pixel interpolation. For images with down-sampling applied, blur is linearly scaled by the down-sampling ratio.

Table 3.1: Parameters used in generating edge images with one and two-stage Gaussian blur and no down sampling

| | |
|---|---|
| First stage blur | 0.1 - 3 pixels |
| Second stage blur | 0 - 3 pixels |
| Combined blur *(two stage images)* | 0.1 - 4.25 pixels |

Table 3.2: Parameters used in generating edge images with two-stage Gaussian blur and integer down-sampling

| | |
|---|---|
| First stage blur | 0.75 - 6 pixels |
| Second stage blur | 0 - 6 pixels |
| Combined blur *(before down sampling)* | 0.75 - 8.5 (pixels) |
| down sampling ratios | 2, 3, 4, 5 (dimensionless) |
| Combined effective blur *(after down sampling)* | 0.15 - 4.2 pixels |

Finally, we assessed the extent to which Gaussian approximations of optical PSFs yielded equivalent RER values when used to blur synthetic edge chips. Do do so, we simulated system point spread functions using the code developed and described by Conran in [99]. Conran's model incorporates the optical system parameters shown in Tab. 3.3, with our simulations encompassing the ranges shown.

Table 3.3: System parameters used in optical PSF simulation

| | |
|---|---|
| f-number | 20 (dimensionless) |
| aperture fill factor | 0.8 (dimensionless) |
| pixel pitch | 8 $\mu m$ |
| wavelength | 0.8 $\mu m$ |
| wavefront error | 0.025 - 0.135 $\mu m$ |
| smear | 0.05 - 0.15 pixel |
| rms jitter | 2.6e-5 - 5e-4 pixel |
| down sampling ratios | 1, 2 (dimensionless) |

For each optical PSF generated, we found the nearest two dimensional Gaussian using a non-linear least squares fitting routine. For each simulated optical PSF and its Gaussian

best fit sibling, we blurred an ideal synthetic edge and measured resulting RER.

## 3.3 Results and Analysis

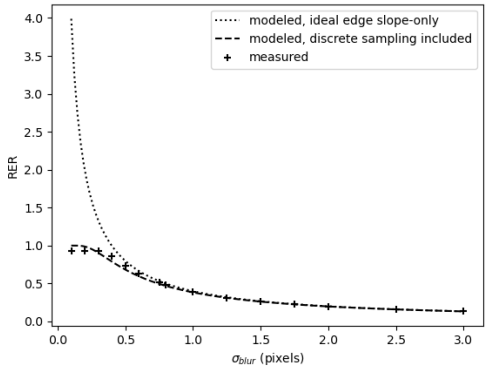
### 3.3.1 Gaussian Point Spread Functions

For synthetic edge images blurred with Gaussian kernels, we observed good agreement between our ideal edge slope model in Eqn. 3.5 for $\sigma \gtrsim 1$ pixel, with predicted RER exploding as $\sigma \to 0$. Our model incorporating discrete sampling in Eqn. 3.13 avoids the small $\sigma$ catastrophy of the first model but still does not fit the data particularly well for $\sigma < 1$. Figure 3.3 depicts these fits for modeled and measured RER using both of these models for edges blurred once and for edges blurred in two stages, where the combined standard deviation $\sigma_{effective}$ is calculated according to Eqn. 3.26 for the edges blurred twice.

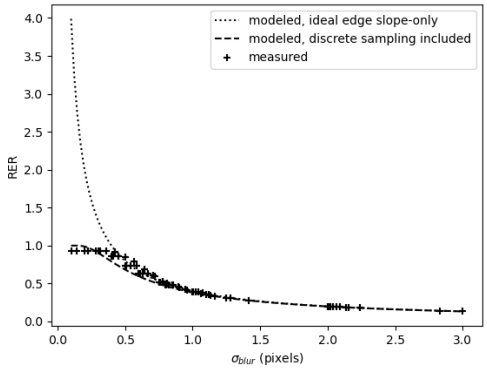
Two factors explain the relatively poor performance performance at small $\sigma$ of our Eqn. 3.13 model. First, the model over-predicts RER for very small $\sigma$ because it neglects the the transfer function of the pixels themselves. Second, at very small $\sigma$, our blur kernels cease to be Gaussian in character due to the discrete sampling inherent in kernel generation. Figure 3.4 shows the 1-dimensional profiles of the blur kernels from the Torchvision library that we used in generating our edge images. As standard deviation $\sigma$ approaches 0, our blur kernels lose their Gaussian character and approach discrete delta functions. This non-Gaussian character of our blur kernels tends to drive RER up for small $\sigma$, leading our 3.13 model to under-predict RER for moderately small $\sigma$. Because of this effect, our simplest model in Eqn. 3.5 yields the best prediction for RER when $\sigma > 0.5$ pixels and the images have not been down-sampled.

To work around the effects of discrete blur kernels, we next consider synthetic edge images blurred at high resolution and then down-sampled. With down-sampling after blurring, our final Gaussian effective standard deviation is scaled by the same ratio as the image, enabling larger kernel standard deviations and therefore kernels of a more Gaussian character before down-sampling occurs. For our synthetic edge images blurred this way to better approximate optical imaging, we observed that our simplest model (Eqn. 3.5) still performs reasonably well for $\sigma > 1$ pixel, above which we avoid the problems inherent in the $1/\sigma$ relationship. We also see that our Eqn. 3.13 discrete sampling now systematically over predicts RER at all small $\sigma$, as shown in Fig. 3.5.

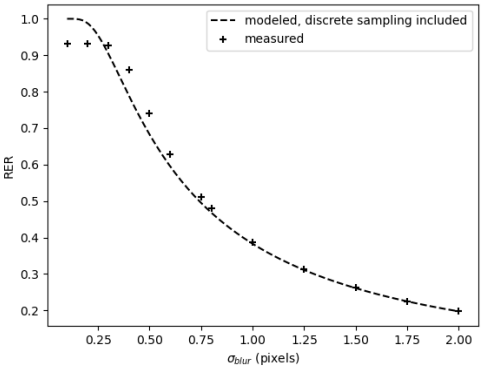
We can understand this divergence between RER as measured and RER as predicted by the discrete sampling model at low $\sigma$ by recognizing that the transfer function of the pixels themselves significantly impacts RER at low blur values. Importantly, this transfer function is *distinct* from the discrete sampling effects considered in 3.1.2 where we derived



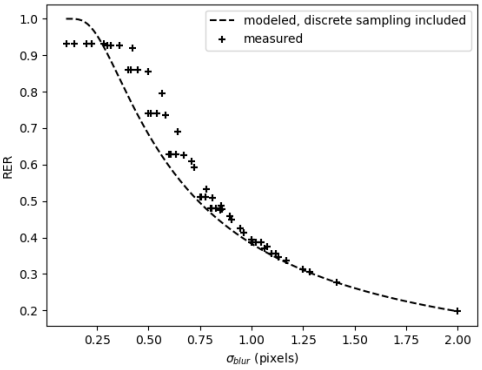

(a) $0.1 \leq \sigma \leq 3$, single blur stage




(b) $0.1 \leq \sigma_{effective} \leq 3$, two blur stages




(c) $0.1 \leq \sigma \leq 2$, single blur stage




(d) $0.1 \leq \sigma_{effective} \leq 2$, two blur stages

Figure 3.3: Relative edge response modeled and measured.

the discrete sampling model. While the discrete sampling model accounts for the impact of approximating the derivative of the edge spread function by measuring discretely at $\pm 1/2$ pixel, the pixel transfer function significantly changes the edge spread function by averaging the image signal across the width of the pixel.

To estimate the impact of the pixel transfer function, we calculated the combined transfer function

$$H_{combined}\left(\xi;\sigma\right) = H_{Gauss}\left(\xi;\sigma\right) \cdot H_{pixel}\left(\xi\right), \tag{3.27}$$

for a range of $\sigma$ values. Because we have specified $\sigma$ in units of pixels, we can conveniently

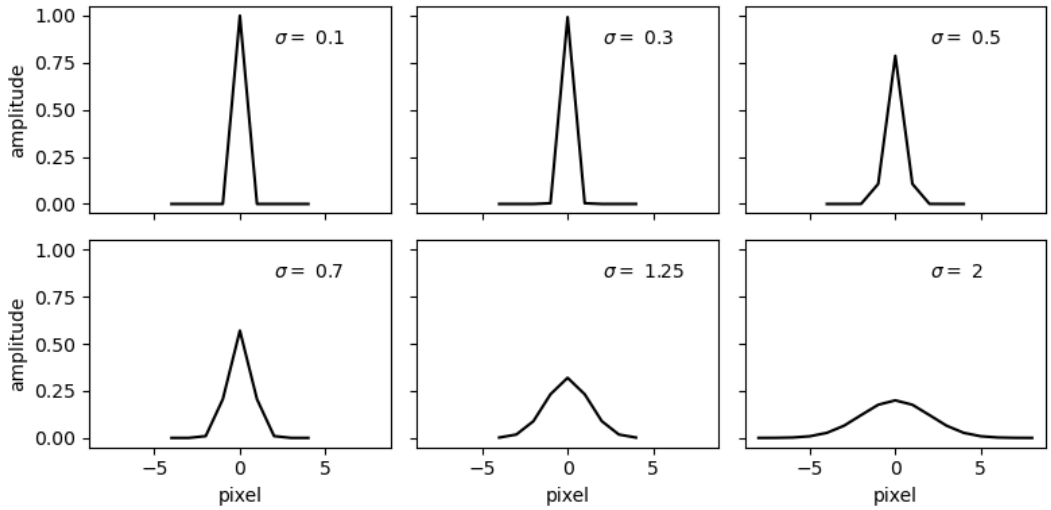



Figure 3.4: Gaussian kernel profiles from the Torchvision library. Here, we see that because of discrete sampling, blur kernels lose their Gaussian character for small $\sigma$

treat our pixels as being unit width, yielding the pixel transfer function

$$H_{pixel}(\xi) = \mathfrak{F}\{\text{RECT}(x)\} = \text{SINC}(\xi), \tag{3.28}$$

where $\text{RECT}(x)$ and $\text{SINC}(\xi)$ are a Fourier transform pair, expressed by

$$\text{RECT}(\xi) \equiv \begin{cases} 1 & \text{when } -\frac{1}{2} < \xi < \frac{1}{2} \\ \frac{1}{2} & \text{when } \xi = \pm\frac{1}{2} \\ 0 & \text{otherwise} \end{cases} \tag{2.34}$$

and

$$\text{sinc}(x) \equiv \frac{\sin(\pi x)}{\pi x} \tag{2.33}$$

respectively. The transfer function of a Gaussian is a second Gaussian with standard deviation inversely proportional to the first Gaussian's standard deviation [20], leading to transfer function

$$H_{Gauss} = \mathfrak{F}\left\{\frac{1}{\sigma\sqrt{2\pi}}\exp\left(\frac{-x^2}{2\sigma^2}\right)\right\} = \exp\left(2\pi^2\sigma^2\xi^2\right) \tag{3.29}$$

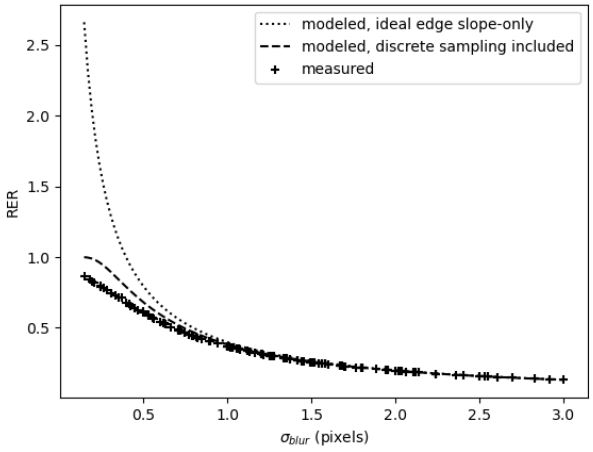



Figure 3.5: RER as a function of Gaussian blur, measured and predicted by ideal edge slope model (Eqn. 3.5) and discrete sampling model (Eqn. 3.13). These

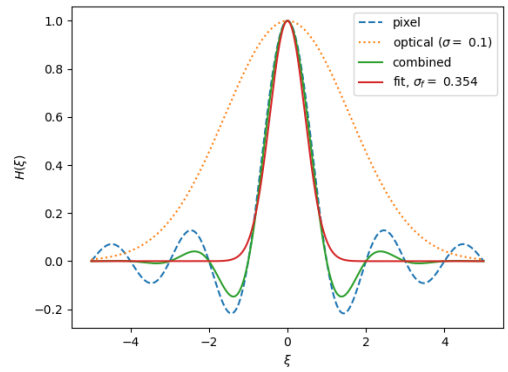
for a Gaussian PSF of standard deviation $\sigma$. For a Gaussian PSF of standard deviation $\sigma$, therefore, we have a combined transfer function

$$H_{combined}\left(\xi;\sigma\right) = \exp\left(2\pi^2\sigma^2\xi^2\right) * \mathrm{SINC}\left(\xi\right). \tag{3.30}$$

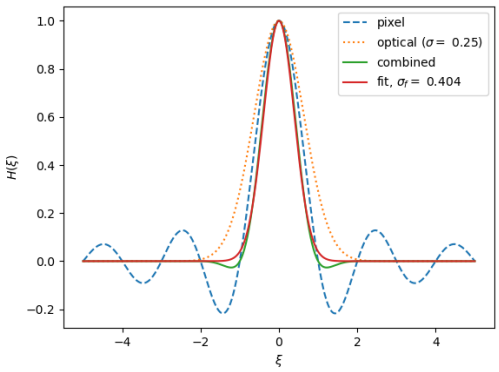
Figure 3.6 illustrates the interactions between the two terms in this transfer function. For wide Gaussian PSFs with large $\sigma$, we have narrow Gaussian transfer functions, in which case the pixel transfer function has minimal impact. Conversely, for narrow Gaussian PSFs with small $\sigma$, we have wide Gaussian transfer functions, in which case the pixel transfer function has a significant impact. We fit a Gaussian of the form $H_f\left(\xi\right) = \exp\left(2\pi^2\sigma_f^2\xi^2\right)$ to each combined transfer function $H_{combined}$ and observed that for small $\sigma$, the difference between the original PSF blur parameter and the best fit Gaussian blur parameter $\sigma_f$ varied significantly due to the impact of the pixel transfer function. Figure 3.7 shows the difference between original $\sigma$ and $\sigma_f$ for varied $\sigma$, with the residuals following an approximately Lorentzian pattern, where the Lorentzian [100] is given by

$$P\left(x\right) = \frac{1}{\pi}\frac{b}{\left(x-m\right)^2+b^2}. \tag{3.31}$$

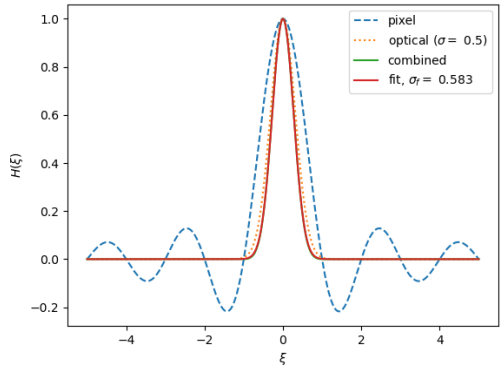
Applying the fit parameters shown in Fig. 3.7 to Eqn. 3.31, we are able to estimate a combined Gaussian that accounts for the the Gaussian PSF as well as the pixel transfer



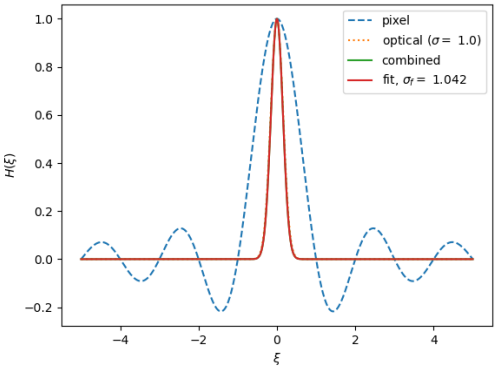

(a) $\sigma = 0.1$, $\sigma_f = 0.354$




(b) $\sigma = 0.25$, $\sigma_f = 0.404$




(c) $\sigma = 0.5$, $\sigma_f = 0.583$




(d) $\sigma = 1$, $\sigma_f = 1.042$

Figure 3.6: Combined transfer functions for varying Gaussian PSF widths and unit width pixels. Note that in all cases $\sigma$ refers to the standard deviation of a normalized Gaussian PSF, whereas it is inversely proportional to the width of the Gaussian transfer function (see Eqn. 3.29). Here, we see that the pixel transfer function begins to significantly change the combined transfer function for $\sigma < 0.5$ pixels.

function. Adding this correction to the original blur parameter $\sigma$, where

$$\sigma_{corrected} = \sigma + \frac{1}{\pi}\frac{b}{(\sigma - m)^2 + b^2}, \tag{3.32}$$

and using the corrected blur value in the RER model given by Eqn. 3.13, we observe

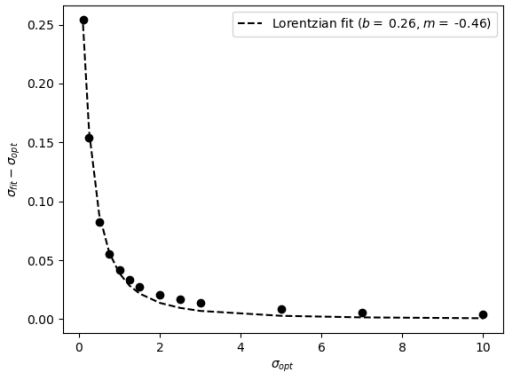
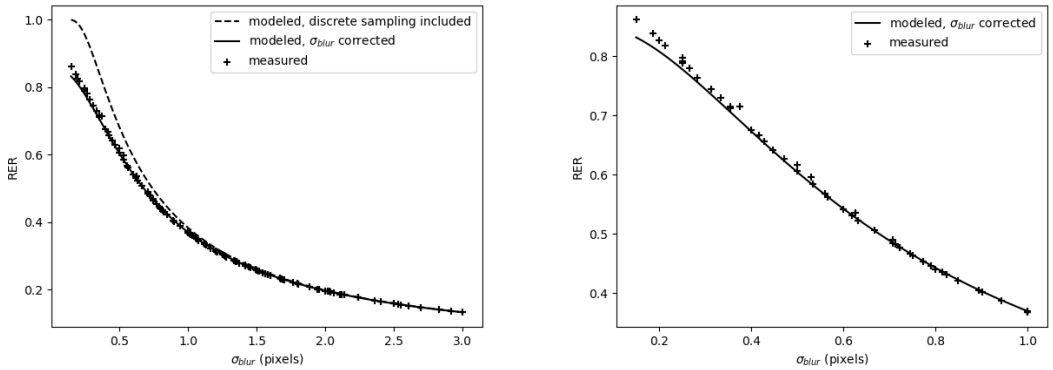



Figure 3.7: Residuals between original Gaussian $\sigma$ and the best fit $\sigma_f$ for the combined transfer functions given by 3.30

excellent agreement between modeled and measured RER across a wide range of blur parameter $\sigma$, as shown in Fig. 3.8.




(a) With and without blur correction

(b) With blur correction, small $\sigma$

Figure 3.8: RER as a function of Gaussian blur, measured and predicted by discrete sampling model (Eqn. 3.13) with and without pixel transfer function correction (Eqn. 3.32) (left) The fully corrected model performs well down to roughly $\sigma \geq 0.25$.

Given the good agreement between modeled and measured RER, we can conclude that the model given by Eqn. 3.13 can accurately predict RER as a function of Gaussian blur

down to very low $\sigma$ if we apply a correction to account for the pixel transfer function. We highlight that the correction itself is found without reference to RER but is the result of finding the best fit Gaussian for the combined transfer function that results when a Gaussian PSF is convolved with a unit width pixel RECT function; our RER model is never fit to RER results, suggesting that the derivation from first principles is sound.

### 3.3.2 Gaussian Approximations of Optical Point Spread Functions

Having established that we can predict the RER of a system with a purely Gaussian PSF, we next examined whether Gaussian approximations of simulated optical PSFs could be used to accurately predict RER. Table 3.3 shows the parameters used for our optical simulation. We note that our simulated PSFs correspond to a $Q = 2$ system before down-sampling, where $Q$ is the optical quality factor given by

$$Q = \frac{\lambda F}{p}, \tag{3.33}$$

for wavelength $\lambda$, f-number $F$, and pixel pitch $p$. After down sampling, our pixel pitch effectively doubles, which causes the drop in $Q$. Figure 3.9 shows the RER that results from simulated optical PSFs and their best fit Gaussian approximations.

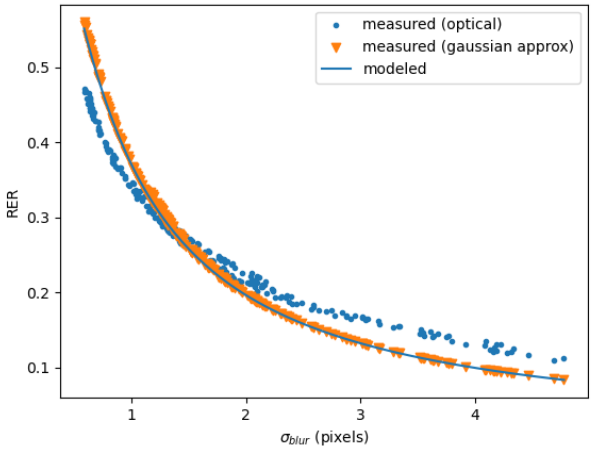



Figure 3.9: RER for images blurred with simulated optical PSFs and their best fit Gaussian approximations. Simulated PSFs are from a combination of $Q = 1$ and $Q = 2$ systems with WFE, jitter, and smear values varying within the ranges shown in 3.3.

From these results, we observe that the specific shape of optical kernels matters; the RER of an image blurred by one of our simulated optical kernels differs from the RER of an image blurred by its best-fit Gaussian kernel. We note that we used other optical simulation parameters and observed similar differences (not shown here) between the RER of images blurred with our optical kernels and their least squares Gaussian siblings.

## 3.4 Conclusions on Gaussian Blur and Relative Edge Response

Here, we have derived closed form approximations for the relationship between Gaussian blur and relative edge response, and we have shown the limitations of these approximations. For images blurred at their final resolution, we find that our models do a reasonably good job of predicting RER down to blur standard deviations of approximately 0.5 pixels, with our simplest relationship given by

$$\text{RER} \approx \frac{1}{\sigma\sqrt{2\pi}} \tag{3.34}$$

performing better than our refined approximation when $\sigma > 0.5$ but diverging rapidly as $\sigma \to 0$.

For images blurred at high resolution and then down-sampled, which better approximates the process of optical blur and sampling by a focal plane, we find that our refined approximation,

$$\text{RER} \approx \text{erf}\left(\frac{1}{2\sqrt{2}\sigma}\right), \tag{3.35}$$

yields good RER predictions for blur values greater than roughly 0.75 pixels, above which the Gaussian transfer function dominates the pixel transfer function. At smaller blur values, we can account for the pixel transfer function with the correction given by Eqn. 3.32. Additionally, when Gaussian blur is applied in two stages, we demonstrated that we can find the combined Gaussian standard deviation by combining the two standard deviations in quadrature,

$$\sigma_{combined} = \sqrt{\sigma_0^2 + \sigma_1^2}. \tag{3.36}$$

Finally, we found that blurring with the least squares Gaussian approximation of an optical PSF does not yield the same RER as blurring with the optical PSF itself.

# Chapter 4

# Classifier Performance

To begin our study of the relationship between image quality and CNN performance, we systematically map the relationship between CNN-based image classifier performance and the first order image quality parameters of resolution, blur, and noise. While a large range of distortion types are possible and have been used in the deep learning literature, we chose these particular distortions because of their relationship to physical imaging, where to first order focal length and pixel pitch drive resolution, optical quality drives blur, and sensor characteristics drive noise. We conceptualize these variables as defining a three-dimensional image quality space, and we study the variation in performance of canonical CNNs across this space as depicted in Fig. 4.1. This research makes the the following contributions:

- We quantify the relationship between three primary image quality drivers–resolution, blur, and noise–and CNN performance.
- We illustrate the ability of appropriately trained CNNs to classify images of very low visual quality sequentially degraded by down-sampling, blur, and additive noise when trained on identically distorted images.
- We demonstrate the capacity of the GIQE functional form to model computer vision performance as a function of image quality and show that the historical form of the GIQE outperforms the current form in modeling CNN accuracy in at least some circumstances.

In this chapter, we describe the details of our approach to studying this problem in Section 4.1. In Section 4.3, we evaluate the first order relationships between image quality drivers and classification accuracy as a function of model training and architecture. Using these results, in Section 4.4 we evaluate the suitability of the GIQE functional form

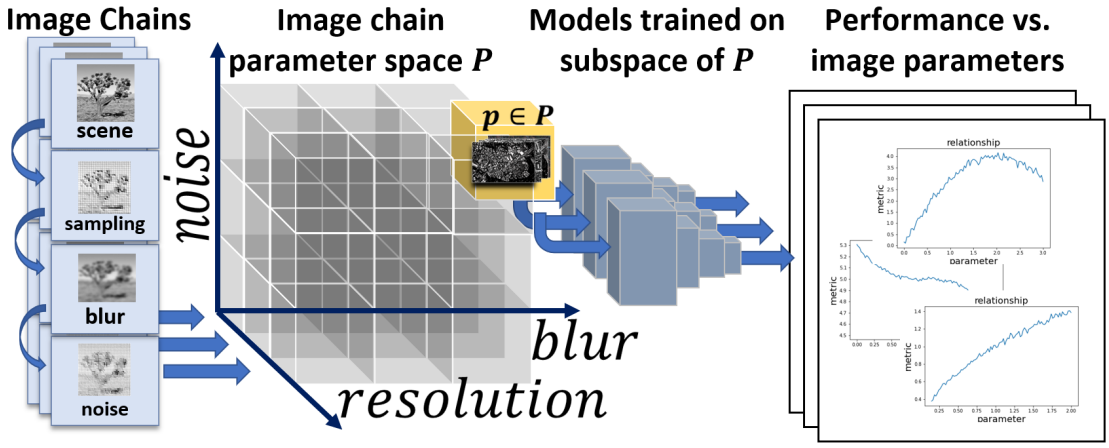



Figure 4.1: Study overview. To assess the relationship between image quality and CNN performance, we generated a series of parametric image chains which adjusted the sampling resolution, blur, and noise of the Places365 dataset. We then trained CNNs on images from a subset of our parametric image chains and measured performance as a function of image chain parameters and metrics on the resulting images themselves.

for predicting computer vision performance as a function of image distortions, and we present final observations and conclusions in Section 4.5. We presented the majority of this chapter's content at Pattern Recognition and Tracking XXXIII in 2022 and in a 2023 Journal of Electronic Imaging paper .

## 4.1 Method

We use two very dissimilar image datasets for our analysis, SAT-6 and Places365. The SAT-6 dataset consists of 405,000, 28 × 28, 1-meter ground sample distance (GSD) airborne image chips, with classes consisting of barren land, trees, grassland, roads, buildings and water bodies [101]. The Places365-Challenge dataset consists of approximately 8 million, 256 × 256 images across 365 scene categories [102]. We used the Places365 validation image set for all of our Places365 testing (the Places365 test labels have not been publicly released), and we used segregated subsets of the training images for validation during model training and tuning. We converted all images in both datasets to grayscale in order to simplify the analysis of image quality metrics and focus this analysis on the spatial information content of images rather than on the spectral. Table 4.1 summarizes the

Table 4.1: Training parameters. Places365 models were download pre-trained. Training and testing were performed on the Rochester Institute of Technology Research Computing Cluster [103].

| | SAT-6 | Places365 |
|---|---|---|
| Train dataset size | 292K | 7.9M |
| Validation split size | 32K | 81K† |
| Pre-training epochs | 30 | – |
| Distortion tuning epochs | 30 - 60 | 10 - 20 |
| Pre-training learning rate | $10^{-4}$ | – |
| Distortion tuning learning rate | $5 \cdot 10^{-5}$ | $5 \cdot 10^{-5}$ |
| Train batch size | 64 | 32 |
| Optimizer | Adam | Adam |

†Places365 validation split extracted from train images

training parameters used. We note that while we trained our SAT-6 models over more epochs than our Places365 models, total training examples for Places365 exceeded training examples for SAT-6 by roughly an order of magnitude due to the difference in dataset size. We set the number of training epochs by determining the time required for training and validation loss to level off. In each training run, we saved a model checkpoint after each epoch and ultimately selected the model from the epoch with the best validation loss.

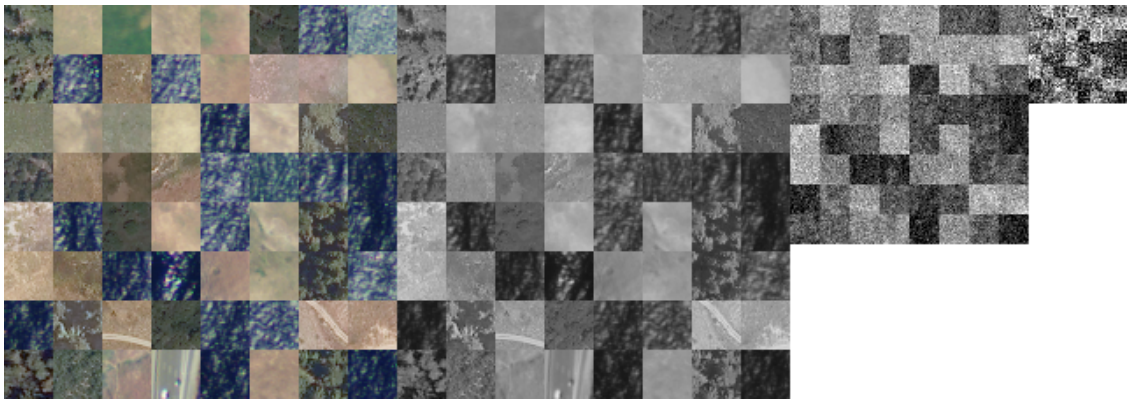
For each dataset, we used the performance of models trained on undistorted images (referred to hereafter as pre-trained models) to choose the boundaries of the distortion space that we used for training our classifiers. Specifically, we tested our pre-trained models on images degraded with a single distortion type and increased the degree of the distortion until accuracy approached chance performance. We then bounded the training distortion space (Table 4.2) using the distortion values for each distortion dimension shown to *independently* drive a pre-trained model to near-chance performance. Later, we moderately narrowed the bounds of the test distortion space to eliminate regions in which models failed to surpass chance performance and to remove low blur values ($\sigma_{blur} \lesssim 0.5$ pixels) for which the blur kernel largely lost its Gaussian character, as discussed in Ch. 3.

We created two primary sets of test images for each dataset, both spanning the full distortion space (see Fig. 4.2). We used the first to fit models predicting performance as a function of distortion and used the second to evaluate the distortion performance fits. We built these sets of test images by stochastically distorting each dataset's original high-quality test images, with each test dataset containing multiple copies of the undistorted

Table 4.2: Distortion space.

| | resolution *(fraction)* | $\sigma_{blur}$ *(pixels)* | $\sqrt{\lambda}_{Poisson}$ *(DN)* |
|---|---|---|---|
| SAT-6 *(train)* | 0.25 - 1 | 0.1 - 1.5 | 0 - 50 |
| SAT-6 *(test)* | 0.25 - 1 | 0.5 - 1.5 | 0 - 50 |
| Places365 *(train)* | 0.1 - 1 | 0 - 5 | 0 - 50 |
| Places365 *(test)* | 0.2 - 1 | 0.5 - 4.5 | 0 - 44 |

(a) Representative image for Places365



(b) $9 \times 9$ mosaic of SAT-6 images

Figure 4.2: Places365 representative images (top) and a $9 \times 9$ mosaic of SAT-6 example images (bottom) starting with **original RGB** ***(far left)***, converted to grayscale but otherwise **undistorted** ***(second from left)***, at the test distortion space **midpoint** ***(second from right)***, and at the test distortion space **endpoint** ***(far right)***. Visually, images are very low quality by the midpoint and effectively indecipherable at the endpoint.

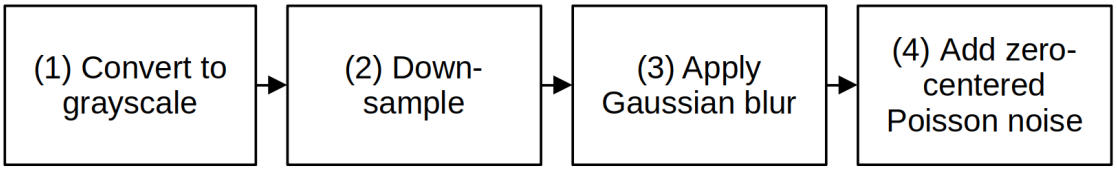



Figure 4.3: Image chain steps

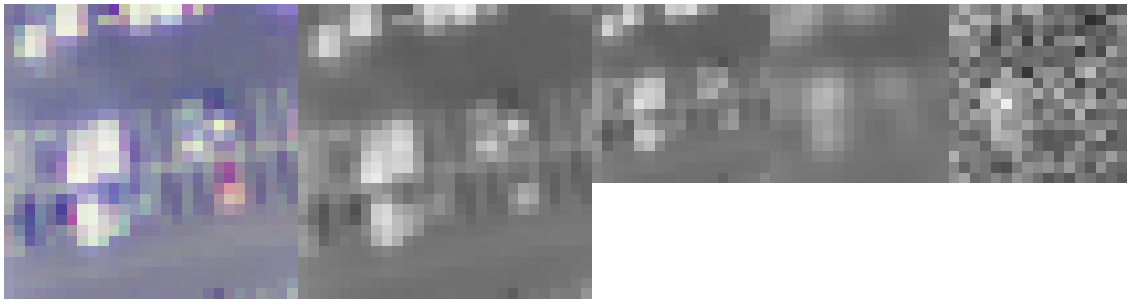
parent images. For SAT-6, our test datasets contained eight stochastically degraded copies of each original test image, and for Places365 our test datasets contained 20 copies, for a total of 648,000 and 700,000 test images respectively. While typical CNN performance evaluations use individual test images only once, we note that this research focuses on performance as a function of distortion. Here, having the same underlying test images appear at varied distortion points helps to isolate and understand the impacts of the distortions applied. Where we tested on point distortion datasets (e.g. Fig. 4.9), we used each of the original undistorted images only once.

To apply our distortions, we loosely emulate the process of imaging with a physical image chain (Fig. 4.3). The preprocessing routine sequentially applied down-sampling, blur, and noise, mimicking the physical imaging process which maps an angular field of view onto discrete detector elements, imparts optical blur, and adds sensor noise in the process of collecting an image. We began with high quality images from our original datasets and converted them to grayscale in order to focus our study on the spatial information content of our images. We then downsampled our images using bi-linear interpolation without anti-aliasing. For ease of analysis and smoother integration with the PyTorch library, we used Gaussian blur kernels. Finally, we added zero-centered Poisson noise, where we subtracted the mean/variance $\lambda$ of the Poisson distribution, resulting in a modified Poisson distribution of the form

$$P\left(N^{'} = n^{'}; \lambda\right) = P(N = n; \lambda) = \frac{\lambda^{n} e^{-\lambda}}{n!}, \tag{4.1}$$

where $n^{'} = n - \lambda$ and $N^{'}$ is the random variable representing our modified Poisson process. We used zero centering to avoid saturating our images, since a Poisson distribution with a standard deviation of only 16-counts would have a mean of 256-counts and saturate an 8-bit image. We chose a Poisson distribution since it governs key physical noise processes (*e.g.*, shot noise and dark current noise), but we readily acknowledge the tenuous link between the Poisson distributions in pre-readout electrons and a Poisson distribution that is specified in terms of digital number (DN), zero-centered, and added after secondary

(a) Places365 image chain stages at distortion space midpoint



(b) SAT-6 image chain stages at distortion space midpoint

Figure 4.4: Examples transformations of original-quality Places365 and SAT-6 color images through their respective image chains: (1) conversion to grayscale, (2) down-sampling, (3) blurring, and (4) addition of zero-centered Poisson noise. Image chains here correspond to the midpoints of the Places365 and SAT-6 test distortion spaces respectively. Note that magnification of the Places365 and SAT-6 image strips differs by roughy a factor of nine, with Places365 images beginning at $256 \times 256$ pixel resolution compared to $28 \times 28$ for SAT-6.

processing of a real image. Fig. 4.4 illustrates the evolution of two images in our synthetic image chain, one from each dataset.

We next trained three model architectures–ResNet18, ResNe50, and DenseNe161–across the full distortion space to verify that the performance trends are similar across model architectures. After verifying the similarity of distortion performance across architectures, we used ResNet18-based models to study distortion-performance relationships in greater detail, choosing ResNet18 because of its lower computational requirements.

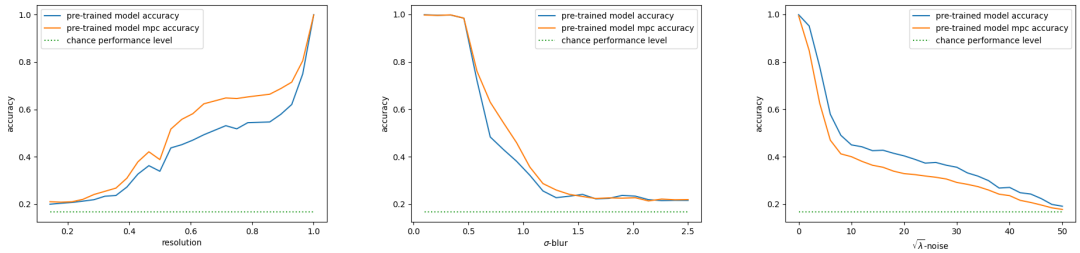
To approximate the performance of an ideally trained CNN, we created what we have termed "composite performance results" from eight constituent models. To do so, we divided each distortion dimension into halves, the combinations of which subdivided the

overall distortion space into $2^3 = 8$ octants, and then tuned a model on each octant's distortion subspace. For instance, the Places365 model trained on the octant with the highest quality images saw resolution fractions chosen randomly between 0.55 and 1, Gaussian blur values ranging from 0 to 2.5 pixels, and Poisson noise standard deviations ranging from 0 to 25 DN, where all distortion values within these ranges are randomly selected for each training image. We then tested each model on both i.i.d. test datasets. Using the octant models' test results against the ***first*** *test dataset*, we identified the best performing model for images belonging to each distortion octant. For images in the ***second*** *test dataset*, we filtered the predictions of our models according to their performance against images from each octant in the first test dataset. In other words, we used the model that performed best against a given octant in the first dataset for images from the same octant in the second dataset. While further subdividing the distortion space would likely allow a closer approximation of the performance of ideally trained models at each distortion point, we believe that continuing to subdivide the distortion space into $3^3 = 27$ or $4^3 = 64$ sub-spaces would offer diminishing returns based on the results to be discussed in Section 4.3.3.

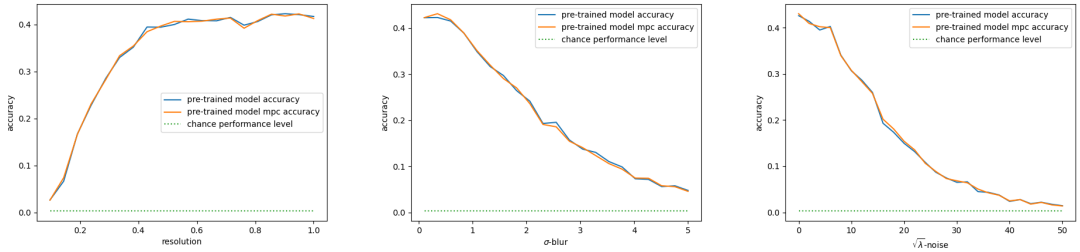
## 4.2 Establishing and Refining Distortion Space Bounds

To set the bounds of our distortion spaces for both Places365 and SAT-6, we searched for the level of each distortion type that drove a *pre-trained* model to approach chance performance. Specifically, we created a single-distortion version of our test datasets for each distortion type and tested a pre-trained ResNet-18 model against each dataset. In several instances we performed this process iteratively in order to find the distortion level required to drive pre-trained model perfomance to chance. Figure 4.5 depicts the results of this process. Table 4.2 shows the distortion levels selected, with the overall *train* distortion space chosen to encompass completely undistorted images (except for conversion to grayscale) through images with resolution, blur, and noise levels shown to *independently* drive pre-trained model performance to chance or near-chance level.

After training as well as testing across the full *train* distortion range, we identified two aspects of the full train *train* distortion space that degraded the ability of our performance prediction models to fit our distortion performance results. First, we found that Places365 models trained over the full distortion space and further tuned on the most extreme corner were unable to surpass chance performance on test images in this extreme corner of the distortion space. Our performance prediction model fits suffered due to the lack of performance variation as a function of distortion level at the extreme of the distortion space. After examining the distortion levels that drove full-range trained models to chance performance, we reduced our Places365 *test* distortion bounds, with minimum resolution increasing from 0.1 to 0.2, maximum blur Gaussian standard deviation decreasing from 5 to




(a) SAT-6




(b) Places365

Figure 4.5: Measurements of pre-trained ResNet-18 model accuracy used to establish our distortion space. Specifically, we measured accuracy as a function of a *single* distortion type to find the distortion levels that independently drove performance to approach chance. MPC accuracy refers to mean per-class accuracy.

4.5 pixels (with kernel size remaining at 31 pixels across), and maximum noise decreasing from a stand deviation of 50 DN to 44 DN.

Second, we found that our performance prediction models returned poor fits for low blur levels. Notably, classification performance of our CNNs was largely flat at low blur levels. In examining RER as a function of blur (Chapter 3), we observed that RER remained virtually constant as a function of blur kernel standard deviation for $\sigma_{blur} <$ 0.5 pixels, as shown in Fig. 4.6. This lack of change in RER with blur kernel standard deviation occurs because the Gaussian blur kernel drops to near-zero outside of the center pixel for low at low $\sigma_{blur}$ (discussed in Sec. 3.3.1). Accordingly, we used a *minimum* blur kernel standard deviation of 0.5 pixels for both the SAT-6 and Places365 test datasets, ensuring that variation in $\sigma_{blur}$ resulted in variation in the resultant RER and image quality.

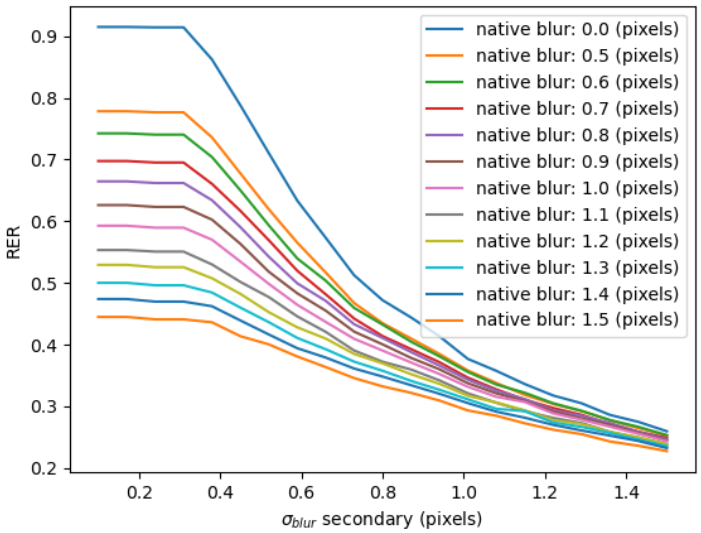



Figure 4.6: Measured RER as a function of secondary blur for varied native blur values

## 4.3 Results

### 4.3.1 Performance loss and recovery

As expected, the performance of models trained only on high quality images ("pre-trained" models in the vernacular of this analysis) declined rapidly with decreasing image quality. Models tuned across the full distortion space, however, proved far more robust to distortions and recovered much of the performance loss exhibited by pre-trained models tested on low-quality images.

Figure 4.7 illustrates the performance loss with blur and noise of a pre-trained model and the performance recovery achieved by tuning the model across the full range of distortions. The model tuned over the full distortion range still loses accuracy with increasing blur and noise, but the decrease is far better behaved, and the model still manages to achieve qualitatively reasonable top-1 accuracy in the presence of *significant* blur and noise. Additionally, it is important to note that for all of the test results shown in this analysis, resolution, blur, and noise distortions are all varying *simultaneously*. Accordingly, the accuracy as a function of blur and noise shown in Fig. 4.7 represents the mean accuracy across the full range of resolution values, with the average resolution in the test

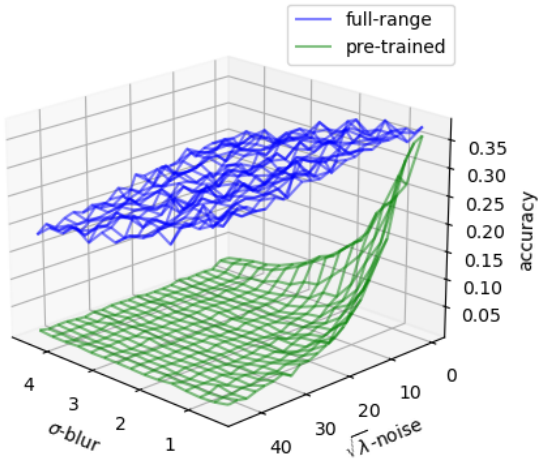
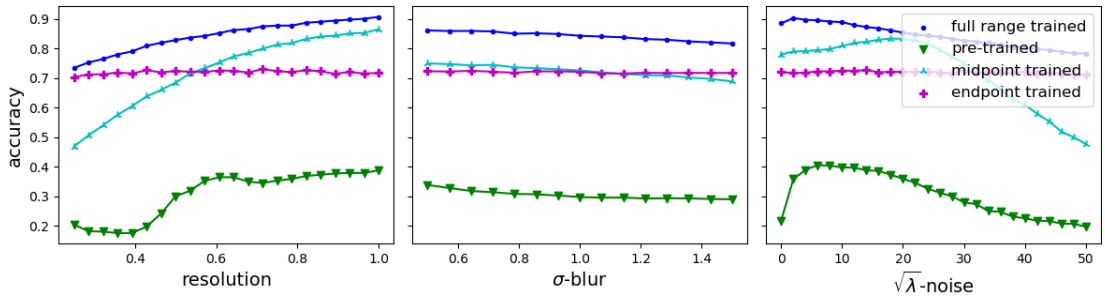



Figure 4.7: Mean accuracy as a function of blur and noise for a ResNet-18 model pre-trained on high quality images and a ResNet-18 tuned across the full distortion range of the Places365 test dataset.

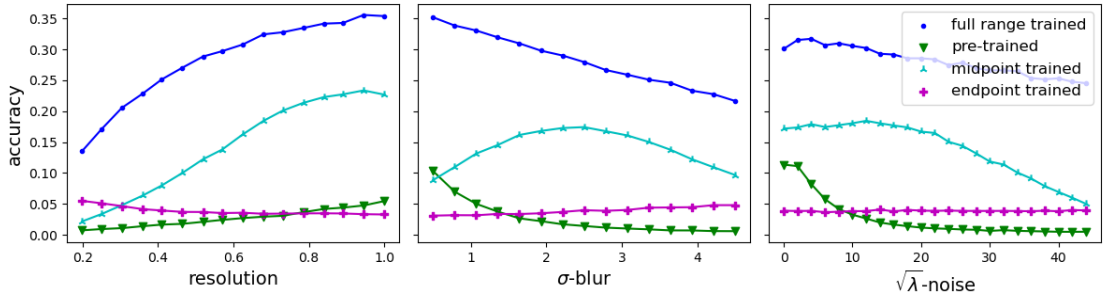
dataset being roughly 60% that of the original undistorted images.

Figure 4.8 illustrates mean accuracy as a function of a single variable for four models on Places365 and SAT-6: a full range model tuned across the full distortion range, a pre-trained model trained only on high quality images, a midpoint model tuned on images with the mean distortion value in each distortion axis, and an endpoint model tuned on images with the extreme distortion value in each distortion axis. As noted above, the test images used were each subjected to all three distortions simultaneously, with accuracy values therefore reflecting the average across the remaining two distortion dimensions.

These plots exhibit several characteristics worth noting. First, the full range trained models consistently outperform the point models, although this performance offset arises because we are taking the mean over two of the three distortion dimensions in each plot. Figure 4.9 shows the slight advantage *at a particular distortion point* of models tuned on identically distorted images. Second, the midpoint models and endpoint models both prefer distortion levels nearer those of their tuning images, particularly for Places365, with performance *improving* for *increased distortion* in many regions of the distortion space. Third, both full range trained models and the SAT-6 pre-trained model all show at least a slight performance improvement with modest added noise. We believe this noise preference results from the added noise compensating for model over-fitting. Finally,




(a) SAT-6

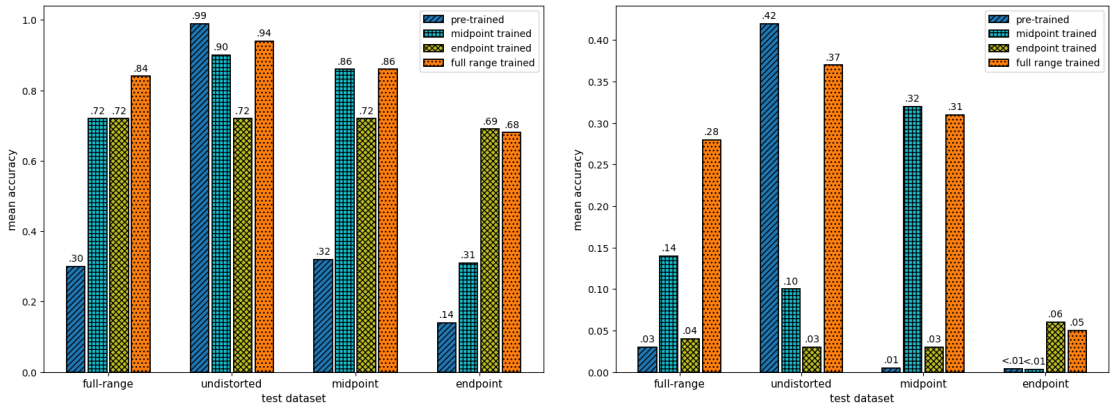



(b) Places365

Figure 4.8: SAT-6 and Places365 classification accuracy as a function of resolution, blur, and noise for ResNet-18 models pre-trained on undistorted images as well tuned at the midpoint, endpoint, or full range of the distortion space for each dataset.

while point model behaviors differ at points between Places365 and SAT-6 (*e.g.*, the SAT-6 pre-trained model performance varies erratically as a function of resolution whereas the Places365 pre-trained model does not exhibit this behavior), the behavior of the full range trained models is very similar between the two datasets.

We observe these trends from a different perspective in Figure 4.9, which depicts the mean accuracy of SAT-6 and Places365 models trained and tested on undistorted images, images at the distortion space midpoint, images at the distortion space endpoint, and images across the full distortion range. Specifically, we see that the full-range models perform better on average than any other model on the full-range dataset. With the exception of the SAT-6 midpoint dataset where the full range and midpoint models achieve the same accuracy, these full range models approach but do not match the performance of the point models on their matching point-distortion datasets. For SAT-6, and endpoint




(a) SAT-6 (b) Places365

Figure 4.9: Performance of pre-trained, full-range, midpoint and endpoint SAT-6 models on full distortion range, undistorted, distortion space midpoint, and distortion space endpoint datasets.

trained model achieves stable performance across the all of the test datasets, while the midpoint model performs well on all except the endpoint dataset. The peak performance of the Places365 midpoint and endpoint models occurs on the midpoint and endpoint datasets respectively, where these models achieve at least twice their respective performance on undistorted images. Finally, while 5- and 6-percent accuracy of the Places365 full-range and endpoint models on the endpoint dataset may be unimpressive in isolation, it is worth noting that images with these distortion levels are all but uninterruptible visually (see Figure 4.2), with the models performing at roughly a factor of twenty above chance against our class-balanced test dataset containing 100 images per label label.

### 4.3.2 Model architecture comparison

To understand the generality of our results across different model architectures, we trained and tested ResNet-18, ResNet-50, and DenseNet-161 models on identical datasets and compared the results. While the larger, more computationally intensive DenseNet-161 model performed better, particularly on the Places365 dataset, variation in performance as a function of resolution, blur, and noise were effectively identical across models, as shown in Figures 4.10 and 4.11. The cross-correlation between the mean performance as a function of resolution, blur, and noise for each pair of these models was at least 0.92. For all of our subsequent analyses, we use ResNet-18 models to take advantage of their

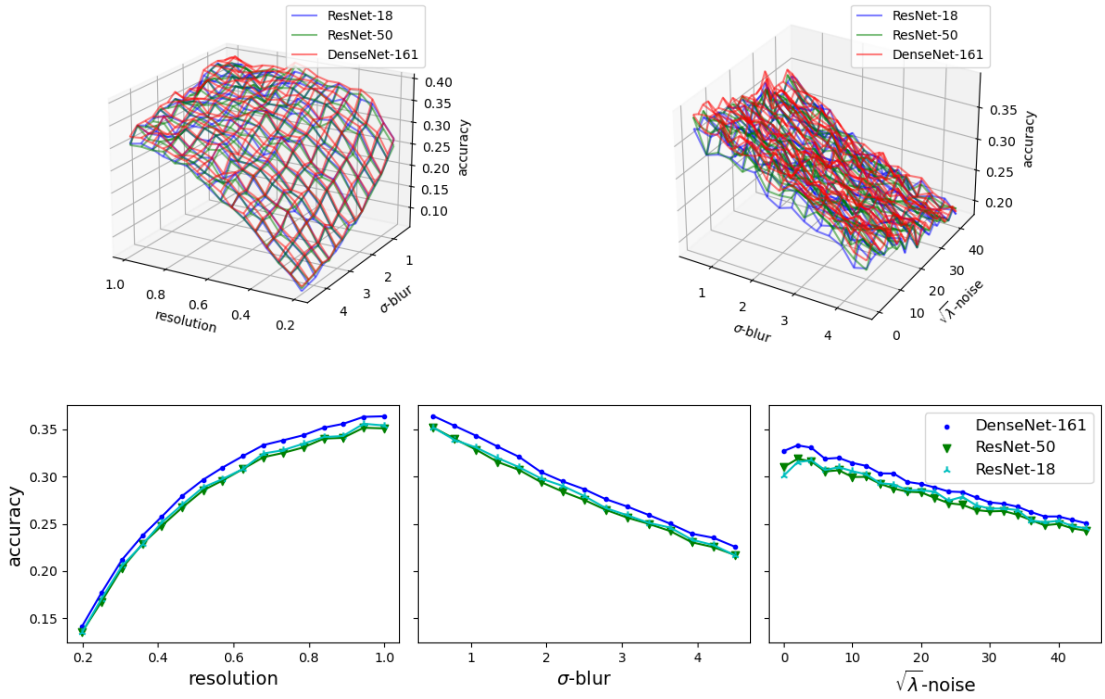



Figure 4.10: Places365 performance variation with resolution, blur, and noise for ResNet-18, ResNet-50, and DenseNet-161 models trained and tested across the full range of the Places365 distortion space.

lower computational overhead. While using a deeper architecture such as a DenseNet-161 would improve absolute performance, we emphasize that our goal is to understand the broad contours of distortion-performance relationships rather than to push computer vision performance benchmarks.

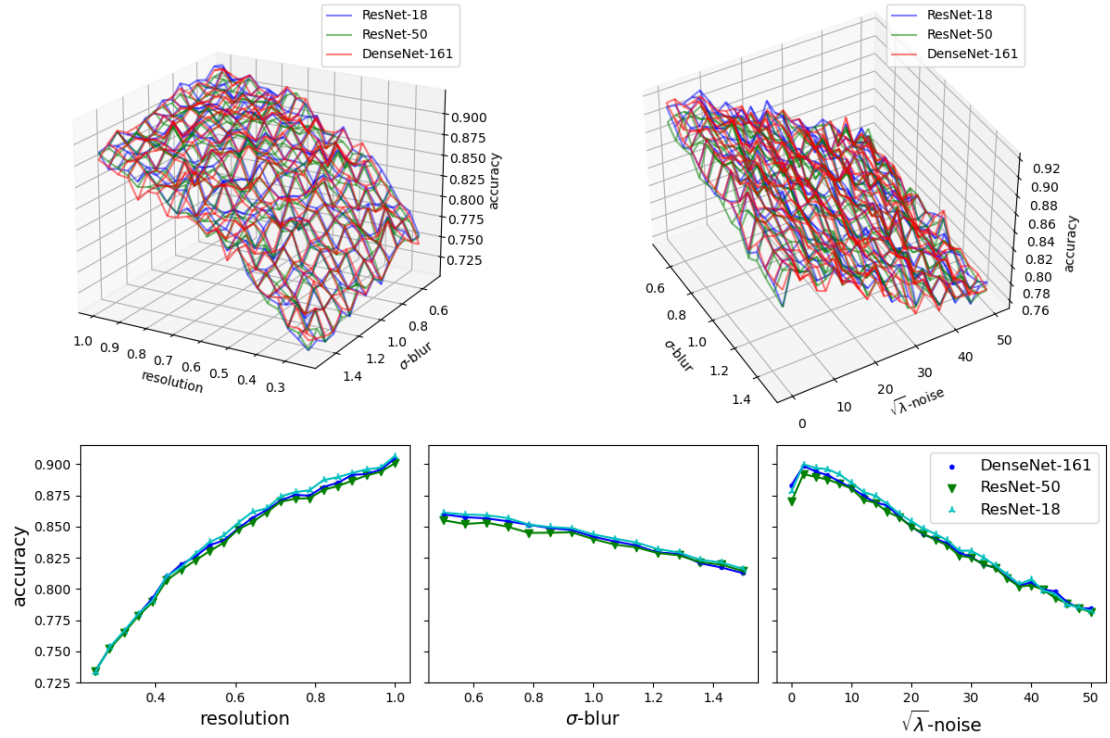



Figure 4.11: SAT-6 performance variation with resolution, blur, and noise for ResNet-18, ResNet-50, and DenseNet-161 models trained and tested across the full range of the Places365 distortion space.

### 4.3.3 Composite performance results

To better approximate the performance of ideally trained CNNs–namely models specifically tuned to maximize performance at each distortion point individually–we constructed composite performance results using the octant models discussed in Section 4.1. Specifically, we used two i.i.d. versions of our test dataset to construct our composite performance results. We used the first dataset to identify the best performing model in each distortion octant (with the assumption that each model would be the best performing in its respective octant). We then filtered our models' predictions on the second dataset; for each octant in the second dataset, we used the predictions from the model which performed best against that same octant in the first test dataset. As expected, taking the best performing model from each octant in the first test dataset and using it for the same octant in

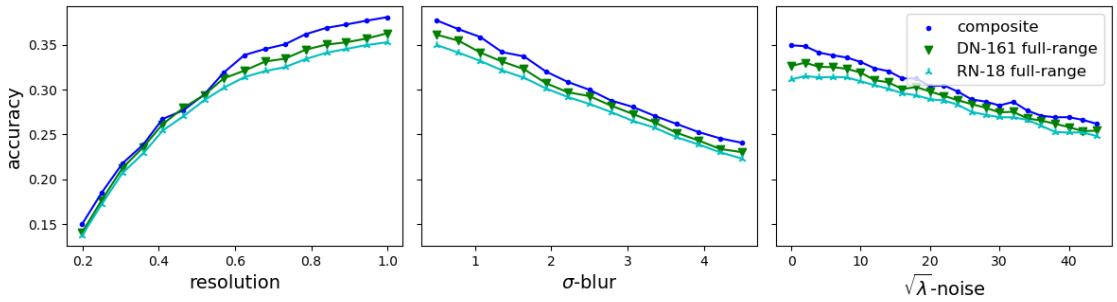



Figure 4.12: Comparison of ResNet-18 octant model composite performance to performance of full-range ResNet-18 and DenseNet-161 models.

the second test dataset resulted in a modest performance improvement compared to using a single model trained across the full range of the distortion space. All of the remaining test results shown here are from our the second version of our Places365 and SAT-6 test datasets. We used the first version of these test datasets to compare performance as a function of training image distortion in Sec. 4.3.1 and to compare model architectures in Sec. 4.3.2. In the remainder of the study, we use the first version of our test datasets to select the octant models and to fit performance predictions, and we use the second version of our test datasets to evaluate composite performance and validate performance predictions.

Table 4.3: Mean accuracy of full range trained models and octant composite result on i.i.d version 2 of our full range test datasets

| | Places365 | SAT-6 |
|---|---|---|
| ResNet-18 full range trained | 0.285 | 0.842 |
| ResNet-150 full range trained | 0.281 | 0.838 |
| DenseNet-161 full range trained | 0.292 | 0.840 |
| octant composite result | 0.303 | 0.855 |

Figure 4.12 compares an octant model composite result with the performance of full range trained ResNet-18 and DenseNet-161 models on the Places365 full range test dataset, and Table 4.3 summarizes the mean accuracy of all full range trained models and the composite performance results for both Places365 and SAT6. Here, we see that the ResNet-

18 octant models' composite performance result achieved a top-1 accuracy improvement of approximately 1.8% over a single ResNet-18 model tuned across the full distortion range (30.3% vs. 28.5% top-1 accuracy). The performance difference between the composite result and a full range trained DN-161 model was even smaller (30.3% vs. 29.2%), with the most pronounced performance improvement occurring for high quality/low distortion images, a result in keeping with the point model performance results shown in Fig. 4.9. We note, however, that the composite performance result does not suffer the performance drop seen in full range trained models moving from modest added noise to zero added noise, as shown in the plot of accuracy as a function of noise in Fig. 4.12.

## 4.4 Application of the GIQE to Computer Vision Performance

Having observed the overarching performance patterns associated with training and testing CNN-based classifiers on distorted images, we explored whether the functional form of the GIQE could form the basis of a model predicting CNN performance as a function of image quality.

Fundamentally, the GIQE maps empirical image quality parameters to a man-made utility metric (NIIRS) capturing the types of tasks that a human analyst can perform with an image of a given rating. The GIQE and NIIRS metric were designed to provide intuitively satisfying relationships (such as requiring $\Delta\text{NIIRS} = 1$ for a doubling of resolution [14]), and the two co-evolved in a feedback loop in which analysts learned to predict NIIRS ratings from the GIQE and the GIQE was tweaked to maximize the agreement between analyst scores and GIQE-predicted NIIRS ratings. NIIRS is both a useful monotonic "goodness" metric and a valuable communication tool, but it is designed around the performance of the *human visual system*. While the GIQE's applicability to computer vision problems is plausible, its utility for predicting the classification accuracy of CNN on imagery from a given imaging system is not obvious *a priori*.

The GIQE predicts image quality based on GSD, RER, and SNR, none of which we measured on our images. We can, however, map the distortion variables we used to these quantities. We know by inspection that for resolution fraction $r$, ground sample distance GSD is given by

$$\text{GSD} \propto \frac{1}{r} \tag{4.2}$$

and that

$$\text{SNR} \propto \frac{1}{\sqrt{n_0^2 + n_1^2}}, \tag{4.3}$$

$n_0$ represents the native noise of our images before distortion, and $n_1$ represents the noise added in our distortion process. RER captures the sharpness of an image by quantifying the slope of its edge response function, which is the spatial derivative of an image in the direction normal to an edge. For an image formed with a Gaussian PSF of standard deviation $\sigma$,

$$\text{RER} \approx \frac{1}{\sigma\sqrt{2\pi}}. \tag{4.4}$$

Here, we are studying images with native blur from the original system PSF that we have blurred with a secondary Gaussian kernel of standard deviation $\sigma_1$. If we approximate the original system PSF as a Gaussian of standard deviation $\sigma_0$, we find that

$$\text{RER} \approx \frac{1}{\sqrt{2\pi\left(\sigma_0^2 + \sigma_1^2\right)}}, \tag{4.5}$$

when $\sigma_0^2 + \sigma_1^2 \gg \frac{1}{2\pi}$. Chapter 3 presents a full derivation and validation of equations 4.4 and 4.5.

### 4.4.1 Native noise estimation

In order to account for the native noise in our images, we created a simple model to estimate the raw noise in our images in units of counts/digital number (DN) after conversion to 8-bit images. Estimating the true SNR of an image that does not contain calibration targets is a non-trivial problem [16]; we made no such attempt. Given the large amounts of additive noise applied in our distortion process (up to 50 counts in an 8-bit image with only 256 gray levels), we need only a very rough estimate of the mean noise content of our images before application of our distortions.

Here, we define mean noise as the standard deviation of the *difference* between light and dark image patches, where the light patch is illuminated to 50% of well depth before accounting for dark electrons and the dark patch has no illumination, with all electrons in the dark patch coming from dark current and read noise. Algorithm 1 shows the process for modeling the noise of an image with a given saturation fraction, well depth, mean dark electron count, read noise, and bit depth using synthetic $m \times m$-pixel light and dark image patches. As with the images used throughout this study, we begin with RGB images which we then convert to grayscale by taking a simple average across the RGB channels.

**Algorithm 1** Estimation of the the raw noise and SNR of a grayscale image

**Input:** Saturation fraction $s$, well depth in electrons $d$, mean dark electron count $\lambda_{dark}$, read noise $\sigma_{read}$, bit depth $b$, patch size $m$, where the light and dark patches are each $m \times m$ pixels

**Output:** Grayscale SNR and estimate raw noise $n$

$e^-_{light}(i,j,k) \leftarrow s \times d + \lambda_{dark}, \quad i,j \in \{1,...,m\}$ and $k \in \{1,2,3\}$

$e^-_{light} \leftarrow \text{Poisson}\left(e^-_{light}\right)$

$e^-_{light} \leftarrow e^-_{light} + \mathcal{N}\left(0, \sigma^2_{read}\right)$

$S_{light} \leftarrow e^-_{light} \times \frac{2^b}{d}$

$S_{light}(i,j) = \frac{1}{3}\sum_{k=1}^{3} S_{light}(i,j,k)$

$e^-_{dark}(i,j) \leftarrow \lambda_{dark},\ i,j \in \{1,...,m\}$

$e^-_{dark} \leftarrow \text{Poisson}\left(e^-_{dark}\right)$

$e^-_{dark} \leftarrow e^-_{dark} + \mathcal{N}\left(0, \sigma^2_{read}\right)$

$S_{dark} \leftarrow e^-_{dark} \times \frac{2^b}{d}$

$S_{dark}(i,j) = \frac{1}{3}\sum_{k=1}^{3} S_{dark}(i,j,k)$

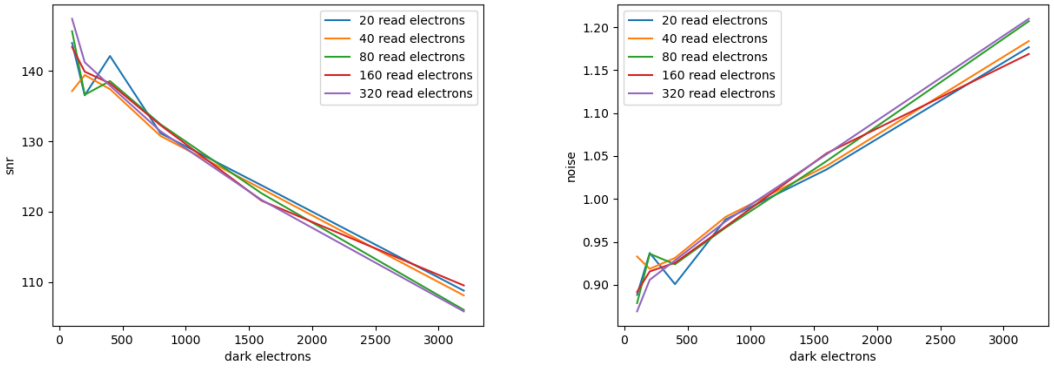
$S_{total} \leftarrow S_{light} - S_{dark}$

$n \leftarrow \text{Std}(S_{total})$

$\text{SNR} \leftarrow \frac{E[S_{total}]}{n}$

**return** SNR, $n$

Using the method shown in Algorithm 1, we estimated the raw noise and SNR of 8-bit grayscale images over a range of read noise and dark current values, where well depth was fixed at 15-thousand electrons and light and dark patches were $64 \times 64$ pixels, with light patch illumination set to 50% of well depth. Figure 4.13 shows that total estimated noise for these conditions. Based on these results, we chose to set our native noise estimate $n_0$ to one count in our performance prediction models using equation 4.6 and 4.8.




(a) Estimate grayscale SNR

(b) Estimate gray scale raw noise

Figure 4.13: Estimated SNR (left) and raw noise (right) as a function of dark electron count for varied read noise values. Across the set of dark electron and read noise values, total noise falls between roughly 0.9 and 1.2 counts in an 8-bit RGB image that has been converted to grayscale.

### 4.4.2 GIQE-based performance prediction

Having mapped our distortion variables to GIQE terms in equations 4.2, 4.3, and 4.5, we substitute these mappings into the GIQE given in 1.1 in order to fit our performance results to a GIQE-based performance prediction model. Specifically, we fit equation 4.6 to our accuracy data:

$$\begin{aligned}\bar{a}\,(r,\sigma,n) \;=\;& c_0 + c_1 \log_{10}\left(\frac{1}{r}\right) \\ &+\; c_2\left(1-\exp\left(c_3\sqrt{n_0^2+n_1^2}\right)\right)\cdot\log_{10}\left(\frac{1}{\sqrt{2\pi\left((c_4 r)^2+\sigma^2\right)}}\right) \\ &+\; c_5\left(\log_{10}\left(\frac{1}{\sqrt{2\pi\left((c_4 r)^2+\sigma^2\right)}}\right)\right)^4 + c_6\sqrt{n_0^2+n_1^2},\end{aligned} \tag{4.6}$$

where we set $n_0 = 1$, which we estimate to be the native noise in counts of a typical 8-bit image that has been converted to grayscale (see Sec. 4.4.1), and $c_4 r$ represents the

native blur $\sigma_0$ of the images scaled by the resolution to account for the sharpening effect of down-sampling. We used a least squares routine for all fitting.

To understand whether the cross term was important or represented an over-parameterization of the performance prediction model, we fit an equation based instead on the simpler, historical functional form of the GIQE which did not include the RER-SNR cross term and did not raise the independent RER term to fourth power: [14]

$$\text{NIIRS} = C_0 + C_1 \log_{10}(\text{GSD}) + C_2 \log_{10}(\text{RER}) + C_3 \frac{G}{\text{SNR}} + C_4 H_{GM}, \tag{4.7}$$

where $G$ represents noise gain due to edge sharpening and $H_{GM}$ represents height of overshoot due to edge sharpening, with coefficients values shown in Tab. 4.4.

Table 4.4: GIQE version versions 3 and 4 coefficients

| GIQE Version | $C_0$ | $C_1$ | $C_2$ | $C_3$ | $C_4$ |
|---|---|---|---|---|---|
| v3 | 11.81 | -3.32 | -3.32 | -1 | -1.48 |
| v4 with RER $> 0.9$ | 10.25 | -3.32 | 1.559 | -0.344 | -0.656 |
| v4 with RER $< 0.9$ | 10.25 | -3.16 | 2.817 | -0.334 | -0.656 |

Our GIQE-v3 model (-v3 rather than -v4 since we do not handle RER piece-wise) took the form

$$\bar{a}(r, \sigma, n) = c_0 + c_1 \log_{10}\left(\frac{1}{r}\right) + c_5 \log_{10}\left(\frac{1}{\sqrt{2\pi\left((c_4 r)^2 + \sigma^2\right)}}\right) + c_6\sqrt{n_0^2 + n_1^2}, \tag{4.8}$$

with the final term of the historical GIQE ($C_4 H_{GM}$) dropping out since we do not apply edge sharping to our images. The coefficients for both fits are shown in Table 4.5.

The surface plots in Fig. 4.14 show the predicted and actual accuracy that result when we fit our performance prediction model (Eq. 4.8) to accuracy as a function of distortion on version 1 of our Places365 test datasets and compared the fit to our test results on version 2 of our test datasets. In all cases, we have used the octant models and composite performance process described in Sec. 4.3.3. As with our other surface plots, we generated these plots by averaging out one of our three distortion dimensions to display average accuracy as a function of the remaining two. Visually, these plots suggest that our performance prediction model captures underlying accuracy reasonably well.

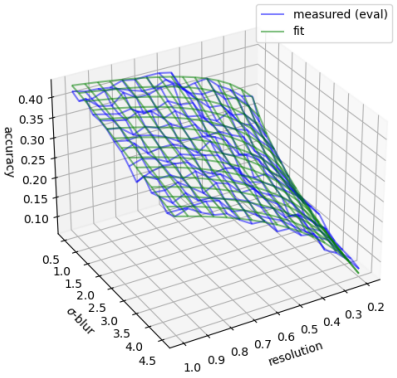
As a check on the visual test results shown in Fig. 4.14, we used predicted accuracy values from our Eq. 4.8 fits as inputs to a simple binomial distribution accuracy simulation. The success or failure of an image classifier on each test image represents a simple Bernouli

Table 4.5: Performance prediction fit coefficients from GIQE-5 based model (Eqn. 4.6) and GIQE-3 based model (Eqn. 4.8) for Places365(†) and SAT-6 (♦)

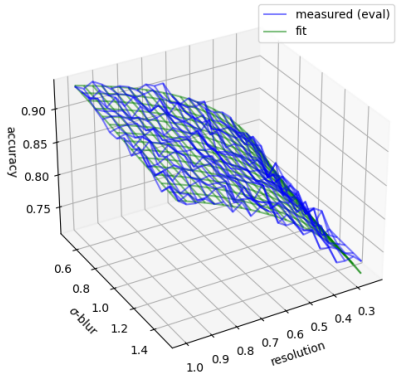
| | $c_0$ | $c_1$ | $c_2$ | $c_3$ | $c_4$ | $c_5$ | $c_6$ |
|---|---|---|---|---|---|---|---|
| GIQE-5♦ | 1.02 | −0.299 | 0.0453 | −0.113 | −0.0221 | −0.278 | $-2.81 \times 10^{-3}$ |
| GIQE-3♦ | 1.08 | −0.336 | - | - | 0.935 | 0.152 | $-3.05 \times 10^{-3}$ |
| GIQE-5† | 0.512 | −0.385 | 0.139 | −0.0319 | 1.88 | −0.0841 | $-2.25 \times 10^{-4}$ |
| GIQE-3† | 0.688 | −0.417 | - | - | 2.25 | 0.272 | $-2.10 \times 10^{-3}$ |

trial, so in aggregate we can treat a series of test images at a given distortion point $(r, \sigma, n)$ as a binomial experiment, where $P_{success}$ is the mean accuracy of our model $\hat{a}\,(r, \sigma, n)$ and the number of trials is the total number of images subjected to those particular distortions. For our simulation, we set the number of trials at each distortion point to the total number of test images in our test dataset divided by the number of distortion points, and we simulated the resulting performance with $P_{success}$ set to our performance prediction.

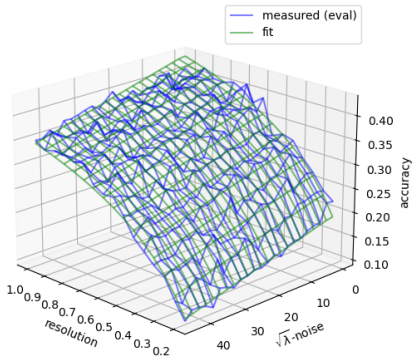
Figure 4.15 compares measured vs. predicted accuracy (left) and simulated vs. predicted accuracy (right) for our SAT-6 and Places365 fits. For both datasets, we observe very similar results from both test and simulation. We note that accuracy quantization levels in simulated plots appear because all simulated binomial experiments use the same number of trials, leading to discrete accuracy levels spaced by $1/n_{trials}$. This quantization does not appear in the measured data, except where accuracy = 1.0, because distortions are applied stochastically, so the number of images at each distortion point varies. Where accuracy = 1.0, changes in the denominator have no effect, leading to quantization step observed in the SAT-6 measured data. Here, we see that $r^2$ values and visual appearance of our actual and simulated results are very similar. Significantly, when we perform a simple linear fit of predicted vs. actual accuracy and predicted vs. simulated accuracy, we observe $r^2$ values 0.815 and 0.843 respectively on Places365, suggesting that much of the variability in the test results represents the inherent variability of the underlying binomial distribution. These results indicate that the functional form of the GIQE works reasonably well for modeling CNN accuracy as a function of image quality, within the constraints of this study.



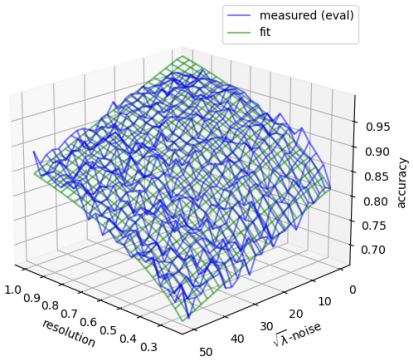

(a) Resolution and blur (Places365)



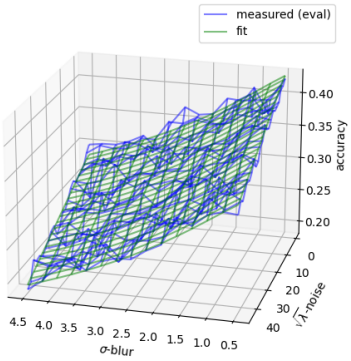

(b) Resolution and blur (SAT-6)



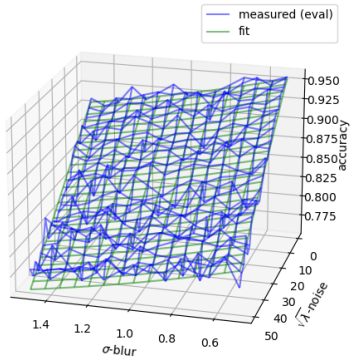

(c) Resolution and noise (Places365)




(d) Resolution and noise (SAT-6)

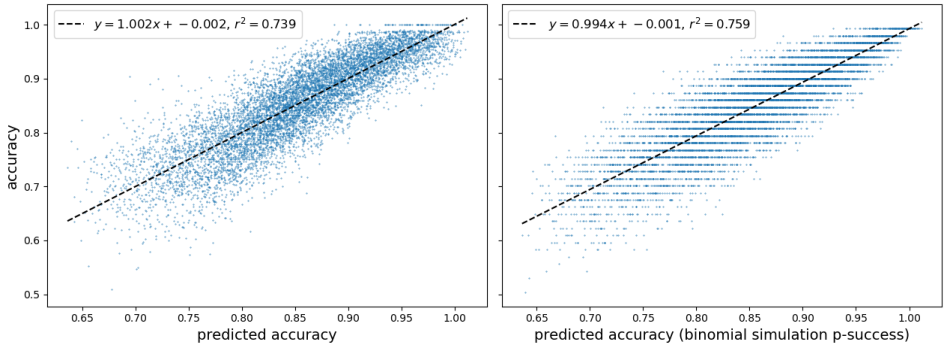



(e) Blur and noise (Places365)

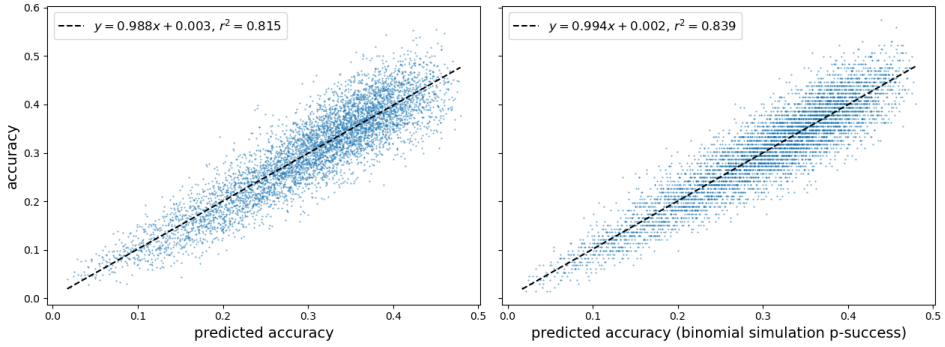



(f) Blur and noise (SAT-6)

Figure 4.14: Predicted and measured accuracy for Places365 and SAT-6 using our GIQE-3 model (Eq. 4.8). The performance prediction model was fit using test results on version 1 of our two i.i.d. test datasets, and with the fit applied and evaluated on version 2 of our i.i.d. test datasets.




(a) SAT-6




(b) Places365

Figure 4.15: Scatter plots of predicted vs. actual accuracy (left) and an equivalent scatter plot (right) showing the resulting accuracy when the predicted accuracies from fitting Eq. 4.8 are used inputs to a binomial probability distribution.

### 4.4.3 Performance prediction model comparison

Based on the results depicted in Figs. 4.14 and 4.15, we believe that that our GIQE-based models, as fitting functions, give *reasonably good approximations* of CNN accuracy over the distortion space. We recognize, however, that in many cases there are multiple non-linear functions that can approximate the same data. For comparison, therefore, we fit naive performance prediction models without any pedigree. Specifically, we fit a power law function of the form

$$\bar{a}\,(r,\sigma,n) = c_0 + c_1 r^{c_2} + c_3 \sigma^{c_4} + c_5 n^{c_6} \tag{4.9}$$

and an exponential function of the form

$$\bar{a}\,(r,\sigma,n) = c_0 + c_1 e^{c_2 r} + c_3 e^{c_4 \sigma} + c_5 e^{c_6 n}. \tag{4.10}$$

We found that these equations succeeded in fitting and predicting our performance data as well as, and in some respects better, than our GIQE-based fits. Table 4.6 summarizes the quality of the fits generated using each functional form. When we generate plots like those in Figs. 4.14 and 4.15, the visual differences are minimal across all of the models we used (not shown).

To better visualize the various fits, we next compared them across a single distortion dimension. For example, when comparing mean predicted and measured accuracy as a function of resolution, the mean accuracy at each resolution value represents the average over all blur and noise values. Figure 4.16 shows these results. For the most part our models are consistent in predicting average performance as a function of each distortion variable; the only notable exception is the Eqn. 4.6 model based on the latest version of the GIQE. Here, we see that our model's predicted accuracy as a function of resolution is concave down when it should be concave up. Figure 4.18, with the same data plotted on a slightly different scale, shows this concavity mismatch more clearly. The concavity arises from the exponent in the RER term, which changed from $\log_{10}(RER)$ in the historical versions of the GIQE to $\log_{10}(RER)^4$ in the current version. The exponent is intended to account for non-optimal edge sharpening, penalizing images with low RER that would typically require greater degrees of sharpening by an image analyst [14].

Based on these fit comparison plots (Figs. 4.16, 4.17 and 4.18), we observe that simple linear correlation metrics do a poor job of discriminating between models. By comparing $r^2$ values between measured and simulated data, we can conclude that much *but not all* of the variance between modeled and measured data arises due to the inherent characteristics of the underlying binomial distribution that describes a classification experiment. The systematically larger $r^2$ values in the simulation column suggest that our models imperfectly describe the underlying probability distribution; the relatively small size of this

Table 4.6: Measured vs. predicted and simulated vs. predicted fit metrics for performance predictions from our GIQE-5 model (Eqn. 4.6), GIQE-3 model (Eqn. 4.8), power law model (Eqn. 4.9), and exponential model (Eqn. 4.10) for SAT-6 (♦) and Places365 (†). *Measured* metrics (left) reflect result from a linear fit ($y = mx + b$ with coefficient of determination $r^2$) of measured accuracy $y$ vs. predicted accuracy $x$. *Simulated* metrics (right) result reflect a linear fit of simulated accuracy $y$ vs. predicted accuracy $x$, where our simulated accuracy values were generated by simulating the results of binomial experiment in which $P_{success}$ is given by predicted accuracy and the number of trials is the average number of images at each distortion point ($\sim 80$).

| | Measured data | | | Simulated data | | |
|---|---|---|---|---|---|---|
| | $m$ | $b$ | $r^2$ | $m$ | $b$ | $r^2$ |
| GIQE-5♦ | 1.002 | −0.002 | 0.739 | 1.002 | −0.007 | 0.765 |
| GIQE-3♦ | 1.002 | −0.002 | 0.739 | 0.993 | 0.0 | 0.762 |
| Power law♦ | 1.003 | −0.002 | 0.740 | 0.998 | −0.004 | 0.761 |
| Exponential♦ | 1.003 | −0.002 | 0.739 | 0.991 | 0.002 | 0.761 |
| GIQE-5† | 0.988 | 0.003 | 0.815 | 0.998 | 0.0 | 0.845 |
| GIQE-3† | 0.988 | 0.003 | 0.815 | 0.988 | 0.002 | 0.841 |
| Power law† | 0.989 | 0.003 | 0.798 | 1.006 | −0.004 | 0.837 |
| Exponential† | 0.989 | 0.002 | 0.801 | 1.002 | −0.001 | 0.837 |

difference, however, suggests that the gap between the modeled and the actual probability distributions is minimal.

We therefore apply two additional tools to compare our performance prediction models, with the goal of understanding whether we can quantitatively discriminate between these models' applicability. First, we consider the Durbin-Watson statistic, which measures the autocorrelation between between adjacent residuals, and is given by

$$d = \frac{\sum_{t=2}^{T} (e_t - e_{t-1})^2}{\sum_{t=1}^{T} e_t^2}, \tag{4.11}$$

where $e_t$ is the residual at each of $T$ total samples [104]. Values of the Dubin-Watson statistic fall between 0 and 4, with $d \approx 2(1-r)$ for autocorrelation $r$ given a reasonably large number of samples [105]. In the presence of underfitting or overfitting, we would expect to observe positive correlation between adjacent residuals in the first case and negative correlation between residuals in the second. Conversely, with a good model we should observe very little correlation between adjacent residuals, leading to Durbin-Watson values near two.

The Durbin-Watson statistic is usually applied to time series models where there is no question about the adjacency of residuals. Here, we are predicting classifier performance as a three dimensional function of three independent distortions, giving each residual $e_{i,j,k}$ three immediate predecessors at $e_{i-1,j,k}$, $e_{i,j-1,k}$, and $e_{i,j,k-1}$. Accordingly, we calculated the Durbin-Watson statistics several different ways. First, we considered the residuals as a function of all three distortion variables and determined adjacency by unraveling the data cube along one of the three dimensions at a time. Formally, for a residual tensor $e(i,j,k)$ of shape $(l,m,n)$, unraveling along the $i$-axis leads to sequential residuals $e_t$, with

$$e_t = e(i,0,0) \frown e(i,1,0) \ldots e(i,m-1,0) \ldots e(i,m-1,n-1), \tag{4.12}$$

where the $\frown$ operator represents vector concatenation such that $(x_0, x_1) \frown (y_0, y_1) = (x_0, x_1, y_0, y_1)$. For each of our performance prediction models, we began by calculating the Durbin-Watson statistic for unraveling along each of the three distortion axes.

Next , we calculated the Durbin-Watson statistics after averaging out two of the three distortion dimensions, analyzing predicted and measured accuracy as a function of the remaining distortion variable (the same process we used to generate all of our one dimensional performance plots, *e.g.*, Fig. 4.16). Having reduced the data to a single dimension here, we calculated the Durbin-Watson statistic directly.

Figures 4.19 and 4.20 show the Durbin-Watson statics for each of the four models considered on SAT-6 and Places365 respectively. From these figures, we can see that all of the models show at least some correlation between adjacent residuals, with the

autocorrelation growing (and Durbin-Watson scores therefore dropping further below two) when we drop to one dimension. Collectively, these results suggest that all of the models are at least slightly underfit and do not fully account for the variability in the data, although none of the 3d statistics fall outside of the range generally considered reasonable. The 1d statistics largely align with what we see in Figs. 4.17 and 4.16, where we can see clearly that the GIQE-5 based model does a poor job of predicting accuracy as a function of blur.

In addition to comparing Durbin-Watson statistics, we also used the Akaike information criterion (AIC) to compare our models to one another. AIC quantifies the "goodness" of a fit by quantifying the likelihood that a particular model describes the behavior of a system, with AIC given by

$$AIC = -2\ln\left(\mathcal{L}\left(\hat{\theta}_{MLE}|\mathbf{y}\right)\right) + 2\kappa_\theta, \tag{4.13}$$

where $\mathcal{L}\left(\hat{\theta}_{MLE}|\mathbf{y}\right)$ is the likelihood of the estimator $\hat{\theta}_{\mathbf{MLE}}$ given the set of observations $\mathbf{y}$, and $\kappa_\theta$ is the number of parameters in the estimator [106]. AIC scores *decrease* as fit fidelity increases (the opposite of the Durban-Watson statistic), and the $\kappa_\theta$ term penalizes models with greater numbers of parameters to account for overfitting.

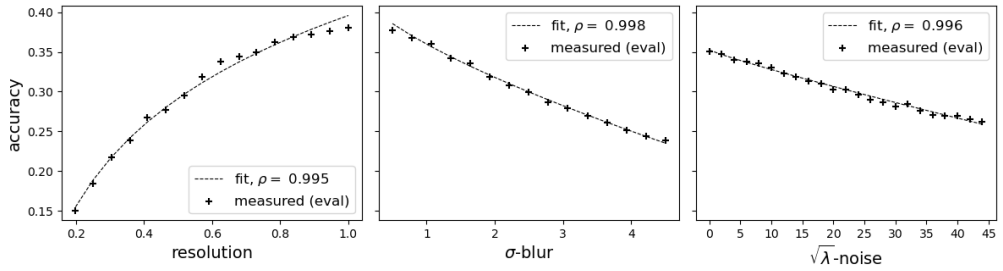
In practice, we can calculate $\mathcal{L}\left(\hat{\theta}_{MLE}|\mathbf{y}\right)$ by taking the product $\prod_i p\left(y_i|\,\hat{\theta}_{MLE}\right)$, where $p\left(y_i|\,\hat{\theta}_{MLE}\right)$ is the probability of each observation $y_i$ given the value predicted by $\hat{\theta}_{MLE}$. Since the measured top-1 accuracy at a given distortion point follows a binomial distribution (discussed in Sec. 4.4.2), we used this distribution to model the probability of each observation $p\left(y_i|\,\hat{\theta}_{MLE}\right)$. Specifically, we calculated the probability of observing accuracy $y_i$ at each distortion point, with $P_{success}$ for the binomial distribution at the distortion point given by the predicted accuracy $\bar{a}_i$, where each $\bar{a}_i$ came from fitting one of our four performance prediction models. Each model's fit parameters were then that model's maximum likelihood estimator, $\hat{\theta}_{MLE}$.

Table 4.7 shows each performance prediction model's AIC scores on our two datasets. AIC scores *by themselves* are not particularly meaningful since they are directly dependent on the number of observations and the distribution of the data, *i.e.* there are no general "rules of thumb" for good or bad AIC scores. Instead, AIC scores are useful for comparing multiple models on a single set of observations, and accordingly we report $\Delta AIC$ for the models in consideration on each dataset, where the $\Delta AIC$ value for each model is simply that model's AIC score minus the best (lowest) AIC score among the models compared. When we compare the AIC scores of our models, we find that the GIQE-3 based model performs best for both SAT-6 and Places365, with the exponential model yielding the next best AIC scores in both instances.

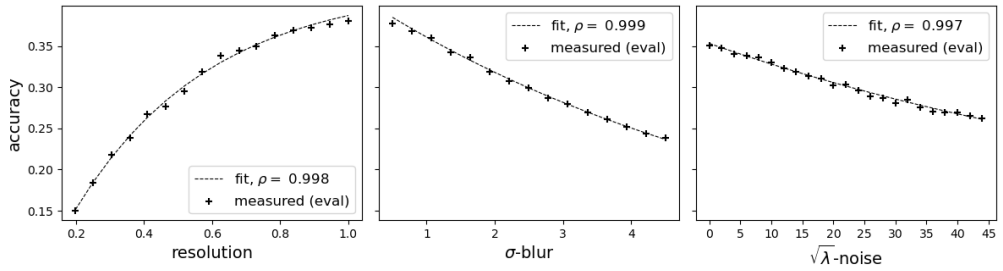
Table 4.7: Performance prediction model Akaike information criterion (AIC) scores for SAT-6 and Places365.

| | SAT-6 | | Places365 | |
|---|---|---|---|---|
| | $AIC$ | $\Delta AIC$ | $AIC$ | $\Delta AIC$ |
| GIQE-5 | 45286.8 | 816.3 | 35227.9 | 627.2 |
| GIQE-3 | 44470.5 | 0 | 34600.7 | 0 |
| Power law | 45316.2 | 845.7 | 35171.1 | 570.5 |
| Exponential | 45034.8 | 564.3 | 35042.7 | 442.0 |

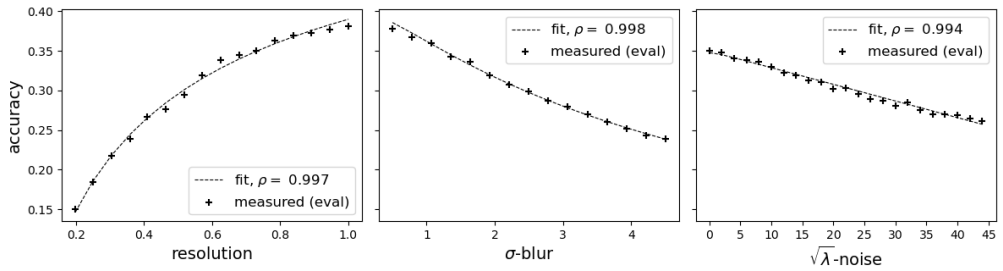
Finally, we highlight two key factors in evaluating and comparing our performance prediction models. First, the GIQE-3 model has the fewest fit parameters of those that we considered (a factor that is accounted for in our AIC scores.) Second, this model is linear with respect to all fit parameters except for $c_4$, which accounts for the native blur of our images in our RER approximation, while our other three models are non-linear with respect to at least two fit parameters. Based on these factors and the strengths of this model's fits discussed above, we consider the performance prediction model based on version 3 of the GIQE to be the best of those we considered.



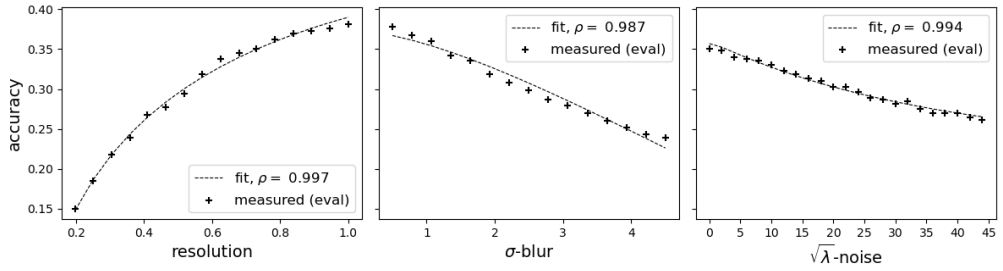

(a) Power law




(b) Exponential

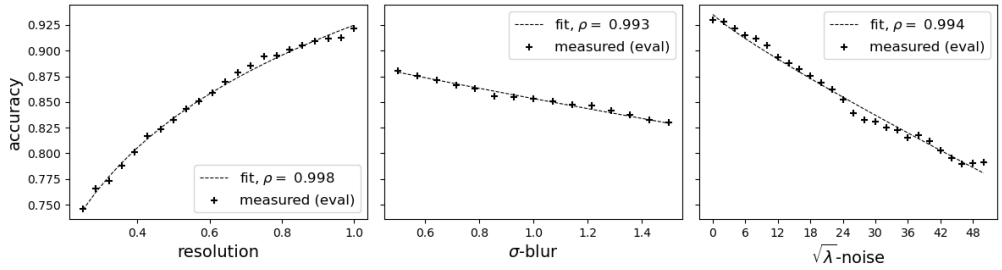



(c) GIQE-3

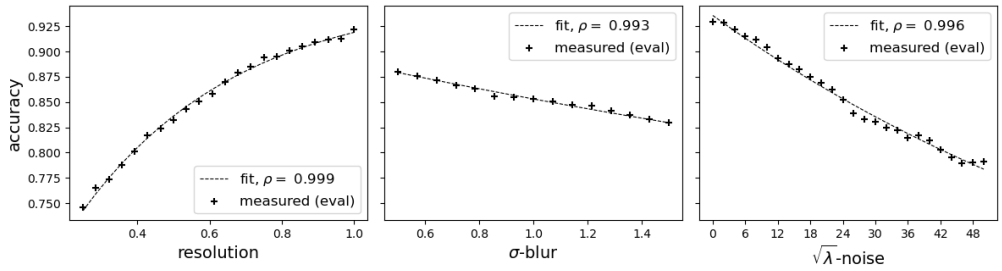



(d) GIQE-5

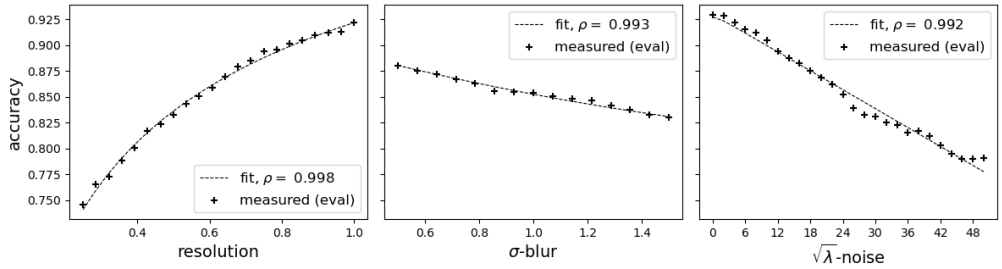
Figure 4.16: Comparison of fit quality for each of our performance prediction model functional forms for Places365, where each distortion has been isolated by averaging out the remaining two.



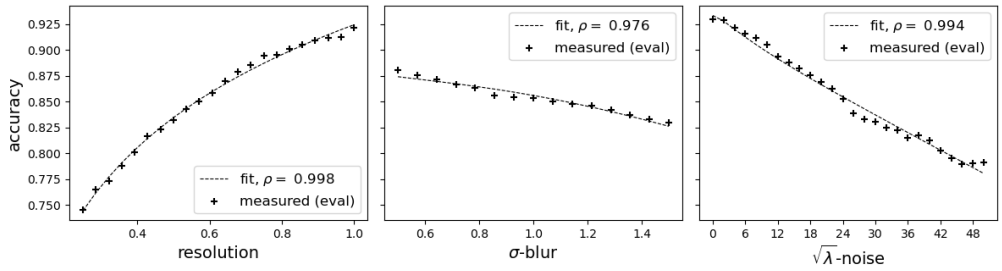

(a) Power law




(b) Exponential

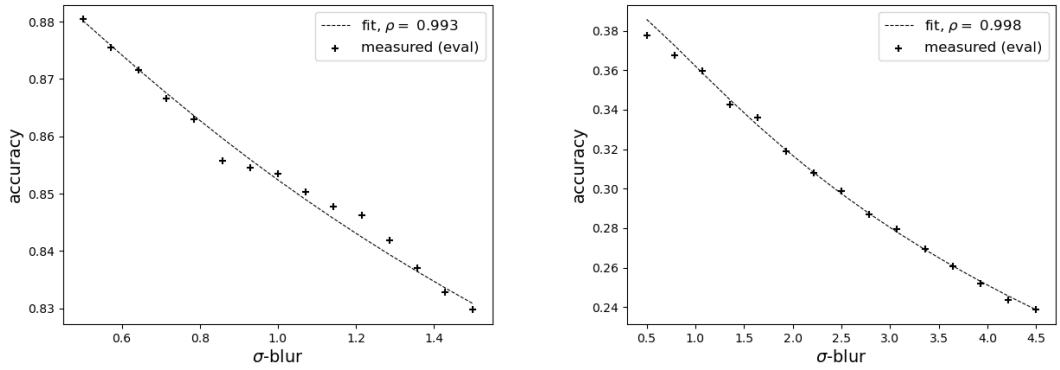



(c) GIQE-3




(d) GIQE-5

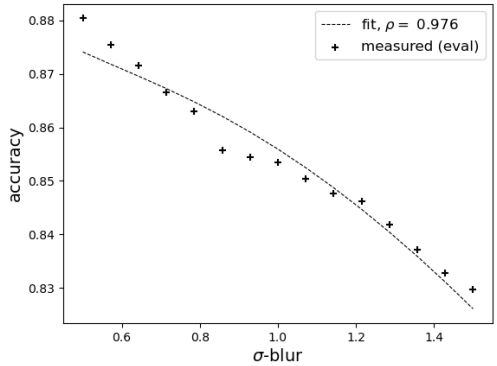
Figure 4.17: Comparison of fit quality for each of our performance prediction model functional forms for SAT-6, where each distortion has been isolated by averaging out the remaining two.



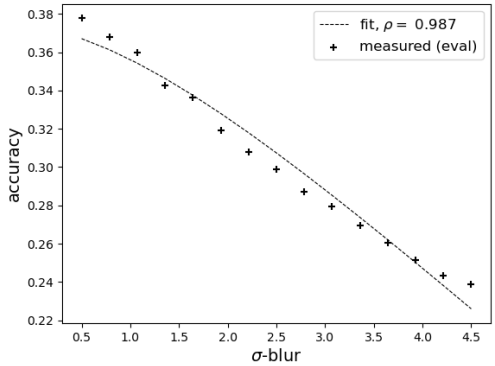

(a) GIQE-3, SAT-6




(b) GIQE-3, Places365

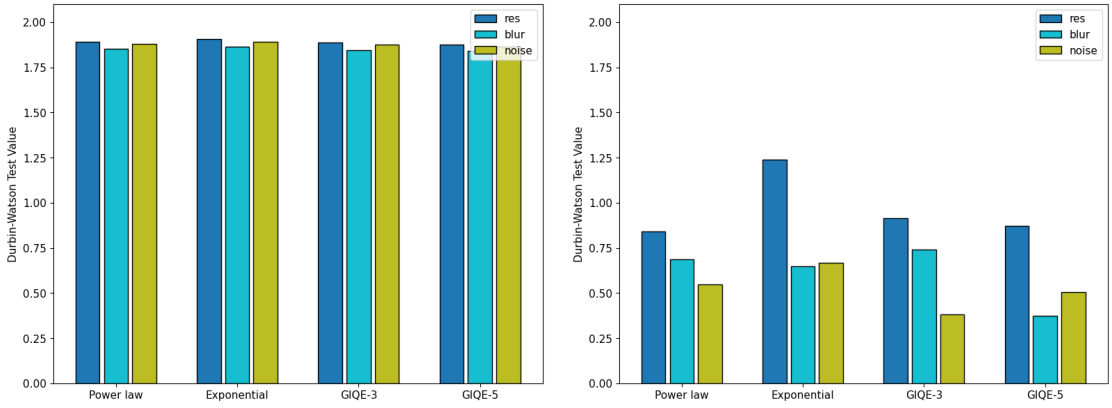



(c) GIQE-5, SAT-6



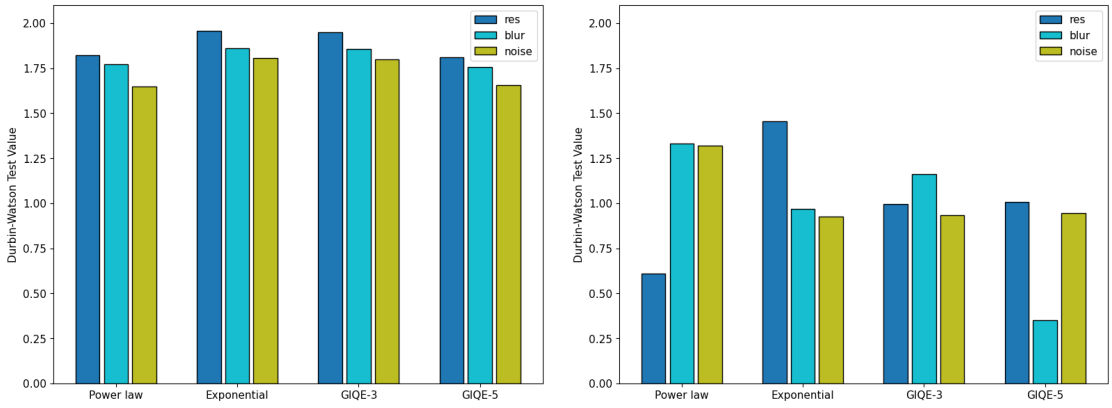

(d) GIQE-5, Places365

Figure 4.18: Comparison of fit quality for blur between the GIQE-3 and GIQE-5 models. Here, we see that our GIQE-3 based model outperforms our GIQE-5 based model in predicting accuracy as function of blur *at least for our data.*




(a) 3-dimensional statistics (b) 1-dimensional statistics

Figure 4.19: SAT-6 Durbin-Watson statistics. The 3d statistics were generated using the process described by Eqn. 4.12, with the distortion labels corresponding the axis axis along which the tensor was unravelled. 1d statics also followed this process after averaging out one of the three distortion dimensions.




(a) 3-dimensional statistics (b) 1-dimensional statistics

Figure 4.20: Places365 Durbin-Watson statistics. The 3d statistics were generated using the process described by Eqn. 4.12, with the distortion labels corresponding the axis axis along which the tensor was unravelled. 1d statics also followed this process after averaging out one of the three distortion dimensions.

## 4.5 Image Classifier Findings

By systematically varying image quality in train and test datasets, we have explored both the vulnerability of naively trained CNNs to image quality degradations as well as the relative robustness of CNNs trained on appropriately degraded images. As observed elsewhere [80, 81, 50, 51, 107, 82, 84], appropriately trained CNNs can perform well against images of very low visual image quality, indicating the potential utility of low-cost sensors paired with tailored computer vision systems and highlighting the centrality of CNN training in optimizing sensing systems that incorporate significant computer vision elements. Most of the relevant research has focused on making CNNs robust to new and un-trained distortions, and generally test distortions have been applied singly. For those designing sensors, it should generally be feasible to train and test on images of similar quality, leaving CNN generalization an important problem but one that is largely orthogonal to the problem of image quality itself. Here, we have focused on showing the extent to which CNNs can learn to see through the types of distortions likely to result from the use of inexpensive imagers, where each of our three distortions has been applied to each image in a sequence that roughly maps to the physical processes of an image chain.

Additionally, we have demonstrated that the functional form of the GIQE version 3 is viable for use in models predicting CNN performance as a function of image quality, at least for the cases described here. Given that the GIQE is used to predict NIIRS, a man-made *utility* metric, the applicability of its functional form for predicting CNN performance as a function of image quality was not obvious *a priori*. That said, our GIQE-derived performance predictor is a five-parameter non-linear model; we recognize that our result may be another demonstration of von Neumann's adage on fitting elephants with non-linear models [108, 109]. The fact that two other non-linear models with no particular pedigree were similarly successful in predicting our classification accuracy suggests the validity of von Neumann's insight.

Significantly, we observe that the historical form of the GIQE does a better of predicting performance in our setting than the current form, GIQE-5. The quartic RER term in GIQE-5 fits our result poorly (Fig. 4.18). We find this result noteworthy but highlight that our study pertains to the accuracy of CNN-based image classifiers. As noted before, this term came about to account for the sub-optimal nature of sharpening kernels often used when analysts apply edge enhancements to images with low RER. CNN training with blurred images optimizes the network's convolutional filters to extract information from images with low RER, and we believe this optimization makes the GIQE-5 adjustment for suboptimal sharpening unnecessary in modeling the performance of appropriately trained CNNs. Additionally, using CNN-based classifiers afforded us the luxury of testing against $\sim 700,000$ images over which we could calculate aggregate accuracy at varied distortion

levels; manually rating the NIIRS values of similar numbers of images is impractical.

Finally, while our GIQE-based models do a reasonable job of predicting the image classification performance of CNNs, we should note that the coefficients of our performance prediction models are dataset dependent. To be useful for a task such as optimizing a remote sensing system *before actual image capture*, the coefficients of any such model should be tuned with representative data passed through computer vision systems performing the task in question.

# Chapter 5

# Object Detector Performance

## 5.1 Introduction

In Chapter 4, we investigated the relationships between image quality and single label image classification accuracy by CNNs. While image classification can be useful on its own, object detection is a task of significant interest in a number of application, particularly in the remote sensing community. Here, we focus our efforts on understanding image quality with respect to the task of object detection by a machine learning algorithm. Image quality and object detection have both been subjects of significant research, with image quality research stretching back decades [4, 5, 6, 8, 7] and object detection research gaining significant traction after the computer vision boom that began with AlexNet in 2012 [9, 110, 111, 112, 113]. A comparatively small body of research, however, has considered the intersection of these topics and examined in detail how image quality affects the performance of deep learning-based object detection algorithms [114, 115]. In general, work related to image quality has focused on capturing and recording images for human viewing, whereas work in computer vision has taken relatively high quality images as its starting point without giving significant thought to the image chains producing them. Given the increasing prevalence of automated image processing for self-driving vehicles and analysis of remote sensing data [116, 117, 118, 119], the relationship between image quality and algorithm performance represents an increasingly important element of end-to-end sensing systems.

Research on the interaction between image quality and computer vision performance has been limited. While a small number of studies have considered the relationship between image quality and the performance of computer vision algorithms, this research has largely focused on image classification rather than object detection [80, 81, 50, 51, 83, 87, 12, 11, 86, 10, 91, 120, 121]. Additionally, several of these studies have observed differences in

image quality as a driver of human visual interpretation and image quality as a determinant of convolutional neural network (CNN) performance [91, 121, 81, 50, 51].

Here, we extend research focused on the relationship between image quality and deep learning algorithm performance [120, 121] to specifically consider the task of object detection. Although image classification has some utility on its own, many of today's interesting computer vision applications such as vehicle autonomy and automated analysis of overhead images tend to be object detection-centric. We believe that the widespread applicability of object detection makes understanding its relationship to image quality an important problem for those interested in systems reliant on computer vision. This research makes the following contributions:

- We quantify the impacts of image quality variables on object detection performance
- We examine the extent to which training object detection models on distorted images leads to performance recovery in testing against images with similar distortions
- We assess the viability of the functional form of the General Image Quality Equation (GIQE) for modeling object detection performance as a function of image quality
- We compare the relationships between image quality and object detection performance to the equivalent relationships for image classifiers

## 5.2 Related Work

As mentioned above and in previous chapters, work on the topics of image quality and computer vision is extensive when the two are considered independently, but only a comparatively small body of research has explored the questions of how image quality affects computer vision algorithm performance. Almost none of this research has specifically examined the effects of image quality on deep learning based object detection algorithms.

For image classification, a number of studies have found CNNs to be prone to classification errors when tested on degraded images, and these studies have generally observed that training against images subjected to similar distortions leads to varying levels of performance recovery [80, 81, 50, 51, 120, 96]. Many of the studies that have considered the relationships between image quality and classifier performance have focused on training schema to make CNNs robust when testing against *unseen* distortions (i.e. distortions not applied to training images [84, 83, 11, 86, 10]). These studies found varying levels of success in producing distortion-robust CNNs, but no approach succeeded in making CNNs impervious to the full range of image distortions considered. Additionally, several studies observed that image degradations affected CNN and human image interpretation

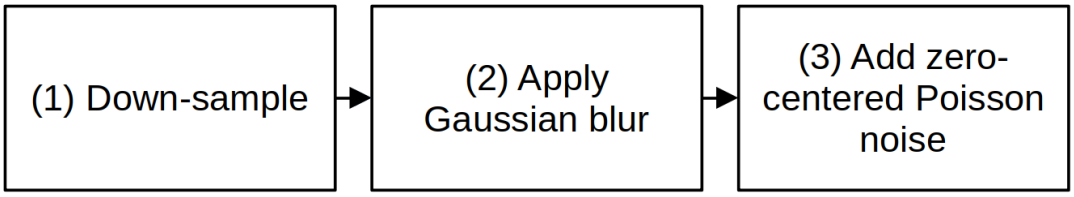



Figure 5.1: Image chain steps

differently, with the image chains optimal for human observers proving sub-optimal for CNN-based analysis and visa versa [91, 81, 92].

Studies exploring object detection performance are far more limited. Kong *et al.* demonstrated an ability to predict object detection performance based on scene statistics after degrading images with down-sampling, blur, and noise, but these studies explored performance relationships for detection algorithms that do not use deep learning [122, 123]. Hsiang *et al.* compared the performance of a number of detection models on videos with varied frame rates, the goal of the research being to understand the tradeoffs in performance associated with varied sensor bandwidths and GPU platforms for automotive applications [114]. Finally, Nath and Behzadan demonstrated that a generative adversarial network (GAN) could be used to up-sample low resolution images and improve the performance of pre-trained YOLO-based models [115]. We have found no systematic evaluations examining the impacts of image quality on CNN-based object detector performance.

## 5.3 Method

In this study, we have used the 2017 Common Objects in Context (COCO) detection dataset and extended the methods presented previously in our work on image classifiers [120, 121] to the problem of object detection [124]. The 2017 COCO detection dataset contains 118K training and 5K validation images with object annotations released, along with 40K testing images for which annotations have not been released [125]. We used two model architectures, Faster-RCNN [112] and YOLOv8 [126]. The YOLOv8 family of models includes multiple model depths/sizes, with suffixes “n”, “s”, “m”, “l”, and “x” denoting the different variants from smallest to largest. Here, we first tested pre-trained Faster-RCNN and YOLOv8 models on distorted images and compare performance as a function of image distortion, observing similar behavior for both architectures. We then trained YOLOv8l models on distorted images and compared the performance of the pre-trained models with the performance of the models tuned on distorted images.

To generate our distorted images, we created parametric image chains that loosely emulate the physical imaging process (Fig. 5.1). Fist, we downsampled our images, which corresponds to a physical imager mapping an angular field of view onto its detector elements, with the angular resolution determined by the ratio of the pixel pitch to the focal length. Next, we blurred our images with Gaussian kernels, which corresponds to the blur inherently imparted by the system point spread function (PSF). Finally, we added zero-centered Poisson noise to roughly emulate the Poisson processes inherent in most image noise sources. Figure 5.2 depicts a COCO image at each stage of the image chain at the distortion space midpoint.

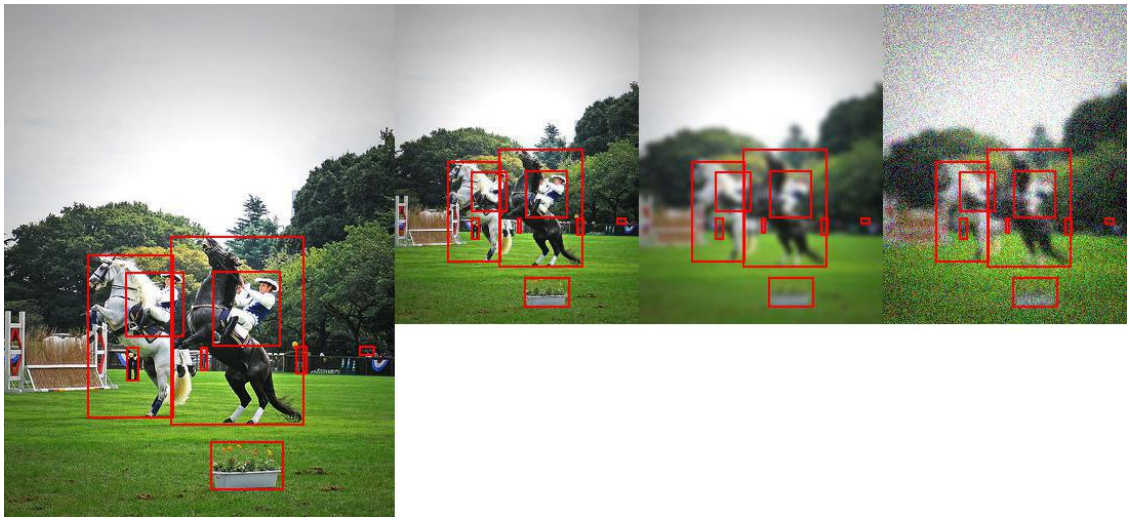


Figure 5.2: COCO image [127] at each stage of the distortion space midpoint image chain; original (left), down-sampled (second from left), blurred (second from right), and noised (right).

### 5.3.1 Primary performance metrics

To evaluate performance, we used mean average precision (mAP) with an intersection over union (IOU) threshold of 0.50, where mAP is defined as the simple mean of the average precision values for all object classes in the dataset. Specifically, in the context of an object detection problem, precision $P$ captures the probability that a predicted detection is a true positive, with

$$P = \frac{TP}{TP + FP} = \frac{\text{total correct object detections}}{\text{total predicted object detections}}, \tag{5.1}$$

where $TP$ represents total number of true positives and $FP$ represents the total number of false positives. Conversely, recall $R$ captures the probability that a ground truth positive will be detected, with

$$R = \frac{TP}{TP + FN} = \frac{\text{total correct object detections}}{\text{total ground truth objects to detect}}, \tag{5.2}$$

where $FN$ represents total false negatives. A true positive occurs when a detector correctly labels an object and assigns a bounding box that satisfies an intersection over union ($IOU$) requirement, where $IOU$ captures the degree of overlap of a predicted bounding box $B_p$ and ground truth bounding box $B_{gt}$, given by

$$IOU = \frac{\text{area}\left(B_p \cap B_{gt}\right)}{\text{area}\left(B_p \cup B_{gt}\right)}. \tag{5.3}$$

A false positive occurs when a detector incorrectly labels an object or assigns a bounding box that does not satisfy an $IOU$ requirement.

For each object detected, the model assigns a confidence between 0 and 1. To generate a precision recall curve, we vary a confidence threshold and discard all detections with confidence below the threshold. A low threshold tends to result in both more true positives and more false positives, leading to high recall and low precision. Conversely, a high confidence threshold tends to result in fewer true positives and more false negatives, leading to low recall and high precision. By varying the confidence threshold, we can interpolate a curve that treats precision as a function of recall. Specifically, we recognize that precision $P$ and recall $R$ are both functions $f_P$ and $f_R$ of a confidence threshold $t_c$, or

$$P\left(t_c\right) = f_P\left(t_c\right) \tag{5.4}$$

and

$$R\left(t_c\right) = f_R\left(t_c\right), \tag{5.5}$$

where in practice $t_c$ is a set of discretely sampled values on $(0, 1)$. We can then define an interpolation operator $\Theta$ that is used to define the curve

$$P_{interp}\left(R\right) \equiv \Theta\left\{P\left(t_c\right), R\left(t_c\right)\right\}. \tag{5.6}$$

Here, we used the interpolation operator used in the PASCAL Visual Object Class (VOC) challenge [128], which is given by

$$P_{interp}\left(R\left(t_c\right)\right) = \max_{t \leq t_c}\left(P\left(t\right)\right) = \max_{\tilde{R} \geq R}\left(P\left(\tilde{R}\right)\right). \tag{5.7}$$

Formally, average precision $AP$ over all recall values is then given by

$$AP = \frac{1}{R_1 - R_0} \int_{R_0}^{R_1} P\left(R\right) dR = \int_0^1 P\left(R\right) dR, \tag{5.8}$$

where $R_0 = 0$ and $R_1 = 1$. In practice, we compute $AP$ by taking the sum over a discretely sampled curve, or

$$AP = \sum_i P_i \Delta R_i. \tag{5.9}$$

For an $N$-class object detection problem, we can generate precision-recall curves for each object class $C_i$ from which we compute an average precision value $AP_i$. Mean average precision $mAP$, then, is the mean of our average precision values $AP_i$ over all classes, or

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i. \tag{5.10}$$

### 5.3.2 Distortion space and training parameters

To establish the bounds of our distortion space, we identified the levels of downsampling, blur, and additive noise that *independently* drove pre-trained models to approach chance performance. Next, we stochastically applied all three distortion types to training and testing images; for each image, we randomly selected a downsampling fraction, blur kernel standard deviation, and Poisson noise standard deviation and applied these three distortions sequentially. Figure 5.3 shows a COCO image with its bounding boxes superposed in its undistorted form (*i.e.*, at the origin of the distortion space), at the test distortion space midpoint, and at the test distortion space endpoint.

Table 5.1: Distortion levels

| | resolution *(fraction)* | $\sigma_{blur}$ *(pixels)* | $\sqrt{\lambda}_{Poisson}$ *(DN)* |
|---|---|---|---|
| Resolution scan | 0.05 - 1 | - | - |
| Blur scan | - | 0.5 - 10 | - |
| Noise scan | - | - | 0 - 100 |
| Full range (train) | 0.2 - 1 | 0.1 - 5 | 0 - 80 |
| Full range (test) | 0.25 - 1 | 0.5 - 4.5 | 0 - 70 |
| Midpoint (train/test) | 0.625 | 2.5 | 35 |
| Endpoint (train/test) | 0.25 | 4.5 | 70 |

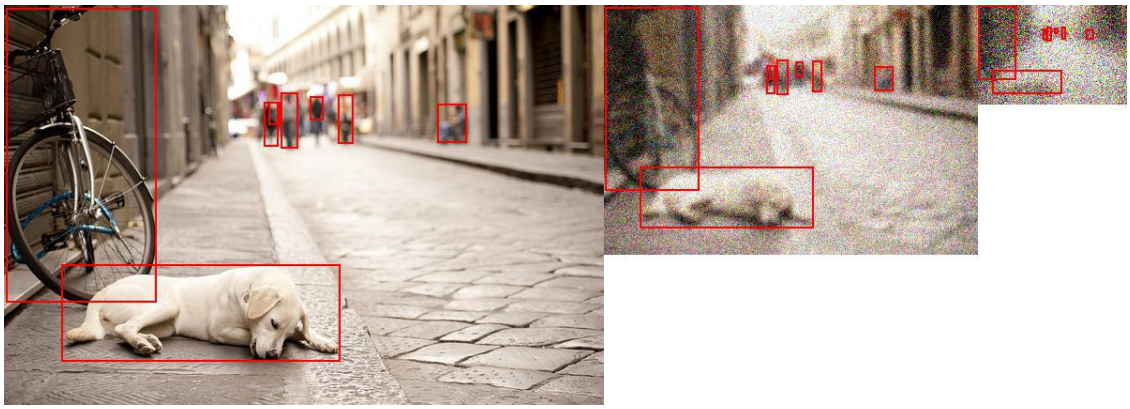


Figure 5.3: COCO image [129] before distortion (left) and at the midpoint (middle) and endpoint (right) of the test distortion space. We note that the image is of very low quality at the midpoint and almost impossible to decipher visually at the endpoint.

Table 5.2: Training parameters.

| | |
|---|---|
| Train dataset size | 118K |
| Validation fraction | 0.05 |
| Distortion tuning epochs | 50 |
| Train batch size | 8 |
| Optimizer | SGD* |
| Learning rate | 0.01 |

*Stochastic gradient descent

After finding the distortion levels that drove object detectors to approach chance performance, we generated a training dataset and an initial testing dataset bounded by these distortion levels. After training and testing across this distortion space, we created *two* slightly narrower test datasets. Table 5.2 summarizes our training parameters. These narrower test datasets remove regions at which performance remained near chance even after training. We also set the minimum blur kernel standard deviation to 0.5 pixels in our test dataset due to the departure from a meaningful Gaussian shape in sampled kernels of extremely low standard deviation, which we have explored in more detail elsewhere [96]. Table 5.1 summarizes the distortion boundaries used.

The two full range test datasets are independently and identically distributed (i.i.d.), meaning the distortions applied to the images across these two datasets are randomly

and independently drawn from identical distributions, giving them equivalent distortion statistics. We generated these test datasets from the same original COCO validation images, and each test dataset contains 100 stochastically distorted copies of the original parent images for a total of 500,000 distorted images in each test dataset. We emphasize that the goal of the study is to understand performance variation with image quality. Because image quality is our primary variable of interest, we take advantage of the ability to measure its effects on performance when the same test images appear at different points in the distortion space.

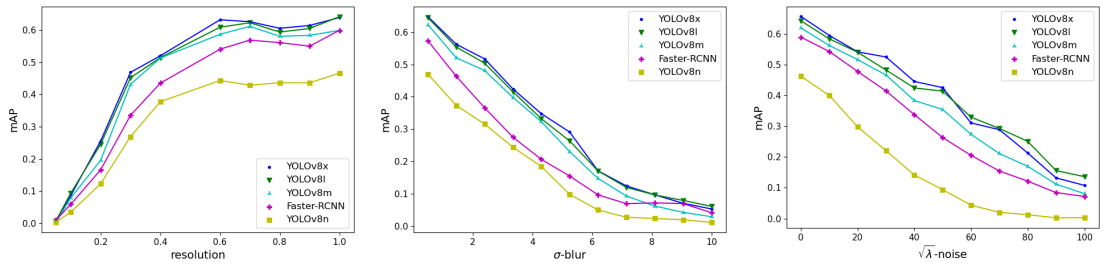
In addition to training and testing models on distortions spanning our train and test distortion space, we also trained and tested models on images at the distortion space midpoint and endpoint (see Table 5.1). To generate our midpoint and endpoint train and test datasets, we applied these midpoint and endpoint distortion combinations to the COCO train and validation datasets respectively. When we trained our midpoint and endpoint models, we started with copies of the model that we had trained across the full training distortion space and then tuned them for an additional five epochs on the midpoint and endpoint training datasets.

## 5.4 Results

### 5.4.1 Single distortion type test results

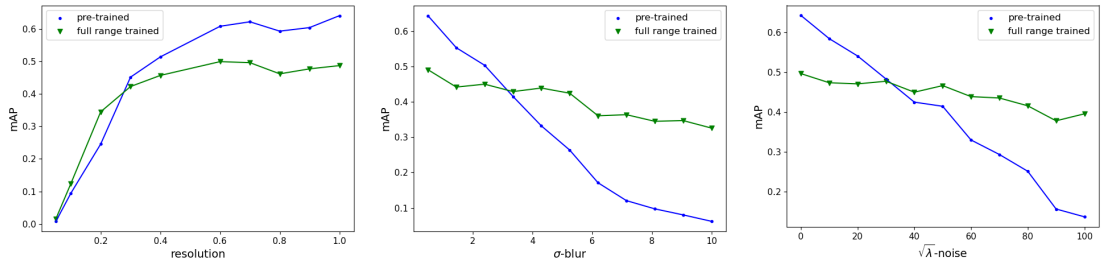
To establish our distortion bounds and understand the behavior of pre-trained models, we tested a subset of the YOLOv8 family of models and a Faster-RCNN model on our single distortion type datasets listed in Tab. 5.1. Figure 5.4 depicts these results. We observe that the larger YOLO models performed best in absolute terms, with all except the smallest YOLO model (YOLOv8n) having very similar absolute performance. All models displayed similar distortion performance trends. For comparison, we also tested a pre-trained Faster-RCNN model on these single distortion datasets and observed performance trends similar to those seen for YOLOv8, with its absolute performance falling in between the performances of YOLOv8n and YOLOv8m.

When we tested the full range trained model on the single distortion datasets, we observed a significant loss in performance at low distortion levels and a significant improvement at higher blur and noise levels (Fig. 5.5b and Fig. 5.5c). Surprisingly, tuning across the full distortion range did little to improve performance at low resolution levels (Fig. 5.5a).




(a) Resolution scan (b) Blur scan (c) Noise scan

Figure 5.4: Performance of pre-trained YOLOv8 models and Faster-RCNN against single-distortion test datasets.




(a) Resolution scan (b) Blur scan (c) Noise scan

Figure 5.5: Performance of a pre-trained YOLOv8l model and a YOLOv8l model tuned across the full range of the distortion space. Here, we observe that tuning across the full distortion range results in a significant performance loss on on high quality images while improving performance on images with large blur and noise distortions. We also note that this full range distortion tuning does little to improve performance against low resolution images.

### 5.4.2 Full distortion space test results

Figures 5.6 and 5.7 summarize the performance of four YOLOv8l models when we trained and tested them on undistorted images, images at the distortion space midpoint, images at the distortion space endpoint, and images spread throughout the full distortion range. In Figure 5.6, which shows the average performance of each model over each test dataset, we see that the models tuned at specific distortion points outperform the models tuned on images of different quality, as we would expect. In Fig. 5.7, which shows average performance as a function of resolution, blur, and noise over the full range test dataset, we also see that the midpoint model demonstrates a preference for images with moderate blur and noise. Additionally, the endpoint model displays performance that is relatively stable with respect to resolution, with performance improving slightly as blur is increased and performance increasing substantially as noise is increased. These figures also show that tuning the midpoint model did little to diminish its performance elsewhere relative to the full range model. We note here that when we tuned our midpoint and endpoint models, we started with copies of the full range model; tuning on very low quality images drove the endpoint model to "forget" more than the midpoint model.

When we compare the performance of our pre-trained and full range models tested on images with distortions spanning the full test distortion space, we observed that our full range tuned models performed better overall, while the pre-trained models continued to perform well against images near the low distortion corner of the distortion space. Figure 5.8 depicts these results.

To generate Figs. 5.8a - 5.8c, we calculated mAP for both models at each distortion point $\left(r, \sigma, \sqrt{\lambda}\right)$, and we then took the average over one of the three distortion dimensions to find the mean performance as a function of the remaining two distortions. For instance, mAP as a function of resolution and blur represents the average across all noise values. In these plots, we can see that our full range trained model performs best on average, while the pre-trained model (which saw only high quality images in training) outperforms the full range model for low distortion values. To generate the 1D plots in Fig. 5.7, we took the average of the mAP across two of the three distortion dimensions to get performance as a function of the remaining distortion variable. Overall, we see that tuning across the full range of distortions decreases peak performance while improving average performance on distorted images.

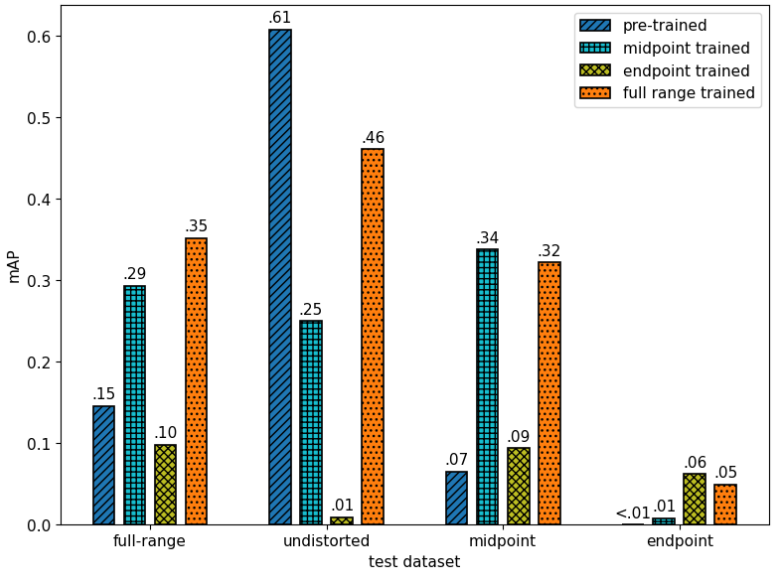



Figure 5.6: Average performance of four YOLOv8l models trained and tested on undistorted images, images at the distortion space midpoint, images at the distortion space endpoint, and images across the full distortion range.

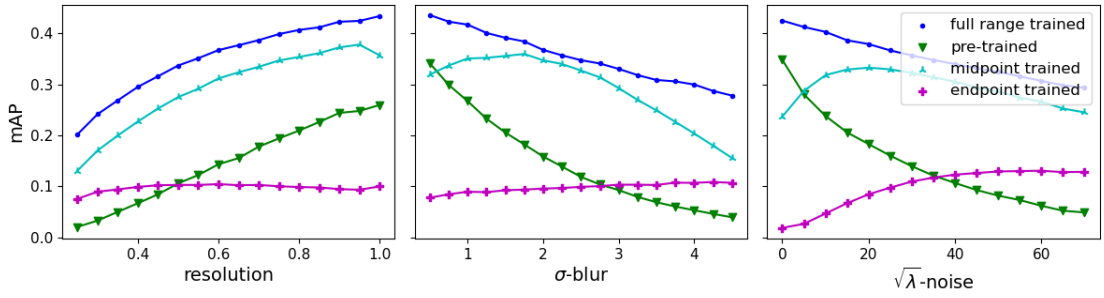
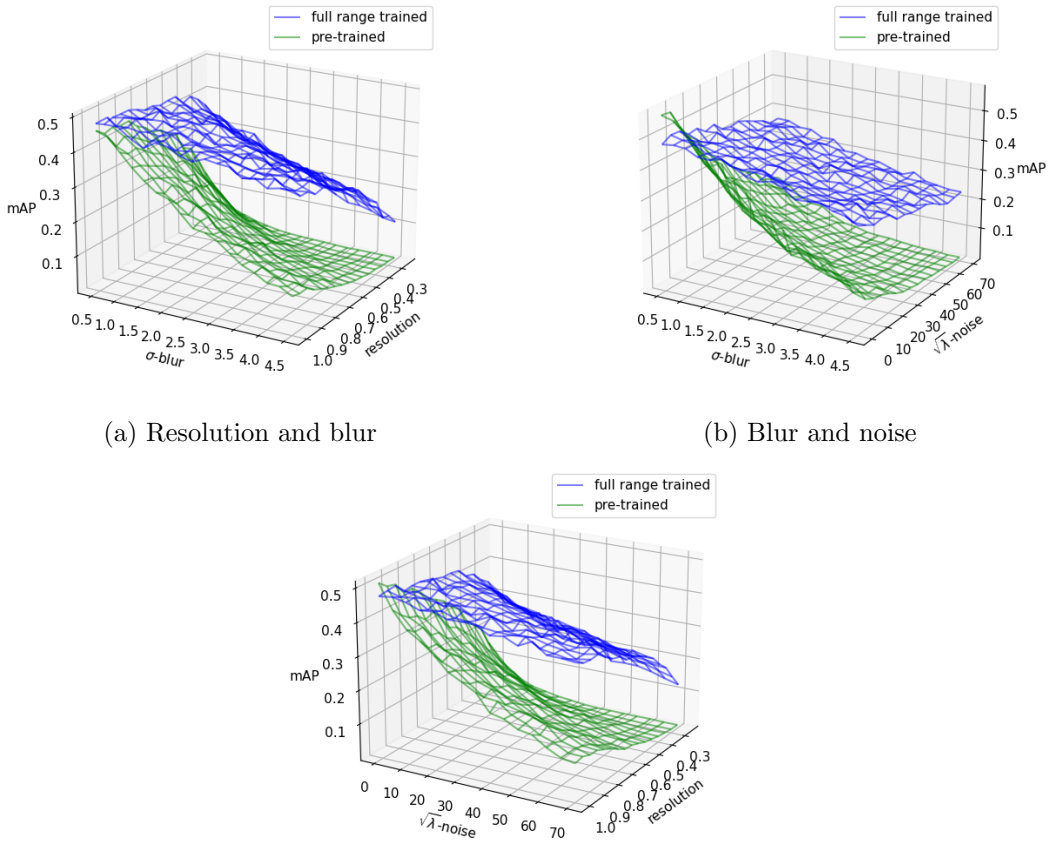



Figure 5.7: Mean performance as a function of resolution, blur, and noise of four YOLOv8l models trained on differing distortions, tested on a full range test dataset.



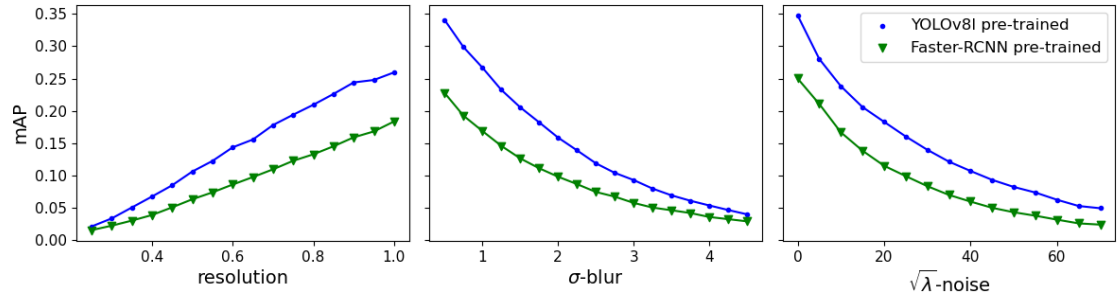

(a) Resolution and blur

(b) Blur and noise

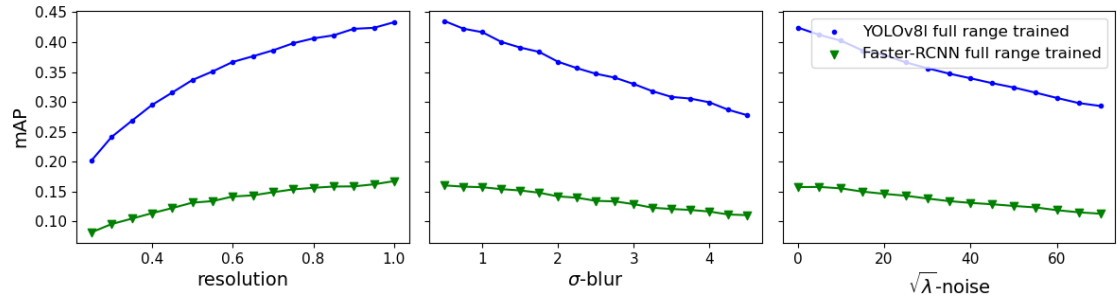
(c) Resolution and noise

Figure 5.8: Performance variation with resolution, blur, and noise for a pre-trained model and a full range model, tested against our full range test dataset. Here, we observe that the full range model outperforms the pre-trained model on average, while the pre-trained model outperforms the full range model on high quality images.

### 5.4.3 Object detector architecture comparison




(a) YOLOv8l and Faster-RCNN pre-trained




(b) YOLOv8l and Faster-RCNN full range

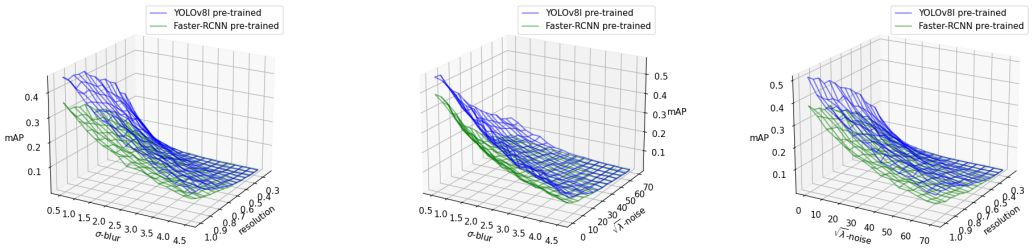
Figure 5.9: Performance of pre-trained and full range trained YOLOv8l and Faster-RCNN models

Similar to the image classification architecture comparison presented in 4.3.2, we compared the performance across image distortions of the YOLOv8 architecture with that of the older Faster-RCNN architecture. We observed qualitatively similar trends in performance as a function of distortion for both architectures, but we found that our YOLOv8l models regained significantly more performance when trained on distorted images. Figure 5.4 in Sec. 5.4.1 shows how the performance of a pre-trained Faster-RCNN model compares with that of various YOLOv8 models over single-distortion test datasets. There, we observe that the Faster-RCNN model's performance falls between the performances of the two smallest YOLOv8 models that we tested.

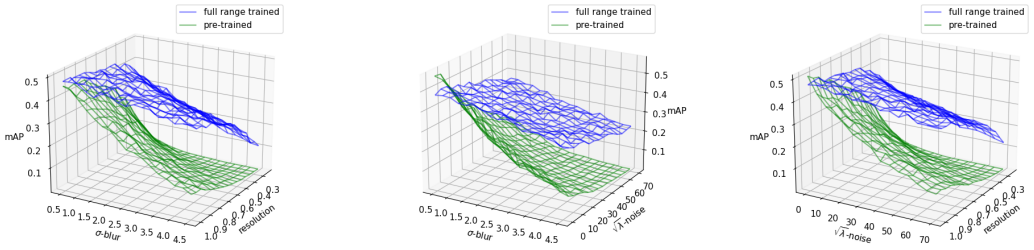
When we compare the performance of a pre-trained YOLOv8l model with that of a Faster-RCNN model across the full range test distortion space, we observe similar behavior. In Figs. 5.9a and 5.10, we see that the pre-trained YOLOv8l model outperforms the pre-

pretrained Faster-RCNN model across the distortion space, with both models exhibiting similar variations in performance as a function of distortion level. When we compare the performances of models trained across the full distortion space, however, we see that the YOLOv8l model trained across the full distortion space performs drastically better than the Faster-RCNN model also trained across the full distortion space. Our full range trained Faster-RCNN model does not appear to learn to "see through" the distortions to nearly the same extent as our YOLOv8l model.

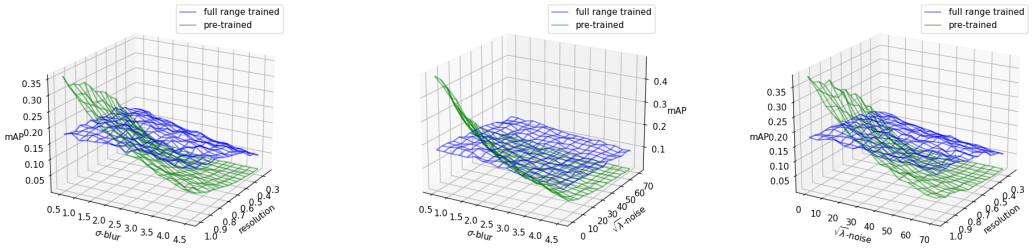
We find the difference in performance of these models after tuning across the full distortion space surprising; we observed very little difference in the performance of our classifier architectures when we trained and tested ResNet and DenseNet models across the full classifier distortion space in Sec. 4.3.2. While factors ranging from fundamentals in the architecture differences to choices in training parameters may be driving the difference in the performances of these models after training across the full distortion range, we have not identified a unique cause. Based on the performance improvements at higher distortion levels, we know that training across the full distortion range does enhance performance in some regimes. We also observed nothing unexpected in our loss curves during training, with training loss smoothly declining and validation loss leveling out and then climbing slightly after roughly 25 epochs (not shown here). Additionally, we observed that Faster-RCNN models trained across the full distortion range for only a few epochs performance far worse than those trained longer. But while the overall performance of the Faster-RCNN model is significantly lower across the full distortion space, the performance trends are similar for both architectures; the performance curves have roughly the same shapes. In short, we do not have an explanation for the difference in performance across these architectures. Given the fact that performance variation as a function of image distortion is similar between the the architectures, we do not believe that unraveling this anomaly is critical in understanding the underlying relationships between image quality and computer vision performance.



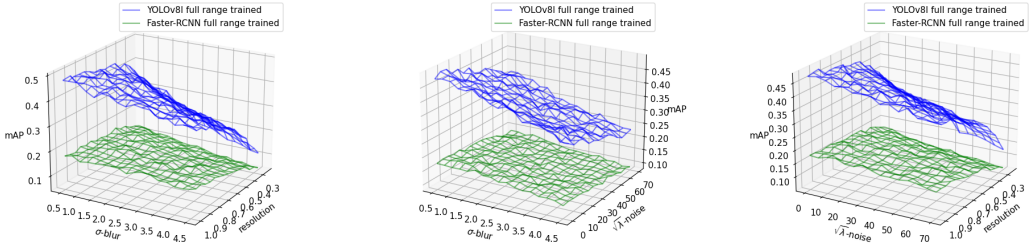

(a) YOLOv8l and Faster-RCNN pre-trained




(b) YOLOv8l full range and pre-trained




(c) Faster-RCNN full range and pre-trained




(d) YOLOv8l and Faster-RCNN full range

Figure 5.10: Performance of pre-trained and full range trained YOLOv8l models *(left)* and Faster-RCNN models *(right)* on a full distortion range COCO test dataset.

### 5.4.4 Composite performance results

As we did in studying image classifiers (see Sec. 4.3.3), we constructed composite performance results by dividing the training distortion space into "octants" and training a model across the distortion range of each octant. We then tested these model $M_i$ on each of our two i.i.d. test datasets and calculated the resulting $\text{mAP}_i\,(r, \sigma, n)$ at each point in the distortion space and calculated each model's average mAP over each octant of the test distortion space. We then constructed a composite performance tensor $\text{mAP}_{composite}$ by assigning the mAP from each octant's best performing model to points in that octant.

We used the results on the first test dataset to determine the best performing model in each octant and to generate the distortion performance $\text{mAP}_{predict}\,(r, \sigma, n)$ used for fitting our performance prediction models. We then used the test results on the second of the two i.i.d. test datasets to construct our $\text{mAP}_{eval}$ which we used for evaluating our these performance prediction models.

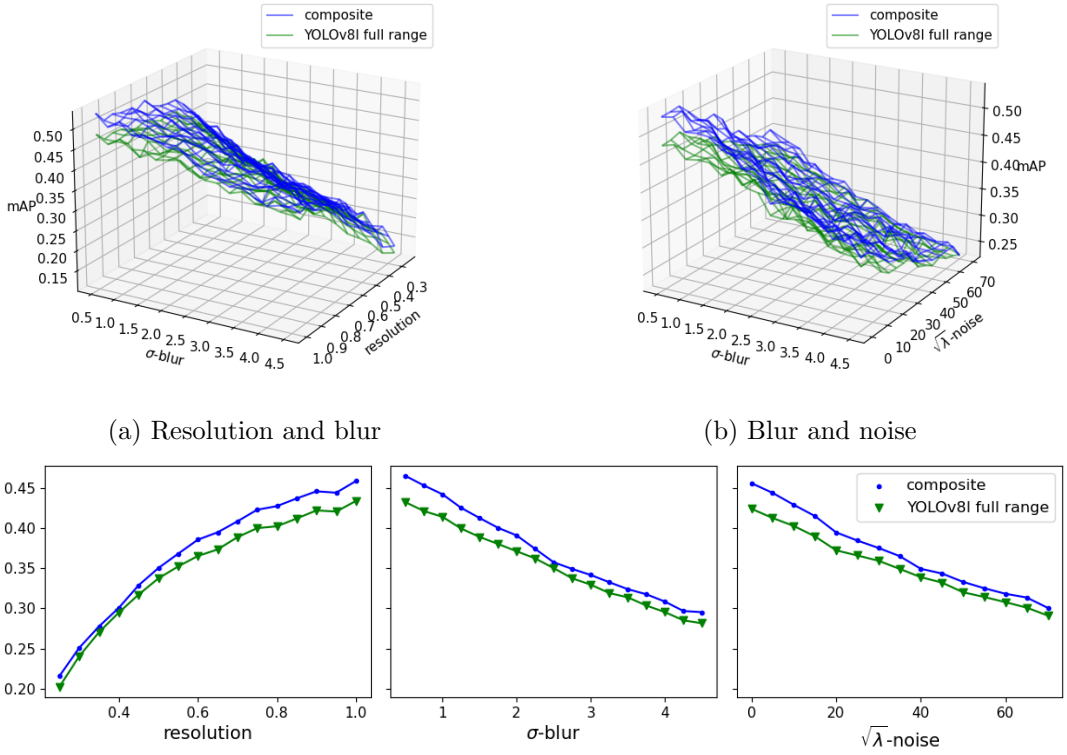
In the classification work presented in Ch. 4, we divided each dimension of the distortion space at its midpoint, creating eight octants of equal volume. Here, we deviate slightly from this original approach. First, we left overlap in our train octants as a means of data augmentation to build model robustness and decrease the likelihood of a performance dip at the octant boundaries. Second, we did not divide the resolution axis of our test distortion space at the midpoint; when we did so, we found that the models trained on higher resolution training octants had higher average performance on the lower resolution test octants than the models trained on the lower resolution training octants. Models trained on lower resolution images outperformed models trained on higher resolution images only near the extreme end of the resolution range. It was only when we adjusted the resolution boundary to be below the midpoint of the test distortion space that all of the models contributed to the composite performance result. We ultimately set the octant boundary at the point that maximized the global average of $\text{mAP}_{predict}$.

Table 5.3 shows the octant boundaries we used for training and testing. The octant with the highest quality images would be use the *high* range range for each distortion dimension, the octant with the lowest quality images would use the *low* column's range, etc.

As we observed in our classifier results (Sec. 4.3.3), using a composite performance result build from octant model results in a modest overall performance improvement with performance trends almost identical to those of a single model trained across the full distortion space. Figure 5.11 illustrates the slight improvement found by moving to the octant model composite performance.

Table 5.3: Train and test distortion octants

| | Train | | Test | |
|---|---|---|---|---|
| | *high* | *low* | *high* | *low* |
| resolution *(fraction)* | 0.55 - 1 | 0.2 - 0.65 | 0.4 - 1 | 0.25 - 0.4 |
| $\sigma_{blur}$ *(pixels)* | 0.2 - 2.7 | 2.3 - 5 | 0.5 - 2.5 | 2.5 - 4.5 |
| $\sqrt{\lambda}_{Poisson}$ *(DN)* | 0 - 45 | 35 - 80 | 0 - 35 | 35 - 70 |




(a) Resolution and blur

(b) Blur and noise

(c) Average performance as a function of resolution, blur, and noise

Figure 5.11: Comparison of octant model composite performance and the performance of a single full range trained YOLOv8l model across the full test distortion space.

## 5.5 Analysis and discussion

### 5.5.1 Comparison to image classification results

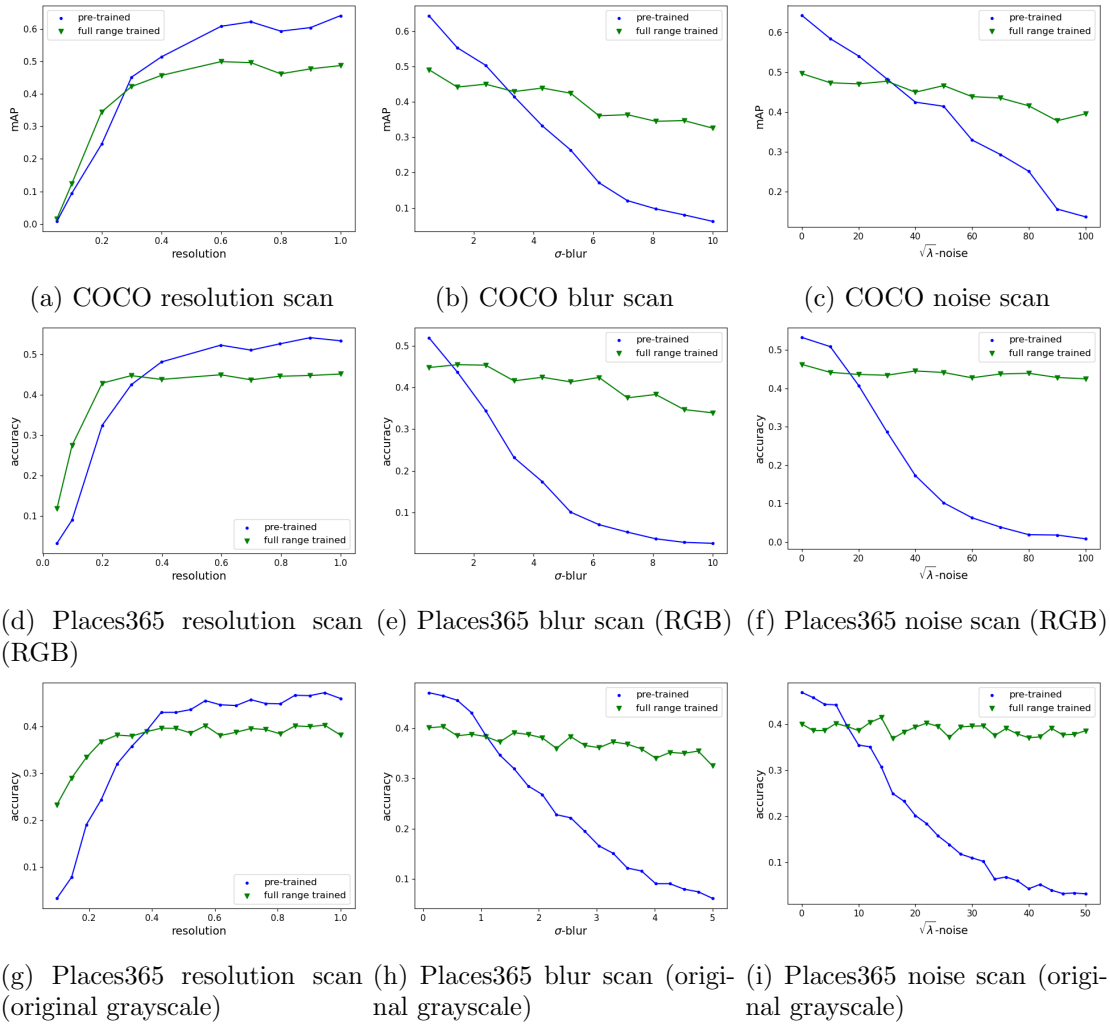
To compare our object detection results to the results we previously obtained studying image classifiers, we first used a DenseNet-161 model tuned across the full training distortion space from our previous study [121] and tested it on single-distortion versions of the Places365 validation dataset. Next, we re-created key elements of our image classification study, this time in RGB using the same distortion space that we used for COCO to enable a direct comparison between the results. Specifically, we trained a Places365 ResNet-18 model across the full COCO distortion space, *with the COCO distortions applied to the Places365 dataset in RGB*. We then created Places365 test datasets with the same distortions that we applied to our COCO test datasets, and we tested both pre-trained and full range trained ResNet-18 models on these "COCO-parallel" datasets.

Figure 5.12 compares our classification results on Places365 with our object detection results on the COCO dataset when distortions are applied singly. We highlight that the performance metrics differ, with COCO object detection measured by mAP and Places365 measured by simple top-1 accuracy. We also point out that the $x$-axis scales differ, with the blur and noise ranges for COCO and Places365 RGB extending roughly twice as far as the blur and noise ranges shown here for the original Places365 results. Having noted these caveats, we see that the performance trends are qualitatively similar for pre-trained and full range models across all datasets; we also observe subtle differences differences and a significant parallel.

First, we observe that the effect of tuning the models across the *full distortion space* before testing on a *single-distortion test dataset* differs between our COCO object detection results and our Places365 image classification results. The performance loss of the full range models against low distortion, higher quality images is less pronounced for the Places365 classification models than it is for the COCO object detection model. This difference is clearest for blur, where the Places365 full range model began outperforming the pre-trained model once the Gaussian blur kernel standard deviation reached $\sigma \approx 1.5$ pixels; conversely, the full range COCO object detection model does not outperform the pre-trained model on the blurred test dataset until standard deviation has reached $\sigma \approx 4$ pixels. Additionally, we find it noteworthy that tuning across the full distortion range does little to improve performance as a function of resolution for the object detection model, whereas this full range tuning does delay the performance drop of the Places365 classification model. Finally, we also find it interesting that performance as a function of resolution is relatively stable for both object detection and classification until resolution drops to roughly 40%.

Second, and perhaps most significantly, we observe that performance as a function of

resolutions remarkably stable in our COCO and Places365 results when resolution remains above roughly 40%, particularly for the full range trained models. Below this threshold we find that resolution drops off quickly in these single distortion results. We also observe that training against distorted images leads to only limited performance improvement at low resolution, particularly in our object detection results. Additionally, we find that full range Places365 RGB model loses almost no performance at all until the resolution fraction drops below 20%, which is the minimum resolution used in training our COCO and Places365 RGB models.




(a) COCO resolution scan (b) COCO blur scan (c) COCO noise scan

(d) Places365 resolution scan (RGB) (e) Places365 blur scan (RGB) (f) Places365 noise scan (RGB)

(g) Places365 resolution scan (original grayscale) (h) Places365 blur scan (original grayscale) (i) Places365 noise scan (original grayscale)

Figure 5.12: Performance of pre-trained and full range models on ***single-distortion*** COCO and ***single-distortion*** Places365 test datasets. We note the $y$-axis metrics differ, with COCO object detection performance quantified by mean average precision (mAP) and Places365 classification performance quantified by top-1 accuracy. Additionally, we highlight that the distortion axis scales differ between the datasets, with COCO and Places365 RGB having the same distortion axes and the Places365 original grayscale having narrower distortion ranges. *(The COCO results shown here are duplicates from Fig. 5.5, repeated for clearer comparison with the Places365 results.)*

### 5.5.2 Application of the GIQE to object detection performance

Previously, we used the the GIQE to model image classification performance as a function of image distortion; there, we found that the functional form of the GIQE could model object detection performance with reasonable fidelity, but we observed that the historical functional form used in versions 3 and 4 of the GIQE outperformed the current form [121]. Here, we apply the GIQE to the object detection performance of our full range YOLOv8l model when tested across the full distortion space. As in our previous work, we map our distortion variables—resolution fraction $r$, Gaussian blur standard deviation $\sigma$, and zero centered Poisson noise standard deviation $n$—to the GIQE variables of GSD, RER, and SNR. We use the relationships $GSD \propto \frac{1}{r}$ and $SNR \propto \frac{1}{n_0^2+n_1^2}$, where $n_0$ represents the noise present in the image before distortion and $n_1$ represents the deliberately added noise. For RER, we again use

$$\text{RER} \approx \frac{1}{\sqrt{2\pi\left(\sigma_0^2+\sigma_1^2\right)}}, \qquad \text{(4.5 revisited)}$$

which holds when $\sigma_0^2+\sigma_1^2 \gg \frac{1}{2\pi}$. A full derivation for Eqn. 4.5 is presented elsewhere [96]. Using this variable mapping, we fit equation 5.11 to our performance results:

$$\begin{aligned}\text{mAP}_{predicted}(r,\sigma,n) \;=\;& c_0 + c_1 \log_{10}\left(\frac{1}{r}\right) \\ &+\; c_2\left(1-\exp\left(c_3\sqrt{n_0^2+n_1^2}\right)\right)\cdot\log_{10}\left(\frac{1}{\sqrt{2\pi\left(\left(c_4 r\right)^2+\sigma^2\right)}}\right) \\ &+\; c_5\left(\log_{10}\left(\frac{1}{\sqrt{2\pi\left(\left(c_4 r\right)^2+\sigma^2\right)}}\right)\right)^4 + c_6\sqrt{n_0^2+n_1^2}, \qquad (5.11)\end{aligned}$$

where the term $c_4 r$ captures the native blur of the images and the sharpening effect of downsampling. We set $n_0 = 2$, which we estimate to be the approximate noise in DN of a typical 8-bit RGB image based on basic sensor modeling.

Figure 5.13 shows the predicted and measured performance when we fit our GIQE-based model to object detection performance on the first of two i.i.d. test datasets and evaluated the fit using object detection performance on the second of these two i.i.d. test datasets. Here, we observe that our model based on GIQE-5 does a qualitatively good job of predicting object detection performance. In our previous work with image classifiers, we observed that this particular model did a qualitatively poor job of predicting accuracy

as a function of blur [121]. Conversely, in the center plot of Fig. 5.13d where we display mAP as a function of blur, we do not see a qualitative mismatch between predicted and measured performance.

In our previous study, we also fit three additional performance prediction models for comparison [121]. The first of these additional models was based on the historical GIQE-v3 and here takes the form

$$\text{mAP}_{predicted}(r,\sigma,n) = c_0 + c_1 \log_{10}\left(\frac{1}{r}\right) + c_5 \log_{10}\left(\frac{1}{\sqrt{2\pi\left((c_4 r)^2+\sigma^2\right)}}\right) + c_6\sqrt{n_0^2+n_1^2}. \tag{5.12}$$

We used this performance prediction model to assess whether the historical form of the GIQE could model performance with fidelity similar to that of the updated GIQE, which includes modifications to account for the sharpening algorithms typically used by (human) analysts in the course of their work [14].

The two other models we used were power law and exponential functions, which we included to assess whether the various forms of the GIQE were unique in their ability to model image quality relationships or whether two somewhat arbitrary non-linear functions could achieve similar results. These exponential and power law comparison functions take the respective forms

$$\text{mAP}_{predicted}(r,\sigma,n) = c_0 + c_1 e^{c_2 r} + c_3 e^{c_4 \sigma} + c_5 e^{c_6 n} \tag{5.13}$$

and

$$\text{mAP}_{predicted}(r,\sigma,n) = c_0 + c_1 r^{c_2} + c_3 \sigma^{c_4} + c_5 n^{c_6}. \tag{5.14}$$

When we fit these additional models to our object detection performance data, we see results qualitatively similar to our Eqn. 5.11 results (see Appendix 8.1). Tables 5.4 and 5.5 contain the fit parameters that we obtained. (We separated these parameters into two tables to emphasize that there is no connection between the GIQE fit parameters and the exponential / power law fit parameters). Examining the GIQE-3 and GIQE-5 fit coefficients, we see that the bias, resolution, and noise terms are similar, while the RER coefficients differ due to the SNR-RER cross term and exponent over the independent RER term in GIQE-5. We observe that all four of these models do a reasonable job of fitting performance as a function of resolution, blur, and noise, with all models having nearly identical correlation coefficients $\rho$. Table 5.6 shows the relatively small variation in AIC scores for these models. We calculated these AIC scores using the approach discussed in 4.4.3, with the exception that here we assumed Gaussian errors since the binomial distribution appropriate to modeling a top-1 accuracy experiment does not apply to object

detection scores. We note a strong qualitative similarity to the image classification results we have previously reported, with the notable exception that the GIQE-5 model fits object detection performance as a function of blur well in these results while it did not fit classification accuracy as a function of blur well in our previous results [121].

Table 5.4: Performance prediction fit coefficients for our GIQE-5 (Eqn. 5.11) and GIQE-3 (Eqn. 5.12) based models. $\rho$ is coefficient of correlation between predicted and measured mAP across the 3D distortion space.

| | $c_0$ | $c_1$ | $c_2$ | $c_3$ | $c_4$ | $c_5$ | $c_6$ | $\rho$ |
|---|---|---|---|---|---|---|---|---|
| GIQE-5 | 0.699 | -0.464 | 0.264 | -2.13 | 2.20 | -0.0205 | -1.84e-3 | 0.946 |
| GIQE-3 | 0.723 | -0.465 | - | - | 2.18 | 0.307 | -1.82e-3 | 0.945 |

Table 5.5: Performance prediction fit coefficients for our power law (Eqn. 5.14) and exponential (Eqn. 5.12) models. $\rho$ is coefficient of correlation between predicted and measured mAP across the 3D distortion space.

| | $c_0$ | $c_1$ | $c_2$ | $c_3$ | $c_4$ | $c_5$ | $c_6$ | $\rho$ |
|---|---|---|---|---|---|---|---|---|
| Exponential | -0.210 | -0.498 | -2.54 | 0.450 | -0.126 | 0.357 | -6.30e-3 | 0.940 |
| Power law | -10.0 | -10.6 | 0.0157 | -0.0665 | 0.759 | -3.03e-3 | -1.82e-3 | 0.939 |

Having seen that the GIQE-5 model fit our object detection results better than it fit our image classification results discussed in Chapter 4, we fit a GIQE-5 performance prediction model to the performance of a Places365 ResNet-18 model trained across the COCO distortion space applied to the Places365 dataset in its native RGB format (discussed in Sec. 5.5.1); there, we found that the GIQE-5 performance prediction model fit the Places365 classification accuracy results reasonably well, as shown in Fig. 5.14. We hypothesize that the better GIQE-5 fits observed in the COCO and Places365 RGB fits result from a comparatively decreased impact of noise in our RGB images. Specifically, in RGB images each color channel's noise is de-coupled from the other two channels, lessening the impact of a given noise level relative to that same noise level applied to grayscale images. We believe that this change in coupling affects the coupled SNR-RER term in the GIQE-5.

Table 5.6: COCO performance prediction model Akaike information criterion (AIC) scores

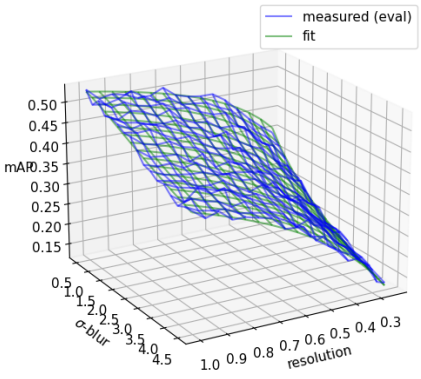
| | $AIC$ | $\Delta AIC$ |
|---|---|---|
| GIQE-5 | 18450.6 | 12.9 |
| GIQE-3 | 18443.3 | 5.6 |
| Power law | 18437.7 | 0 |
| Exponential | 18446.2 | 8.5 |

## 5.6 Object Detection Findings

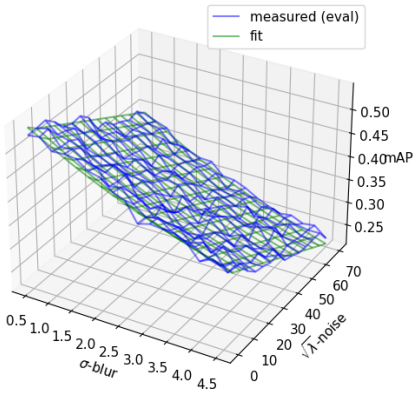
Here, we have shown that CNN-based object detection performance as a function or resolution, blur, and noise follows similar patterns to those seen when studying image classifier performance in the presence of similar distortions, with some differences [121]. First, we have observed that training our object detector on distorted images resulted in a substantial performance loss when tested on high quality images. While we have observed similar performance drops on high quality images when classifiers are tuned on distorted images, this performance decrease is proportionally larger for our object detector.

We have also observed that object detection performance as a function of resolution is relatively stable for resolution fractions above roughly 0.4 when other distortions are not present. Because we have applied blur after down-sampling, the impacts of blur and resolution are coupled; a blur kernel of a given size is proportionally larger after down-sampling. We also find it significant that training on distorted images does little to improve performance as a function of down-sampling *when other distortions are not present* (Fig. 5.5a); in the presence of deliberate blur and noise, however, tuning on distorted images still improves performance as a function of resolution (Fig. 5.7). We point out that this insensitivity to resolution is consistent with the findings of Jaffe *et al.* [91]

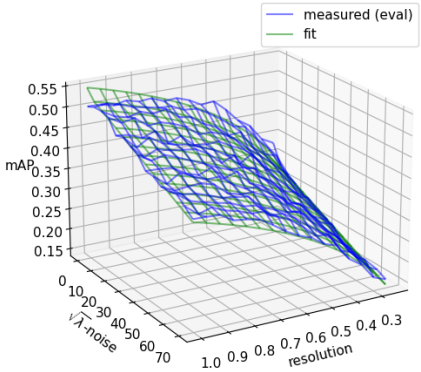
Finally, we observe that both the current and the historical forms of the GIQE are able to model object detection performance well in our study. We find it noteworthy that the current form of the GIQE fits our object detection results better than it fit our classification results in previous work [121], with the exception the updated classification results in RGB discussed in Sec. 5.5.2. We also point out that other non-linear functions with no particular pedigree model performance as well as our GIQE-based models. Finally, we highlight that the historical form of the GIQE models our performance result roughly as well as the current form of the GIQE, which we consider significant given the comparative simplicity of the historical form.



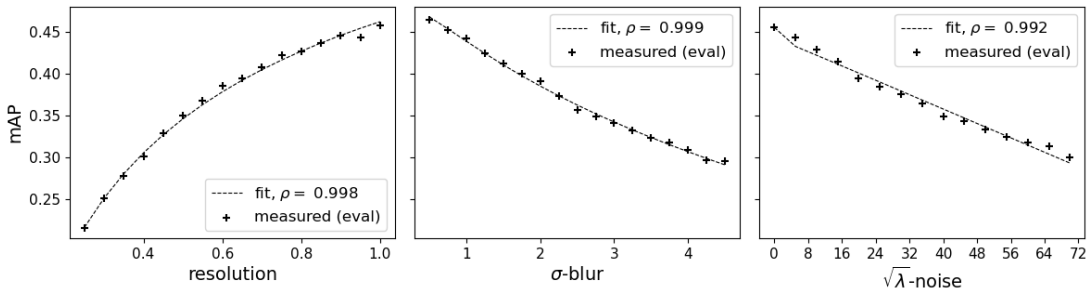

(a) Resolution and blur




(b) Blur and noise

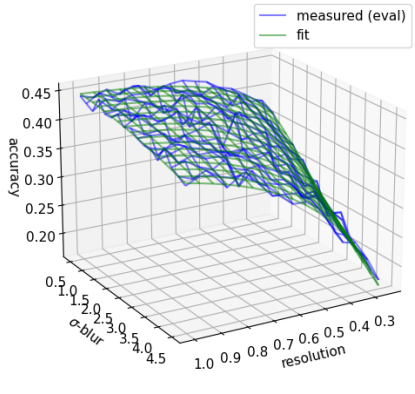



(c) Resolution and noise

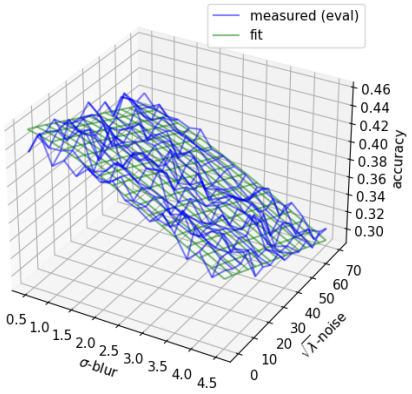



(d) Mean measured and predicted performance as a function of resolution, blur, and noise

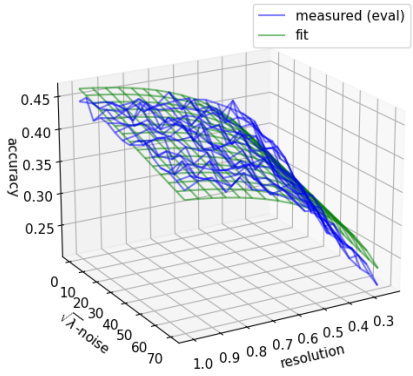
Figure 5.13: Measured and predicted performance as a function of resolution, blur, and noise for our octant model composite performance. We fitted Eqn. 5.11 to performance on the first of two i.i.d. test datasets evaluated the fit on the second.



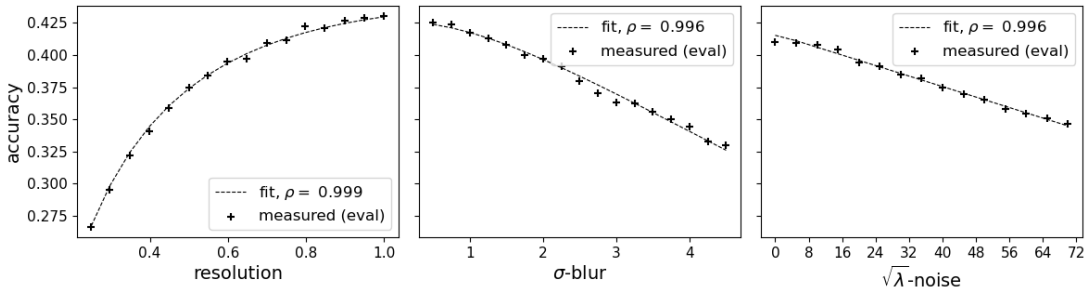

(a) Resolution and blur




(b) Blur and noise




(c) Resolution and noise




(d) Mean measured and predicted performance as a function of resolution, blur, and noise

Figure 5.14: Measured and predicted performance as a function of resolution, blur, and noise for a ResNet-18 model trained and tested on on the Places365 dataset in RGB with the COCO train and test distortions applied respectively . We fitted Eqn. 4.6 to performance on the first of two i.i.d. test datasets evaluated the fit on the second.

# Chapter 6

# Parametric Sensor Modeling

In the preceding chapters, we relied on distortion-based image chains with two key traits. First, the distortions were all applied independently of one another, with down-sampling applied first, followed by a blur function and then a noise function that did not "know" anything about the upstream distortions. Second, the order in which we applied these distortions maximized their cumulative effect on image quality. By downsampling first in our image distortion chain, we magnified the effect of the subsequent blurring. By applying noise last, we avoided the de-noising effects of both down-sampling and blurring.

This approach offered the advantage of simplicity, and it enabled us to demonstrate the distortion-robustness of CNNs under stressing conditions. By applying the distortions independently of one another, however, we ignored coupling inherent in a real image system. To first order in a physical system, ground sample distance (GSD), relative edge response (RER), and signal-to-noise ratio (SNR) are driven simultaneously by the system's aperture diameter, focal length, and pixel pitch. Specifically, GSD is proportional to the ratio of pixel pitch to focal length. Relative edge response is approximately proportional to the ratio of pixel pitch to PSF width (see Ch. 3), and PSF width is driven primarily by aperture diameter assuming the system is near diffraction limited. Finally, in high signal, shot noise limited conditions, SNR is approximately proportional to the ratio of aperture diameter to focal length; in low signal conditions, pixel size, readout noise, and well depth also play a significant role in determining SNR. A more detailed discussion of these relationships appears in Sec. 2.1.2.

For a real system, we would ultimately measure these parameters directly or indirectly during calibration processes that encompass all inter-connected effects. Here, however, we do not enjoy the luxury of an array of calibrated imaging systems with which we can test computer vision performance. Instead, we updated our parametric image chains to incorporate these first order relationships, with the goal of better understanding the

relationships between imaging system design choices and computer vision algorithm performance

## 6.1 Updated image chain

To capture the first order coupling described above in more physically representative parametric image chains, we made several modification to the image chains used in Chapters 4 and 5.

First, we reversed the order of our resolution and blur steps, applying blur first and then downsampling our blurred image in our updated image chains. Physically, this change would be equivalent to adjusting *optical* resolution by changing aperture diameter and then adjusting *sampling* resolution by either (a) changing the focal length and leaving pixel pitch fixed or (b) changing pixel pitch and leaving focal length the same. We chose to treat down-sampling as a result of decreasing focal length for two reasons. First, pixel pitch does not affect SNR to first order under normal imaging conditions without debatable secondary assumptions about the relationship between pixel pitch, well depth, and read noise. Second, pixel pitch is not generally an adjustable parameter during system design; we typically pick a sensor early on and make later optimizations with optical design.

Next, to account for the radiometric impacts of varying aperture size and focal length, we used a simple system model to simulate imaging under varied SNR-conditions. To do so, we referenced all of our images to a baseline system with an f-number, $F_0$, and well depth, $w_0$. We then mapped our blur and down-sampling to relative changes in aperture and focal length, and we simulated the results of these changes on image SNR under varied signal / illumination conditions using the camera equation,

$$E_{detector} = \frac{L_{aperture}}{G\#} = \frac{\pi\tau}{1+4F^2} L_{aperture}, \qquad \text{(2.36 revisited)}$$

to determine the relative change in detector irradiance as a function of changing f-number $F$ and aperture radiance $L_{aperture}$.

To calculate this relative change in detector irradiance, we began by mapping blur to relative aperture diameter using the Gaussian approximation of an Airy diffraction pattern presented by Zhang *et al.* [130]. Zhang *et al.* showed that for a diffraction limited PSF,

$$\sigma^* = 0.21\frac{\lambda}{\text{NA}}, \tag{6.1}$$

where NA is the numerical aperture and $\sigma^*$ is the standard deviation that minimizes the $L_2$ error between a Gaussian approximation and a diffraction limited Airy pattern. We

can rewrite Eqn. 6.1 in terms of aperture and focal length, recognizing that

$$\text{NA} = \frac{d}{2f} = \frac{1}{2F}, \tag{6.2}$$

which yields

$$\sigma^* = 0.42\frac{\lambda f}{d} = 0.42\lambda F \tag{6.3}$$

after substitution for aperture diameter $d$ and focal length $f$. We can therefore write relative aperture size in terms of relative blur kernel standard deviation, with

$$\frac{d\left(\sigma\right)}{d_0\left(\sigma_0\right)} = \frac{\sigma_0}{\sigma}. \tag{6.4}$$

From this relationship, then we know that $F \propto \sigma$.

Next, we can map changes in resolution fraction $r$ to changes in focal length $f$ by recognizing that the instantaneous field of view (IFOV) $\alpha$ of a pixel of pitch $p_0$ is given by

$$\alpha = \frac{p_0}{f}. \tag{6.5}$$

Starting with IFOV $\alpha_0$ before downsampling, we have $\alpha = \frac{\alpha_0}{r}$, and so

$$\alpha = \frac{p_0}{rf_0} = \frac{p_0}{f}, \tag{6.6}$$

meaning $f = rf_0$. Combining the findings of Eqns. 6.4 and 6.6, we have

$$F = \frac{r\sigma F_0}{\sigma_0}. \tag{6.7}$$

We simulated images under the three SNR regimes shown in Table 6.1. We assumed that all of our input images corresponded to a sensor with a 2,000 electron full well, $w_0$. For each SNR regime, we scaled our assumed input signal $L_{ap}$ by fraction $\eta$ and our well depth by a factor of $\sqrt{\eta}$, with well depth in each SNR regime given by

$$w = \sqrt{\eta}w_0. \tag{6.8}$$

Table 6.1 summarizes the signal fractions and well depths that we used.

Next, to keep the dynamic range of our input and output images the same, we stipulated that all signal-adjusted versions of the same image would use the same fraction of the adjusted well depth, with

$$\frac{s}{w} = \frac{s_0}{w_0}, \tag{6.9}$$

Table 6.1: Simulated signal fractions and well depths

| | Signal fraction, $\eta$ | Well depth, $w$ |
|---|---|---|
| Low SNR | 0.0025 | 100 $e^-$ |
| Medium-low SNR | 0.01 | 200 $e^-$ |
| Medium SNR | 0.25 | 1000 $e^-$ |
| High SNR | 1.0 | 2000 $e^-$ |

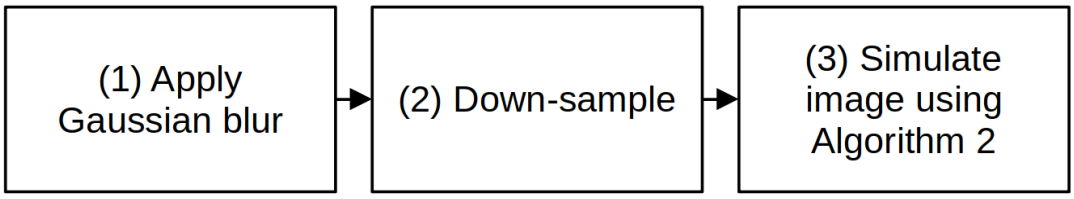



Figure 6.1: Updated image chain steps

which is achieved by adjusting integration time. With our input signal scaled by $\eta$ and our well depth scaled by $\sqrt{\eta}$, we find that

$$\frac{t_{int}}{t_0} = \frac{1+4F^2}{\sqrt{\eta}\left(1+4F_0^2\right)} = \frac{1+4\left(r\frac{\sigma}{\sigma_0}F_0\right)^2}{\sqrt{\eta}\left(1+4F_0^2\right)} \equiv \varphi_t \tag{6.10}$$

In all cases, we assumed a 10 electron standard deviation Gaussian read noise and a dark current $i_{d0}$ that yielded an average dark count $n_{d0} = i_{d0}t_0 = 5\ e^-$ per pixel at baseline integration time $t_0$. Given these relationships, we can determine the expected dark count as a function of signal fraction $\eta$, blur standard deviation $\sigma$, and resolution fraction $r$, with

$$n_d = i_{d0}t_{int} = \frac{n_{d0}}{t_0}t_{int} = \frac{t_{int}}{t_0}n_{d0} = \varphi_t n_{d0}. \tag{6.11}$$

For an 8-bit input image with a signal of $S_{DN}$ at each pixel, then, we begin by applying a Gaussian blur of standard deviation $\sigma$ followed by down-sampling to resolution fraction $r$ (Fig. 6.1). We then converted our 8-bit input signal into electrons, using the relationship

$$s = \frac{S_{DN}}{2^8-1}w. \tag{6.12}$$

Table 6.2: Baseline system parameters

| | |
|---|---|
| dark count,* $n_{d0}$ | 5 $e^-$ |
| read noise, $\sigma_{read}$ | 10 $e^-$ |
| well depth, $w_0$ | 2000 $e^-$ |
| f-number, $F_0$ | 6.5 |
| blur, $\sigma_0$ | 1 pixel |
| pixel pitch, $p_0$ | 1.5 $\mu m$ |
| wavelength, $\lambda$ | 0.55 $\mu m$ |

**electrons per unit integration time $t_0$*

We then recognize this signal $s$ to be the expected number of electrons from the scene represented by the input image. We therefore apply a Poisson distribution with $\lambda_{Poisson} = s$ to account for the photon noise (shot noise) of the image. (We note here that in noising our simulated images, we are not accounting for the native noise of our input images. We highlight that the the COCO dataset generally contains good quality, high SNR images, and we emphasize that our goal is to understand broad trends and first order relationships.) Next, we add Gaussian read noise with a standard deviation of 10 electrons, followed by Poisson distributed dark current, where $\lambda_{Poisson}$ is given by Eqn. 6.11. For the sake of simplicity, we subtract the mean of this dark count to avoid the normalization issues associated with a variable DC offset. Finally, we clip our electron count to fall between 0 and $w$ before converting back to an 8-bit integer by inverting 6.12. Algorithm 2 summarizes the process for scaling the SNR properties of an image based on relative aperture and focal length.

**Algorithm 2** Image distortion chain capturing the first order relationships between aperture, focal length, and SNR.

**Input:** 8-bit image $S_{DN}$ with Gaussian blur $\sigma$ & resolution fraction $r$

**Input:** Baseline image chain parameters well depth $w_0$, signal fraction $\eta$, average dark electron count $n_{d0}$, read noise $\sigma_{read}$, baseline f-number $F_0$ at baseline blur $\sigma_0$

**Output:** Degraded image $S'_{DN}$

$w \leftarrow \sqrt{\eta} \times w_0$
$n_d \leftarrow \varphi_t n_{d0}$, with $\varphi$ given by Eqn. 6.10
$s \leftarrow \frac{S_{DN}}{2^8-1} \times w$, Eqn. 6.12
$s \leftarrow \text{Poisson}(s)$, Eqn. 2.15
$s \leftarrow s + \text{Poisson}(n_d) - n_d$
$s \leftarrow s + \mathcal{N}\left(0, \sigma^2_{read}\right)$
$s \leftarrow \text{clip}(s; 0, w)$
$S'_{DN} \leftarrow \left(2^8 - 1\right) \frac{s}{w}$
**return** $S'_{DN}$

To implement the parametric image chain described above, we needed to establish a baseline f-number $F_0$ from which we could calculate relative changes in required integration time using Eqn. 6.10. Since the COCO dataset is an aggregate of Flickr images captured by varied consumer cameras, we do not have the luxury of single set of fixed sensor parameters such as f-number, pixel pitch, etc. Accordingly, we established a baseline minimum blur $\sigma_0$ and made several simplifying assumptions in order to establish our baseline f-number.

Specifically, we applied a minimum $\sigma_0 = 1$ blur to all of our images, and we used Eqn. 6.3 to map this baseline blur to our baseline f-number $F_0$. The astute observer will notice that this relationship is wavelength dependent and that $\sigma_0$ is in units of pixels. Accordingly, we estimated that the typical sensor in the COCO dataset would have a pixel pitch on the order of a few microns with a peak spectral response near the middle of the visible spectrum. We therefore assigned $1.5\mu m$ and $0.55\mu m$ to our baseline pixel pitch $p_0$ and central wavelength $\lambda$ respectively, yielding a baseline f-number $F_0 = 6.5$.

Figures 6.2, 6.3, 6.4, and 6.5 show the results of this image chain approach. Most importantly, these figures illustrate the coupling present between aperture diameter, focal length, and noise under various signal conditions. Particularly for the low and medium-low SNR cases, we can clearly see SNR decrease with decreasing aperture diameter / increasing blur. We can also see SNR increasing with decreasing focal length / resolution.

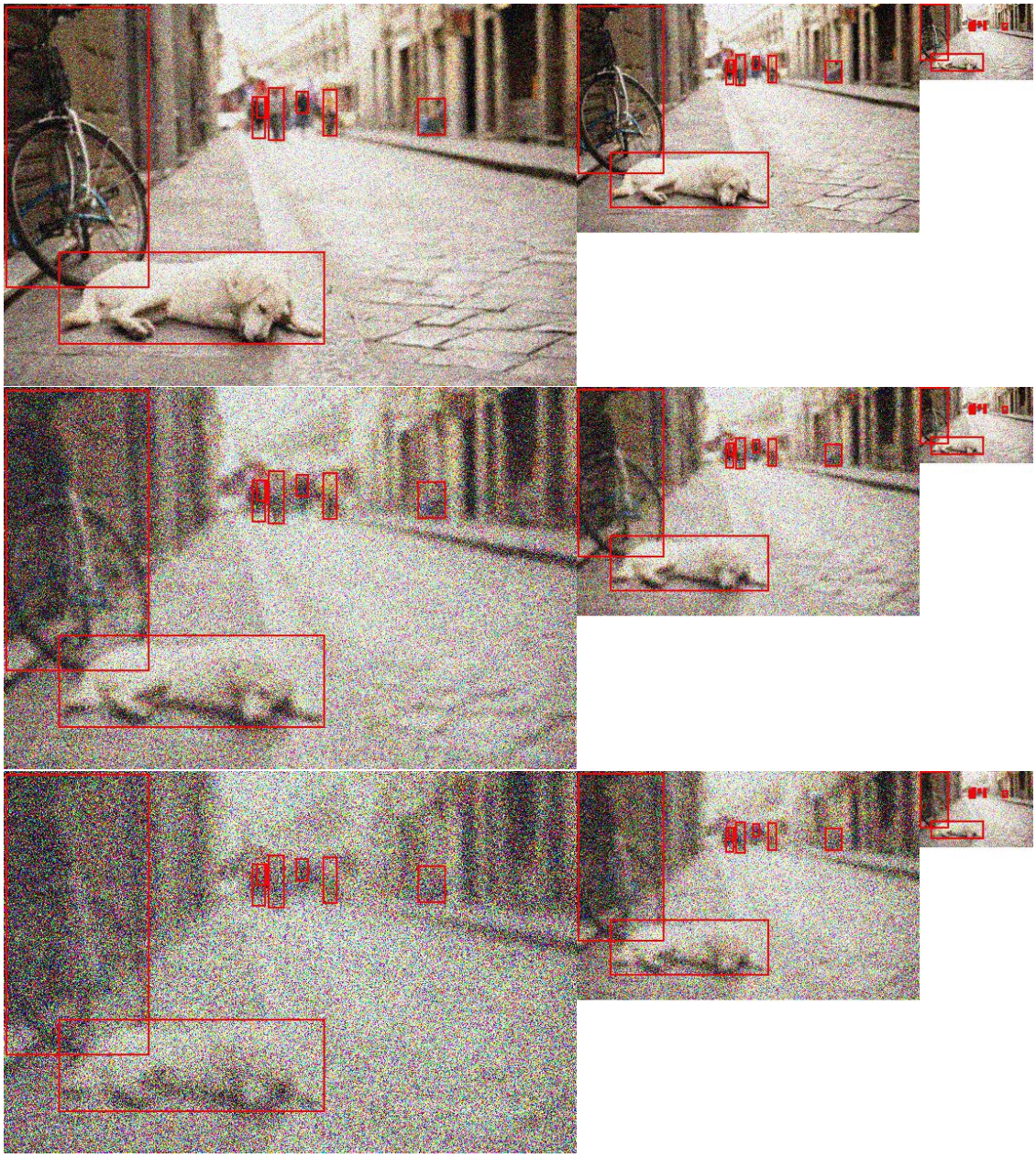


Figure 6.2: Low SNR image, with aperture blur increasing from $\sigma = 1$ (top) to $\sigma = 5$ (bottom) and resolution decreasing from $r = 1$ (left) to $r = 0.2$ (right). Here, we can see the SNR impact of increased diffraction blur due to a shrinking aperture as well as the improvements in SNR that come with decreased resolution due to shortened focal lengths.

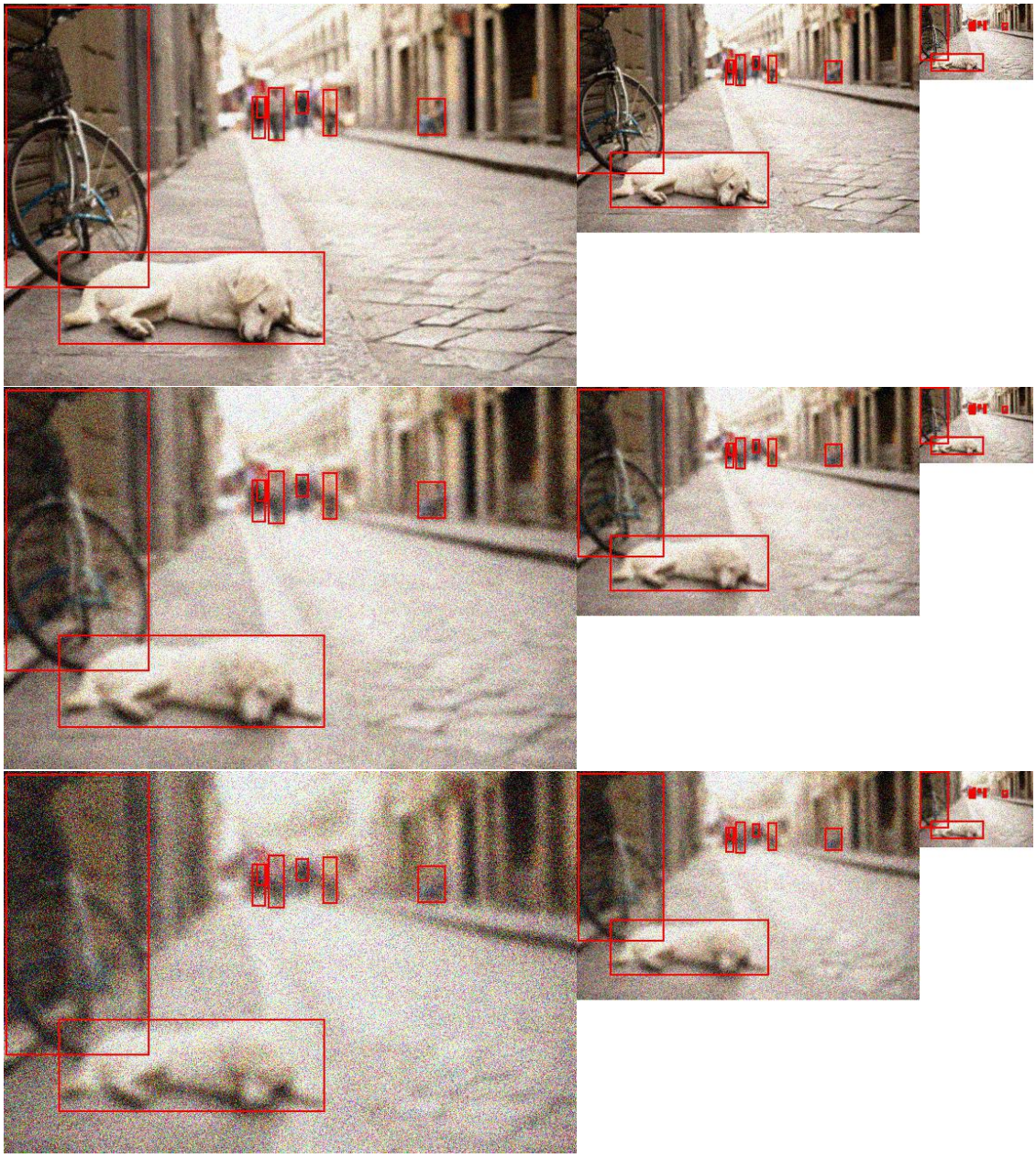


Figure 6.3: Medium-low SNR image, with aperture blur increasing from $\sigma = 1$ (top) to $\sigma = 5$ (bottom) and resolution decreasing from $r = 1$ (left) to $r = 0.2$ (right). Here, at medium SNR we see the differences in blur are apparent, while the SNR effects of aperture and focal length changes are less apparent than at lower SNR.

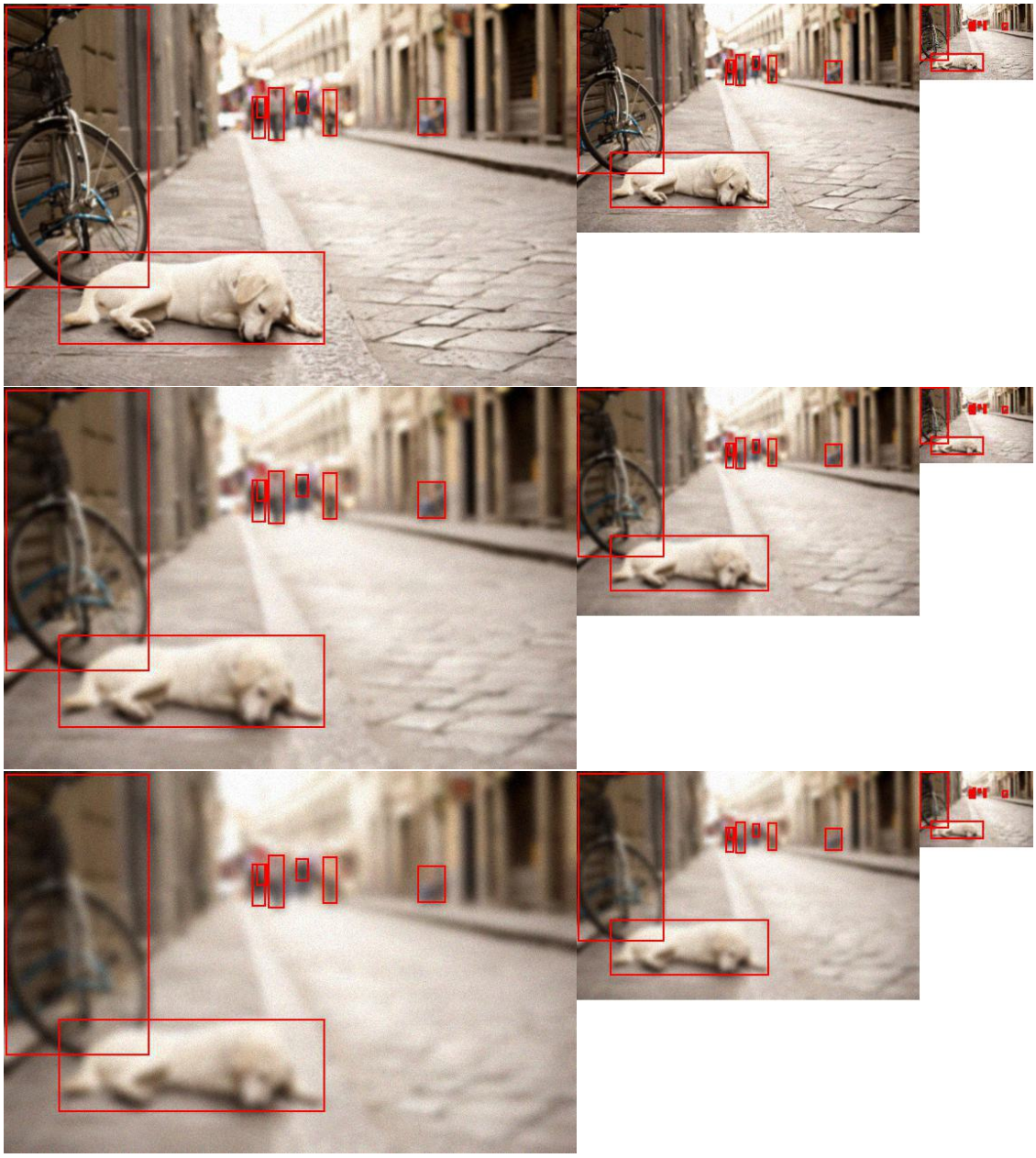


Figure 6.4: Medium SNR image, with aperture blur increasing from $\sigma = 1$ (top) to $\sigma = 5$ (bottom) and resolution decreasing from $r = 1$ (left) to $r = 0.2$ (right). Here, at medium SNR we see the differences in blur are apparent, while the SNR effects of aperture and focal length changes are less apparent than at lower SNR.

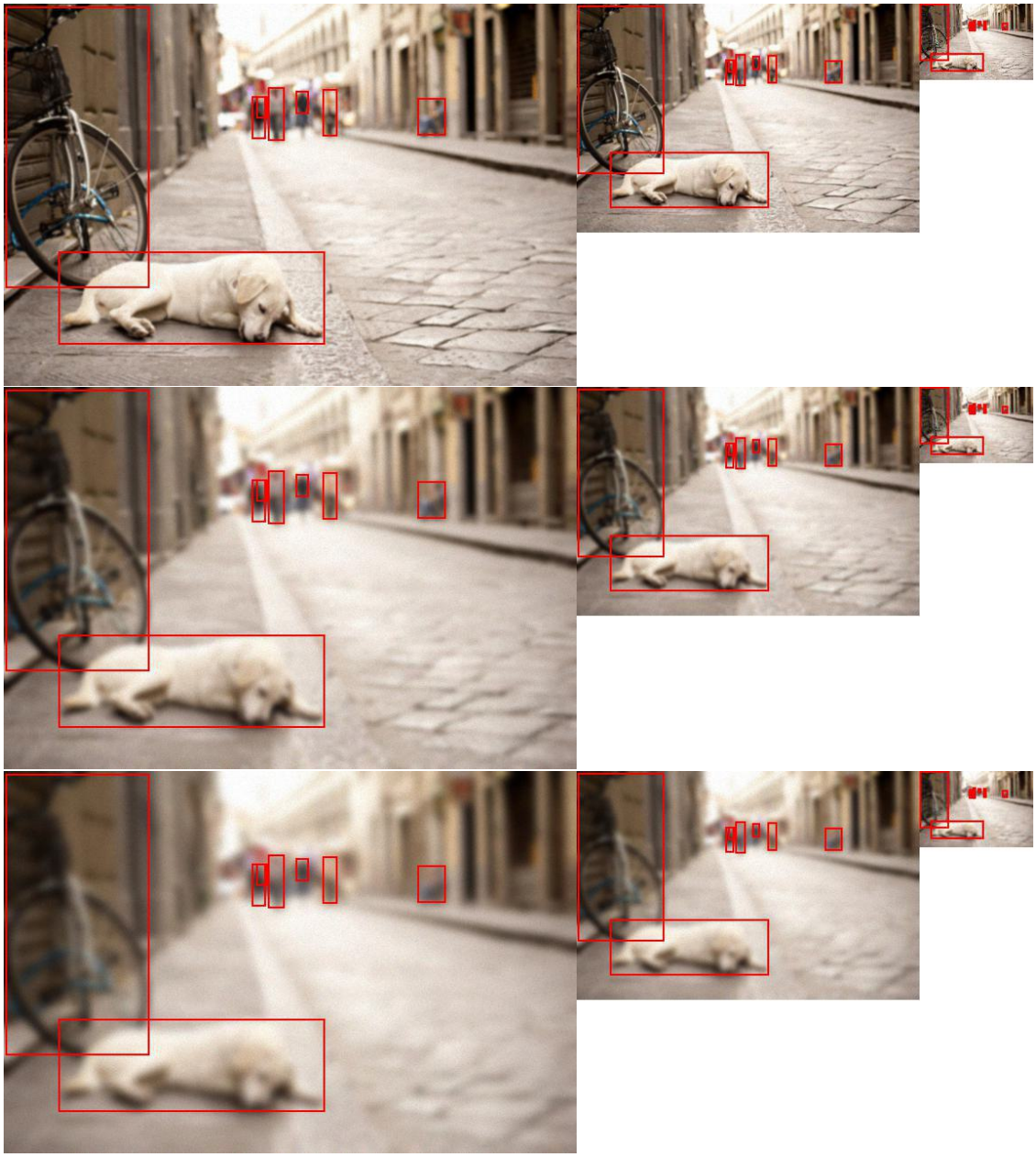


Figure 6.5: High SNR image, with aperture blur increasing from $\sigma = 1$ (top) to $\sigma = 5$ (bottom) and resolution decreasing from $r = 1$ (left) to $r = 0.2$ (right). Here, at high SNR we see the differences in blur are apparent but the SNR effects of aperture and focal length changes are more difficult to discern.

## 6.2 Method

In studying computer vision performance with our updated image chain, we again used the 2017 Common Objects in Context (COCO) dataset and extended most of the methods presented in Chapters 4 and 5. We generated a training and testing dataset for each of the signal fraction and depth combinations in Tab. 6.1.

We modified our resolution and blur distortion bounds slightly from the bounds used in Ch. 5, and here we used the same bounds for both our training and testing (Tab. 6.3). In Chapters 4 and 5, we narrowed our test distortion space slightly when test results for full range trained models were still at or near chance performance at the extremes of the distortion space. Here, because our updated image chain applies blur before down-sampling rather than the opposite, the cumulative effect of down-sampling and blur is decreased because all blurring is applied at the highest original resolution when the pixels remain at their smallest. We used the same training parameters used we used previously in training our YOLOv8l object detection models (Tab. 4.1).

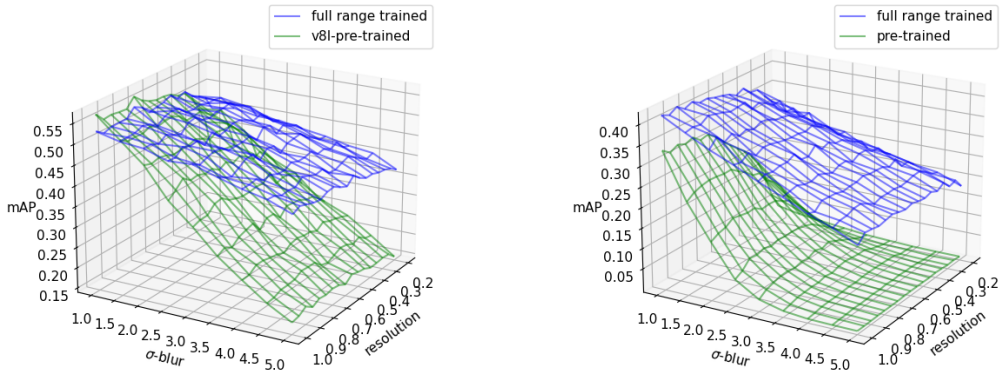
Table 6.3: Full range distortion levels *(train and test)*

| | |
|---|---|
| blur *(pixels)* | 1 - 5 |
| resolution *(fraction)* | 0.2 - 1 |
| noise | Algorithm 2 |

## 6.3 Results

As observed in Chapters 4 and 5, we found that the performance of pre-trained models dropped relatively quickly as a function of blur, with much of the performance recovered by models trained across the full distortion space (Fig. 6.6). In this figure we see that performance as a function of resolution remains comparatively stable over much of the distortion space for both pre-trained and full range trained models for each SNR regime. Conversely, blur strongly impacts the performance of pre-trained models for each SNR case, but training against distorted images mitigates much of this impact.

Figures 6.7 and 6.8 illustrate the differing performance relationships for blur and resolution. In Fig. 6.7, we see a strong relationship between blur and pre-trained model performance for each SNR case, whereas we observe stability in pre-trained model performance as a function of resolution when $r \gtrsim 0.4$. In Fig. 6.8, see that overall performance improves significantly when models are trained on distorted images. We note, however,




(a) High SNR dataset

(b) Low SNR dataset

Figure 6.6: Performance as a function of resolution and blur for pre-trained and full range trained models on the high and low SNR pseudo-system datasets. Full range models are each trained and tested in the same SNR regime.

that the general shapes of the performance curves for blur and resolution remain qualitatively similar for pre-trained and full range trained models. The primary *qualitative* difference we observe is that performance as a function of blur does not drop to chance and level off for our full range trained models in the low and medium-low SNR cases. Additionally, in Fig. 6.8 we observe a steeper performance loss with blur in the low SNR cases than in the high SNR cases; since our model associates blur with diffraction due to a decreased aperture size, we would expect the coupling between aperture and SNR to show up more strongly in a low SNR environment.

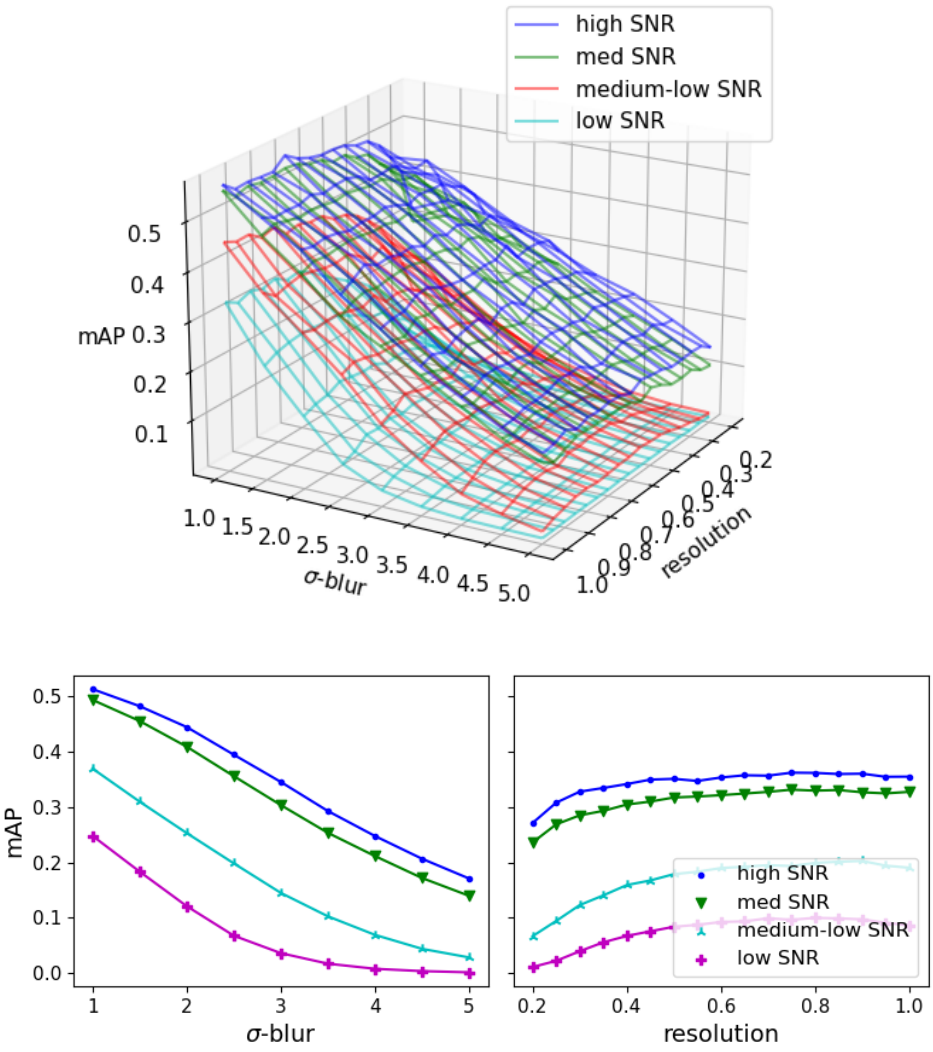



Figure 6.7: Pre-trained model performance for each pseudo-system SNR case.

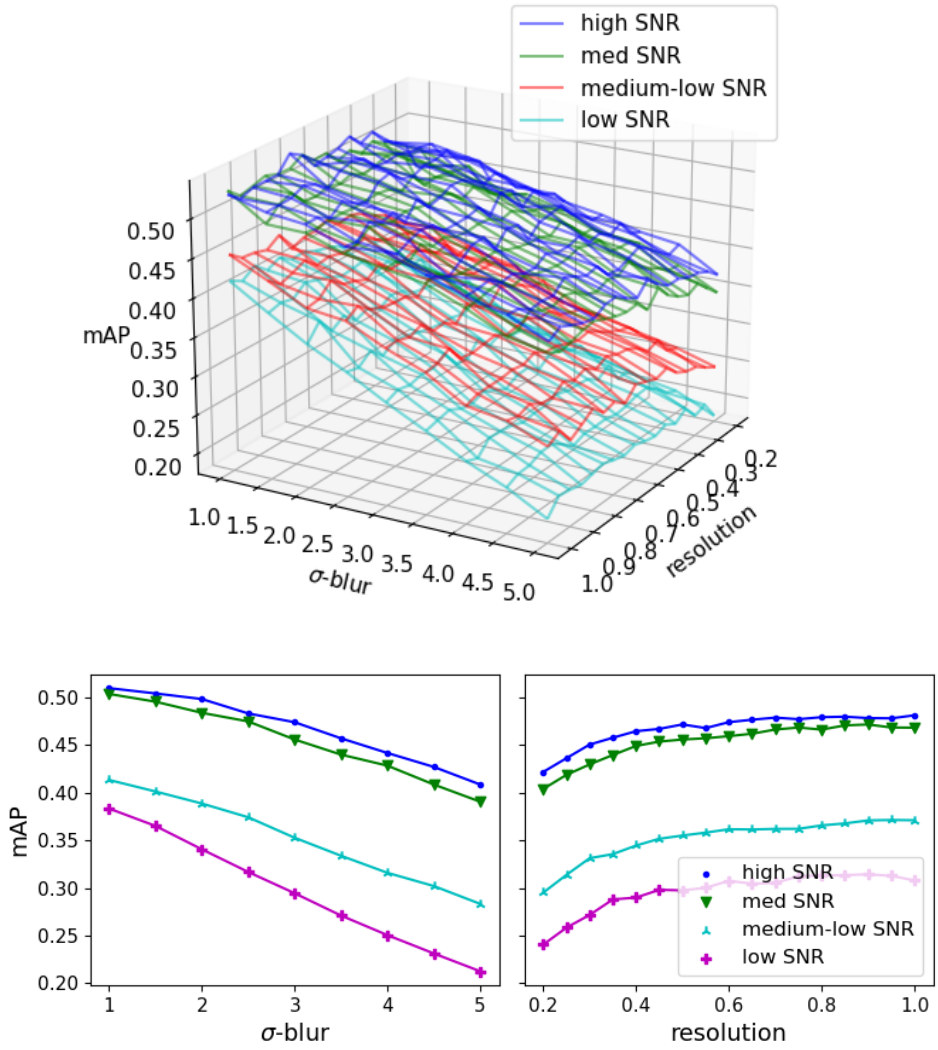
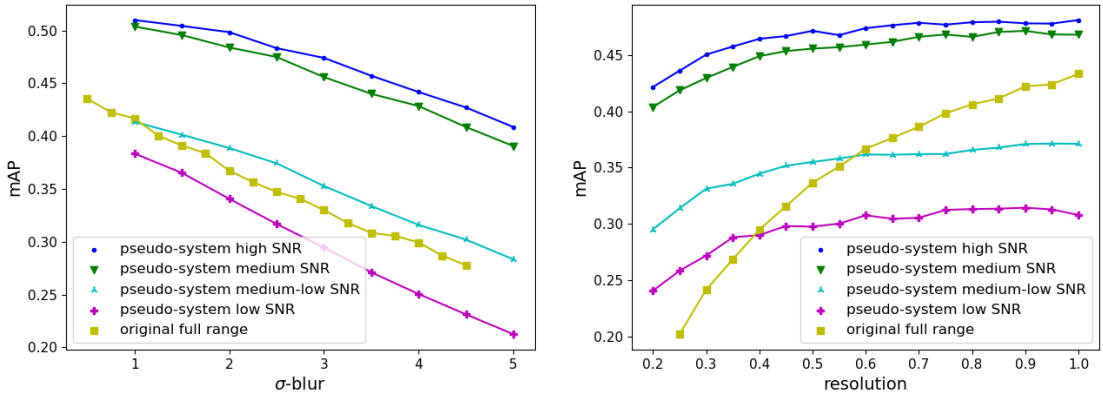



Figure 6.8: Performance of full range trained models for each pseudo-system SNR case. Each full range models is trained trained and tested in the same SNR regime.

## 6.4 Comparison and Discussion

We observe both a significant difference and a significant similarity between the performance relationships that emerged using our original distortion image chain used in Chapters 4 and 5 and our updated pseudo-system image chain. Specifically, we find that performance as a function of blur is qualitatively similar in both instances, while performance as a function of resolution differs significantly. Figures 6.9 and 6.10 illustrate these relationships. Figure 6.9 shows the one-dimensional variations in average performance with blur and resolution, while Fig. 6.10 compares performance as a function of resolution and blur on the original full range test dataset and on the pseudo-system medium-low SNR full range dataset.




(a) Performance as a function of blur

(b) Performance as a function of resolution

Figure 6.9: Performance as a function of blur and resolution for full range trained models on the four pseudo-system datasets and on the original COCO full range test dataset.

In Fig. 6.9a, we see that performance as a function of blur exhibits similar variation across all datasets, which we believe results from separate effects in the original and in the pseudo-system datasets. In the pseudo-system datasets, where blur is modeled as a consequence of decreased aperture size, the blur itself and the associated SNR drop from a smaller aperture both impact performance. In the original full range dataset from Ch. 5, blur and down-sampling amplify each other due to the application of blur *after* down-sampling. For a blur kernel that is agnostic, its effective size is larger on a down-sampled image. Figure 6.10 illustrates the mutual amplification of blur and down-sampling, with performance on the original full range test dataset dropping rapidly toward the low resolution / high blur corner of the distortion space.

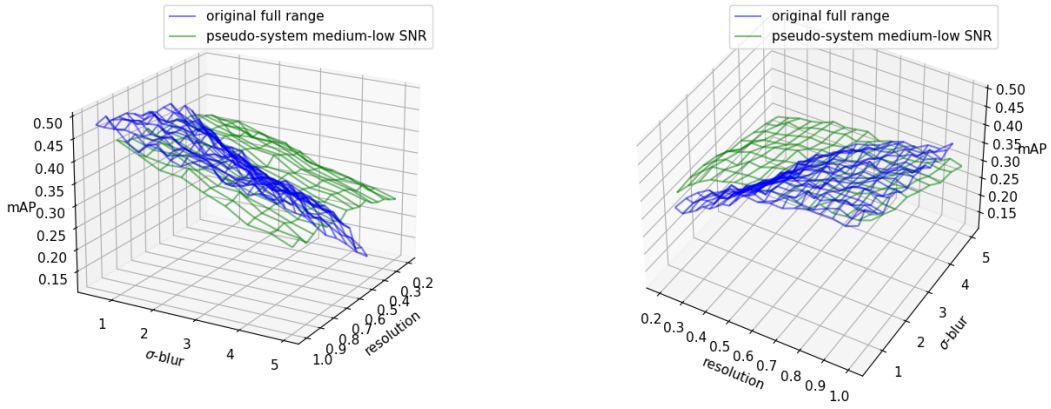



Figure 6.10: Performance of an original full range trained model on the original full range test dataset and performance of a model trained and tested on the on the medium-low SNR pseudo-system train and test test datasets, viewed from two perspectives. Overall average performance on the medium-low SNR pseudo-system dataset was closest to the overall average performance on original full range test dataset.

Conversely, the pseudo-system image chain outlined in Algorithm 2 does not exhibit this mutual reinforcement between resolution and blur due to the application of blur *before* down-sampling. With blur already applied, down-sampling amounts to a sharpening of the final image; the size of the blur kernel as measured in *output* pixels decreases by the resolution fraction $r$. Additionally, in the original distortion image chain, changes in resolution did not appreciably impact final image SNR; noise was specified in DN and applied after down-sampling and blurring, without regard to the blur and resolution levels applied. In the pseudo-system image chain, where changes in resolution are modeled as a consequence of changes in focal length, decreasing resolution *increases* SNR. These differences in the parametric image chains drive the significant differences in performance as a function of resolution observed in Figs. 6.9 and 6.10.

## 6.5 Parametric Sensor Modeling Findings

In Chapters 4 and 5, we observed the impacts of down-sampling, blur, and noise when each of these distortions was applied *agnostically* to the remaining two distortions. Here, by modeling resolution and blur as functions of varied focal length and aperture respectively, we have coupled each to image SNR. While these results are admittedly coarse, they help to map out the performance relationships likely to emerge when tweaking physical image

chain designs.

Specifically, two findings emerge. First, we observe that performance as a function of Gaussian blur is relatively linear with respect to kernel standard deviation $\sigma$, a result largely in line with our findings in previous chapters. Second, we observe that performance as a function of resolution is relatively stable before dropping below resolution fraction $r \approx 0.4$; this result differs from our findings on the full range test datasets used in Chapters 4 and 5 where decreased resolution amplified blur and did not improve SNR, although it is in line with the performance trends we observed when blur was not present in the resolution scan results (see Fig. 5.12). Here, where down-sampling is associated with improved relative edge responses (RER) as measured on down-sampled output pixels and with low noise, we find that performance can be stable across a large resolution range. These results also align with the findings of [91], who found that the optimal resolution for CNN performance on simulated remote sensing images was lower than the GIQE would predict to be optimal for human interpretation.

# Chapter 7

# Summary, Conclusions, and Future Work

## 7.1 Conclusions

In this research, we sought to coarsely map the relationship between image quality and the performance of CNN-based computer vision algorithms. While much of the literature relating to image quality and CNN performance has focused on making CNNs robust to new and unseen distortions, we were particularly interested in understanding how CNNs perform when they have been trained on images of the same quality as those against which they will be evaluated. CNN generalization remains an interesting and important question, but in many cases it is both practical and appropriate to train CNNs on images of similar quality to those that the CNN will see in operations. In these situations, it will be useful to measure the ability of CNNs to "see through" image distortions and to understand the driving relationships between image quality factors and computer vision performance. Additionally, most image quality work to date has focused on making images interpretable to human viewers. As computer vision algorithms become the "users" for images in many applications, it becomes necessary to understand the extent to which historical image quality metrics are appropriate for these new applications. Here, we have attempted to at least start answering some of these questions, with the ultimate goals of both informing follow-on image quality research and informing the design and optimization of imaging systems to be used with computer vision algorithms.

Several broad takeaways emerge from the totality of this research. First, although image quality certainly impacts computer vision performance, CNN-based classifiers and object detectors are capable of performing reasonably well against images of low visual quality with appropriate training. This particular result is not unique in the deep learning

and computer vision literature, but most if not all of the research involving image quality and computer vision performance has focused on image distortions applied *singly rather than in combination*. Our results reinforce the thesis that CNNs can learn to "see through" substantial image distortions and still perform reasonably well. And while much of the research on image quality and CNN performance has focused on strategies to increase CNN robustness to new, untrained distortions, we emphasize that in many computer vision applications the system designer has the luxury of understanding the image quality to be encountered by the CNN with great precision; in these instances, the generalization becomes a largely tangential question and the designer can exploit the capacity of CNNs to perform well when trained on images of similar quality to those against which they will be tested. For systems whose primary end users are CNNs, therefore, the process of design optimization likely should include iterative image simulation, model training, and model evaluation to understand the performance of appropriately trained CNNs against the resulting images at each design iteration.

Second, we find that the functional form of the GIQE is capable of modeling CNN performance with reasonable fidelity, but we observe that other simple, non-physical models invented solely for the sake of comparison do as well or better than the GIQE in modeling performance as a function of image distortion level. We note, however, that the GIQE is an unbounded equation that predicts NIIRS scores, whereas here were are using the *functional form* of the GIQE to predict accuracy and mean average prediction (mAP), which are both bounded between zero and one. Additionally, it is noteworthy that the historical form of the GIQE does a better job of modeling performance as a function of blur / relative edge response in our work with classifiers and grayscale images. We do not have a satisfying explanation for the cause of the poor fit between accuracy and blur for the GIQE-5 functional form, except to note that it results from the inclusion of the quartic RER term (*i.e.* $\text{RER}^4$), which is not present in the historical form of the equation. When we remove the exponent above the independent RER term in GIQE-5, measured and predicted performance as a function of blur match well. We also find that measured and predicted performance as a function of blur agree when we apply the COCO distortions to the Places365 dataset in RGB. The GIQE-5 fitting improvement in RGB may result from the comparatively decreased impact of noise when it is applied to each channel independently rather than being applied to a single-channel grayscale image; this relative decrease in the impact of noise in RGB may affect the behavior of the coupled RER-SNR term in GIQE-5, which would in turn change the relative weight placed on the independent RER term. It is also possible that this poor fit is simply a fitting anomaly, but we observe it on both the SAT-6 and Places365 datasets. Regardless, both the current and historical forms of the GIQE fit object detection performance well.

Third, we find that object detector performance is qualitatively very similar to image

classifier performance in the presence of image distortions. Table 7.1 shows the fit coefficients for our GIQE-based performance predictions on all three datasets used. While the coefficients differ, as we would expect for different datasets and CNN tasks, the coefficients' signs and orders of magnitude are all similar. Although the performance similarity between classification and object detection is not especially surprising, we note that to date we have not found any systematic studies on the relationship between object detection performance and image quality; effectively all of the literature on image quality and deep learning has dealt with image classification. We believe that our object detection results will help to fill this gap in the literature on image quality and computer vision performance.

Table 7.1: Performance prediction fit coefficients from GIQE-5 based model (Eqn. 4.6) and GIQE-3 based model (Eqn. 4.8) for SAT-6 (♦), Places365 (†), and COCO (□). We note that the Places365 and SAT-6 models both predict classification accuracy, while the COCO model predicts mean average precition (mAP).

| | $c_0$ | $c_1$ | $c_2$ | $c_3$ | $c_4$ | $c_5$ | $c_6$ |
|---|---|---|---|---|---|---|---|
| GIQE-5$^{\blacklozenge}$ | 1.02 | $-0.299$ | 0.0453 | $-0.113$ | $-0.0221$ | $-0.278$ | $-2.81 \times 10^{-3}$ |
| GIQE-3$^{\blacklozenge}$ | 1.08 | $-0.336$ | - | - | 0.935 | 0.152 | $-3.05 \times 10^{-3}$ |
| GIQE-5$^{\dagger}$ | 0.512 | $-0.385$ | 0.139 | $-0.0319$ | 1.88 | $-0.0841$ | $-2.25 \times 10^{-4}$ |
| GIQE-3$^{\dagger}$ | 0.688 | $-0.417$ | - | - | 2.25 | 0.272 | $-2.10 \times 10^{-3}$ |
| GIQE-5$^{\square}$ | 0.699 | $-0.464$ | 0.264 | $-2.13$ | 2.20 | $-0.0205$ | $-1.84 \times 10^{-3}$ |
| GIQE-3$^{\square}$ | 0.723 | $-0.465$ | - | - | 2.18 | 0.307 | $-1.82 \times 10^{-3}$ |

Fourth, we observe that the computer vision performance appears to vary relatively smoothly with both blur and noise both for pre-trained models and for models trained on distorted images. Training against distorted images results in substantial performance gains against high blur and high noise images, with generally modest performance losses at low blur and high resolution when models are trained across a wide image quality range. Conversely, performance as a function of resolution is more interesting and less predictable. In our full range test results in Chapters 4 and 5, we applied blur after down-sampling, causing down-sampling to have the effect of amplifying blur. When blur did not follow down-sampling, however, we observed that performance as a function of resolution remained relatively stable above a certain threshold but dropped rapidly around this threshold. This effect was particularly pronounced for datasets in which we modeled resolution as being a consequence of changing focal length, with decreasing resolution and focal length associated with increased SNR. Additionally, we found the performance recovery

associated with training against distorted images to be minimal with respect to resolution. Collectively, our results suggest that decreasing resolution above a certain threshold may do little to harm performance; computer vision systems may be able to function with shorter focal length sensors providing greater fields of view without significant losses in performance. We believe this result could have significant implications across a large number disciplines and merits further investigation.

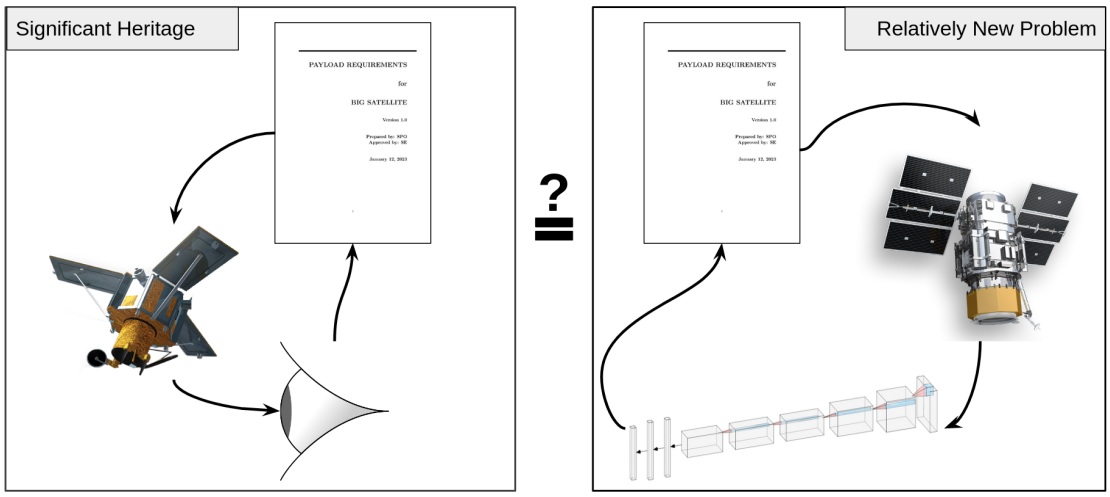



Figure 7.1: Repeat of Fig. 1.1, shown again here to highlight our goal of understanding the transferability of heritage imaging system requirements for current applications reliant on computer vision algorithm performance.

Finally, we note we that all of our results have been task and dataset dependent; they have been fully reliant on having training data representative of the eventual testing images. In Chapters 4 and 5, we created two i.i.d. versions of testing datasets, using the first to fit performance prediction models and the second to evaluate these models. While this approach enabled us to evaluate our performance prediction models with largely different images at each specific point in the distortion space, the underlying images scattered randomly across the distortion space were the same. Accordingly, while we were able to fit and to some extent predict computer vision performance as a function of image quality, we must be circumspect about the general applicability of these results. We have come nowhere near proposing a general, predictive model for computer vision performance as a function of image quality. While such a model may be possible, as hinted by the similarities between the coefficients in Tab. 7.1, a number of assumptions or ground rules for the types of scenes and illumination conditions would likely be necessary.

In short, to predict computer vision performance for a given application as a function of system design, it is still necessary to begin with representative scenes for training and testing computer vision algorithms. We have not presented a method for *a priori* performance performance prediction. Instead, we have shown the types of performance relationships that may emerge and demonstrated an approach for exploring the system trade space. With a high quality initial dataset—one for which subsequent distortions from modeled image chains will significantly out weigh initial image distortions—it may be possible to faithfully simulate representative training and testing images to predicatively evaluate computer vision performance, helping to answer the question posed graphically in Fig. 7.1. We believe that these results can inform such future efforts.

## 7.2 Future Efforts

To acknowledge what is likely apparent to those who have read this far, the research presented here barely scratches the surface of the work required to understand the fundamental relationships between image quality and computer vision performance. A large number of future activities could help to better define image quality as it relates to computer vision performance.

### 7.2.1 Immediate follow on efforts

A number of efforts could be helpful for increasing the fidelity and broadening the applicability of these results.

First, object detection is a data-rich problem. In this effort, we considered only mean average precision (mAP) as a function of image distortion across the full set of COCO object classes. Extending this analysis to consider

- performance at various object sizes
- mAP with different intersection-over-union (IOU) thresholds
- precision and recall as separate metrics across the distortion space

Next, all of the results presented here are dataset dependent. Extending this study to other datasets could significantly enhance the generality of these results. Additionally, extending this work to overhead datasets would be particularly interesting given the importance of image quality in many overhead imaging applications. It would also be helpful to extend this work to datasets with consistent resolution and sensor parameters with metadata to help estimate native SNR. It is plausible that the results obtained from a high quality overhead dataset captured with consistent sensor parameters would differ

significantly from an equivalent autonomous driving dataset; in the first, object range and image quality drivers such as jitter and smear would likely remain relatively consistent across the dataset, while in the second object range, jitter, and smear would all vary significantly across the dataset.

Finally, most of this work relied on distortion-only "image chains" that were not anchored to physical sensor models or the interrelationships between fundamental sensor parameters. Chapter 6 began to address the questions of how distortion performance relationships evolve, but true image chain modeling could significantly increase the fidelity of these results.

### 7.2.2 JPEG Compression-Decompression Layers and Loss Functions

As discussed briefly in Sec. 2.4, the tables used to truncate JPEG DCT coefficients were derived to maximize image quality as perceived visually. It is plausible that different sets of tables could lead to image quality improvements as determined by CNN performance. Additionally, if different tables do indeed produce better compressed image quality for computer vision, the differences between these modified tables and the current HVS-based tables could help to understand differences in the ways that humans perceive images relative to the ways that computer vision algorithms analyze them.

It should be possible to study this problem by using CNN back-propagation rather than through a high dimensional parameter search. Specifically, by building JPEG compression and decompression into the initial layers of a CNN, it should be possible to create a loss function that drives the JPEG compression layer to discard spatial frequencies with less value to the CNN task at hand (*e.g.*, classification) while retaining spatial frequency content of important to the classification task.

JPEG compression works by dividing an image into $8 \times 8$ blocks and performing a DCT on each block, resulting in a total of 64 DCT coefficients for each block. To achieve compression, these DCT are divided by values in a table selected based on the desired compression ratio. After division, the resulting DCT coefficient quotients are rounded to the nearest integer, with many of the quotients rounding to zero. The (most often) high fraction of zeros enables an efficient Huffman coding of the post-division DCT coefficient quotients. To recreate the image, we simply multiply these post-division DCT coefficient quotients by their respective divisors and perform an inverse DCT on the recovered coefficients. To first order (ignoring the rounding errors imparted to the non-zero DCT coefficient quotients), we lose the information at the frequencies whose DCT coefficient quotients rounded to zero and retain the information at the frequencies of quotients not rounded to zero.

To implement a similar process in the initial layers of a CNN, we would begin by hard-coding the 64 $8 \times 8$ DCT filters into the initial layer as convolutional filter weights.

(This layer would not be updated via back-propagation.) With a stride of eight, meaning each convolutional filter (kernel) moves eight pixels at a time rather than the standard one pixel, the output of the first layer of would be a DCT of the image. If we started with a $256 \times 256$ input image, the output of this first layer would be a $32 \times 32 \times 64$ tensor $T$, with each $8 \times 8$ block in the original image fully represented by its 64 coefficients. We could then perform an operation equivalent to division by JPEG table value using what is typically called a $1 \times 1$ convolution layer (where in this case our $1 \times 1$ filter is in reality a $1 \times 1 \times 64$), subtracting a bias value $b$, and applying a rectified linear unit (ReLU) activation function, where

$$\mathrm{ReLU}\,(x) = \max\,(0, x)\,. \tag{7.1}$$

Any outputs less than the subtractive bias $b$ would be set to zero by the ReLU activation function, where the activation is applied element-wise to the $32 \times 32 \times 64$ tensor $T - b$. We could then re-construct the image by performing a $1 \times 1$ convolution with reciprocal weights to those used in the previous $1 \times 1$ convolution and then performing an up-convolution using the same DCT filter weights applied in the first layer.

To drive our JPEG compression layers to actually compress our image representation, we would include a term in a loss function proportional to the sum of the weights in our $1 \times 1$ convolutional filter applied immediately after the DCT. This term would drive the initial filter toward lower valued coefficients. It would be offset by a standard cross-entropy loss term applied to output of the classification portion of the CNN. To vary the compression ratio, we would increase the weight applied to the convolutional filter sum in the combined loss function. In this way, our CNN would learn JPEG tables optimal for computer vision.

To validate our results, we would then compare the performance of a CNN on images compressed with the standard JPEG tables to images compressed with our learned table values.

### 7.2.3 Proposed research program

A substantial follow on research effort would ideally include several key elements:

- Medium fidelity image chain modeling coupled to datasets with well defined meta-data such as resolution / GSD, system MTF, and SNR.
- Anchoring of models with real data collected in a controlled fashion
- Comparisons between human image quality scores and computer vision performance against the same images

- Systematic exploration of the relationship between image chain transfer function and CNN performance

An approach encompassing these elements could begin by simulating realistic imagery starting from a high image quality dataset. Assuming this dataset had good resolution and high SNR, it would be possible to degrade these images in a way that realistically simulated the outputs of a plausible sensor.

Next, it would be useful to build several sensors to anchor the data. To do so, we could design these sensors to operate from a single platform and capture the same imagery simultaneously. In doing so, it would be possible to create labeled data for each sensor with a single collection and labeling effort by labeling one of the datasets and then transferring the labels to the the images from the other sensors. Doing so would allow the properties of the sensors to drive the image quality differences between the datasets. Finally, we could test humans' abilities to classify or label a subset of these images.

Finally, to better understand the results of the more detailed image chain simulations' performance results, it could be informative to systematically investigate the relationship between CNN performance and image train transfer function. Rather than considering blur to be a function of point spread function or Gaussian kernel, a Fourier-first approach that isolates the spatial frequency content of the output images would help to inform a fundamental understanding on what image information content can best drive performance. To start, such an approach could apply simple band-pass filters to the images' Fourier transforms and evaluate performance as a function of bandpass center frequency and bandwidth.

# Bibliography


[1] Rayleigh, L., "Xxxi. investigations in optics, with special reference to the spectroscope," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* **8**, 261–274 (1879).

[2] Rayleigh, L., "Xv. on the theory of optical images, with special reference to the microscope," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* **42**, 167–195 (1896).

[3] Robinson, T. R., "On the construction of the cassegrain telescope," *Proceedings of the Royal Irish Academy (1836-1869)* **6**, 20–28 (1853).

[4] Ives, H. E., "Television1," *Bell System Technical Journal* **6**, 551–559 (10 1927).

[5] Rayton, W. B. and Cook, A. A., "The effect of aberrations upon image quality," *Journal of the Society of Motion Picture Engineers* **28**, 377–387 (1937).

[6] Eskicioglu, A. M. and Fisher, P. S., "Image quality measures and their performance," *IEEE Transactions on Communications* **43**, 2959–2965 (1995).

[7] Kamble, V. and Bhurchandi, K. M., "No-reference image quality assessment algorithms: A survey," *Optik* **126**, 1090–1097 (6 2015).

[8] Fiete, R. D., "Image quality and [lambda] fn/p for remote sensing systems," *Optical Engineering* **38**, 1229–1240 (1999).

[9] Krizhevsky, A., Sutskever, I., and Hinton, G. E., "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems* **25**, 1097–1105 (2012).

[10] Gong, Y., Wang, L., Guo, R., and Lazebnik, S., "Multi-scale orderless pooling of deep convolutional activation features," *European conference on computer vision* , 392–407 (2014).

[11] Zhu, H., Tang, P., Park, J., Park, S., and Yuille, A., "Robustness of object recognition under extreme occlusion in humans and computational models," *arXiv preprint arXiv:1905.04598* (2019).

[12] Wang, A., Sun, Y., Kortylewski, A., and Yuille, A. L., "Robust object detection under occlusion with context-aware compositionalnets," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* , 12645–12654 (2020).

[13] Leachtenauer, J. C., Malila, W., Irvine, J., Colburn, L., and Salvaggio, N., "General image-quality equation: Giqe," *Applied optics* **36**, 8322–8328 (1997).

[14] Harrington, L., Blanchard, D., Salacain, J., Smith, S., and Amanik, P., "General image quality equation: Giqe version 5," *Nat. Geospatial-Intell. Agency, Fort Belvoir, VA, USA, Tech. Rep* (2015).

[15] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L., "Imagenet: A large-scale hierarchical image database," 248–255 (3 2010).

[16] Fiete, R. D. and Tantalo, T. A., "Comparison of snr image quality metrics for remote sensing systems," *https://doi.org/10.1117/1.1355251* **40**, 574–585 (4 2001).

[17] Schott, J. R., Gerace, A., Woodcock, C. E., Wang, S., Zhu, Z., Wynne, R. H., and Blinn, C. E., "The impact of improved signal-to-noise ratios on algorithm performance: Case studies for landsat class instruments," *Remote Sensing of Environment* **185**, 37–45 (11 2016).

[18] Eskicioglu, A. M. and Fisher, P. S., "A survey of quality measures for gray scale image compression," *Proc. 1993 space and earth science data compression workshop, Utah, 1993* , 49–61 (1993).

[19] Avcibas, I., Sankur, B., and Sayood, K., "Statistical evaluation of image quality measures," *Journal of Electronic Imaging* **11**, 206–223 (4 2002).

[20] Jr, R. L. E., [*Fourier methods in imaging*], John Wiley & Sons (2010).

[21] Goodman, J., [*Introduction to Fourier Optics*], McGraw-Hill, international ed ed. (1996).

[22] Carrere, J. P., Place, S., Oddou, J. P., Benoit, D., and Roy, F., "Cmos image sensor: Process impact on dark current," *IEEE International Reliability Physics Symposium Proceedings* (2014).

[23] Singh, K., "Noise analysis of a fully integrated cmos image sensor," *https://doi.org/10.1117/12.342862* **3650**, 44–51 (3 1999).

[24] Starkey, D. A. and Fossum, E. R., "Determining conversion gain and read noise using a photon-counting histogram method for deep sub-electron read noise image sensors," *IEEE Journal of the Electron Devices Society* **4**, 129–135 (5 2016).

[25] Cochrane, A., Schulz, K., Bell, R., and Kendrick, R., "Q selection for an electro-optical earth imaging system: theoretical and experimental results," *Optics Express, Vol. 21, Issue 19, pp. 22124-22138* **21**, 22124–22138 (9 2013).

[26] Ientilucci, E. J. and Schott, J. R., [*Radiometry and Radiation Propagation*]. unpublished at time of access, to be publshed by Oxford University Press, 2023.

[27] Cochrane, A., Schulz, K., Kendrick, R., and Bell, R., "Mtf and integration time versus fill factor for sparse-aperture imaging systems," **40**, 13 (2001).

[28] Galili, I., "Optical image and vision: From pythagoras to kepler," *Science: Philosophy, History and Education* , 103–144 (2021).

[29] Spillmann, L., "Receptive fields of visual neurons: The early years," *Perception* **43**, 1145–1176 (1 2014).

[30] "Spatial contrast sensitivity — metropsis."

[31] KUFFLER, S. W., "Discharge patterns and functional organization of mammalian retina," *https://doi.org/10.1152/jn.1953.16.1.37* **16**, 37–68 (1 1953).

[32] Hubel, D. H. and Wiesel, T. N., "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *The Journal of Physiology* **160**, 106 (1 1962).

[33] Rosenblatt, F., "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review* **65**, 386–408 (11 1958).

[34] Kanan, C., "Imgs 682 spring 2021 lecture 5 neural networks part 1," (2021).

[35] LeNail, A., "Nn-svg: Publication-ready neural network architecture schematics.," *J. Open Source Softw.* **4**, 747 (2019).

[36] "File:artificialneuronmodel english.png - wikimedia commons."

[37] Rumelhart, D. E., Hinton, G. E., and Williams, R. J., "Learning representations by back-propagating errors," *Nature 1986 323:6088* **323**, 533–536 (1986).

[38] Zeiler, M. D., "Adadelta: An adaptive learning rate method," (12 2012).

[39] Kingma, D. P. and Ba, J. L., "Adam: A method for stochastic optimization," *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings* (12 2014).

[40] Loshchilov, I. and Hutter, F., "Decoupled weight decay regularization," *7th International Conference on Learning Representations, ICLR 2019* (11 2017).

[41] Polyak, B. T. and Juditsky, A. B., "Acceleration of stochastic approximation by averaging," *SIAM Journal on Control and Optimization* **30**, 838–855 (7 1992).

[42] Dozat, T., "Incorporating nesterov momentum into adam," (2 2016).

[43] Riedmiller, M., Riedmiller, M., and Braun, H., "A direct adaptive method for faster backpropagation learning: The rprop algorithm," *IEEE INTERNATIONAL CONFERENCE ON NEURAL NETWORKS* **16**, 586–591 (1993).

[44] Sutskever, I., Martens, J., Dahl, G., and Hinton, G., "On the importance of initialization and momentum in deep learning," (2013).

[45] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D., "Backpropagation applied to handwritten zip code recognition," *Neural Computation* **1**, 541–551 (12 1989).

[46] Cun, L., Henderson, J., Cun, Y. L., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D., "Handwritten digit recognition with a backpropagation network," *Advances in Neural Information Processing Systems* **2** (1989).

[47] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P., "Gradient-based learning applied to document recognition," *Proceedings of the IEEE* **86**, 2278–2323 (1998).

[48] Krizhevsky, A., "Learning multiple layers of features from tiny images," (2009).

[49] Bergstrom, A., "Homework 2, imgs 682 spring 2021," (2021).

[50] Geirhos, R., Temme, C. R. M., Rauber, J., Schütt, H. H., Bethge, M., and Wichmann, F. A., "Generalisation in humans and deep neural networks," *CoRR* **abs/1808.08750** (2018).

[51] Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F. A., and Brendel, W., "Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness.," *International Conference on Learning Representations* (2019).

[52] Geirhos, R., Jacobsen, J. H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., and Wichmann, F. A., "Shortcut learning in deep neural networks," *Nature Machine Intelligence 2020 2:11* **2**, 665–673 (11 2020).

[53] Sheikh, H. R. and Bovik, A. C., "Image information and visual quality," *IEEE Transactions on Image Processing* **15**, 430–444 (2 2006).

[54] Wang, Z. and Bovik, A. C., "A universal image quality index," *IEEE Signal Processing Letters* **9**, 81–84 (3 2002).

[55] Wang, Z., Bovik, A. C., and Lu, L., "Why is image quality assessment so difficult?," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings* **4**, 3313–3316 (2002).

[56] Diamant, E., "Searching for image information content, its discovery, extraction, and representation," *Journal of Electronic Imaging* **14**, 13016 (2005).

[57] Yu, H. and Winkler, S., "Image complexity and spatial information," *2013 Fifth International Workshop on Quality of Multimedia Experience (QoMEX)* , 12–17 (2013).

[58] Hosseini, R., Sinz, F., and Bethge, M., "Lower bounds on the redundancy of natural images," *Vision research* **50**, 2213–2222 (2010).

[59] Larkin, K. G., "Reflections on shannon information: In search of a natural information-entropy for images," *arXiv preprint arXiv:1609.01117* (2016).

[60] Shannon, C. E., "A mathematical theory of communication," *The Bell system technical journal* **27**, 379–423 (1948).

[61] Singh, K., Sandu, A., Jardak, M., Bowman, K. W., and Lee, M., "A practical method to estimate information content in the context of 4d-var data assimilation," *SIAM/ASA Journal on Uncertainty Quantification* **1**, 106–138 (2013).

[62] Liu, L., Liu, B., Huang, H., and Bovik, A. C., "No-reference image quality assessment based on spatial and spectral entropies," *Signal Processing: Image Communication* **29**, 856–863 (2014).

[63] Sheikh, H. R., Bovik, A. C., and Veciana, G. D., "An information fidelity criterion for image quality assessment using natural scene statistics," *IEEE Transactions on image processing* **14**, 2117–2128 (2005).

[64] Tsai, D.-Y., Lee, Y., and Matsuyama, E., "Information entropy measure for evaluation of image quality," *Journal of digital imaging* **21**, 338–347 (2008).

[65] Wainwright, M. J., Simoncelli, E. P., and Willsky, A. S., "Random cascades on wavelet trees and their use in analyzing and modeling natural images," *Applied and Computational Harmonic Analysis* **11**, 89–123 (7 2001).

[66] Sheikh, H. R., Sabir, M. F., and Bovik, A. C., "A statistical evaluation of recent full reference image quality assessment algorithms," *IEEE Transactions on Image Processing* **15**, 3440–3451 (11 2006).

[67] Dhillon, A. and Verma, G. K., "Convolutional neural network: a review of models, methodologies and applications to object detection," *Progress in Artificial Intelligence 2019 9:2* **9**, 85–112 (12 2019).

[68] Ajit, A., Acharya, K., and Samanta, A., "A review of convolutional neural networks," *International Conference on Emerging Trends in Information Technology and Engineering, ic-ETITE 2020* (2 2020).

[69] Li, Z., Liu, F., Yang, W., Peng, S., and Zhou, J., "A survey of convolutional neural networks: Analysis, applications, and prospects," *IEEE Transactions on Neural Networks and Learning Systems* , 1–21 (6 2021).

[70] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A., "Towards deep learning models resistant to adversarial attacks," *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings* (6 2017).

[71] Arnab, A., Miksik, O., and Torr, P. H., "On the robustness of semantic segmentation models to adversarial attacks," (2018).

[72] Akhtar, N. and Mian, A., "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access* **6**, 14410–14430 (2 2018).

[73] Qiu, S., Liu, Q., Zhou, S., and Wu, C., "Review of artificial intelligence adversarial attack and defense technologies," *Applied Sciences 2019, Vol. 9, Page 909* **9**, 909 (3 2019).

[74] Akhtar, N., Mian, A., Kardan, N., and Shah, M., "Advances in adversarial attacks and defenses in computer vision: A survey," *IEEE Access* **9**, 155161–155196 (2021).

[75] Zanjani, F. G., Zinger, S., Piepers, B., Mahmoudpour, S., Schelkens, P., and de With, P. H. N., "Impact of jpeg 2000 compression on deep convolutional neural

networks for metastatic cancer detection in histopathological images," *Journal of Medical Imaging* **6**, 1 (4 2019).

[76] Duan, L. Y., Liu, X., Chen, J., Huang, T., and Gao, W., "Optimizing jpeg quantization table for low bit rate mobile visual search," *2012 IEEE Visual Communications and Image Processing, VCIP 2012* (2012).

[77] Chao, J., Chen, H., and Steinbach, E., "On the design of a novel jpeg quantization table for improved feature detection performance," *2013 IEEE International Conference on Image Processing, ICIP 2013 - Proceedings* , 1675–1679 (2013).

[78] Liu, Z., Liu, T., Wen, W., Jiang, L., Xu, J., Wang, Y., and Quan, G., "Deepn-jpeg: A deep neural network favorable jpeg-based image compression framework," *Proceedings of the 55th Annual Design Automation Conference* (2018).

[79] Li, Z., Sa, C. D., and Sampson, A., "Optimizing jpeg quantization for classification networks," (3 2020).

[80] Dodge, S. and Karam, L., "Understanding how image quality affects deep neural networks," *2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX)* , 1–6 (2016).

[81] Dodge, S. and Karam, L., "A study and comparison of human and deep learning recognition performance under visual distortions," *2017 26th International Conference on Computer Communications and Networks, ICCCN 2017* (9 2017).

[82] Hendrycks, D. and Dietterich, T., "Benchmarking neural network robustness to common corruptions and perturbations," *7th International Conference on Learning Representations, ICLR 2019* , International Conference on Learning Representations, ICLR (2019).

[83] Hendrycks, D., Mu, N., Cubuk, E. D., Zoph, B., Gilmer, J., and Lakshminarayanan, B., "Augmix: A simple data processing method to improve robustness and uncertainty," (12 2019).

[84] Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., Song, D., Steinhardt, J., and Gilmer, J., "The many faces of robustness: A critical analysis of out-of-distribution generalization."

[85] Vasiljevic, I., Chakrabarti, A., and Shakhnarovich, G., "Examining the impact of blur on recognition by convolutional networks," *arXiv preprint arXiv:1611.05760* (2016).

[86] Rawat, W. and Wang, Z., "Deep convolutional neural networks for image classification: A comprehensive review," *Neural computation* **29**, 2352–2449 (2017).

[87] Borkar, T. S. and Karam, L. J., "Deepcorrect: Correcting dnn models against image distortions," *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society* **28**, 6022–6034 (12 2019).

[88] Schneider, S., Rusak, E., Eck, L., Bringmann, O., Brendel, W., and Bethge, M., "Improving robustness against common corruptions by covariate shift adaptation," *Advances in Neural Information Processing Systems* **2020-December**, Neural information processing systems foundation (2020).

[89] Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., and Le, Q. V., "Autoaugment: Learning augmentation policies from data," *Cvpr 2019* , 113–123 (5 2018).

[90] Yin, D., Lopes, R. G., Shlens, J., Cubuk, E. D., and Gilmer, J., "A fourier perspective on model robustness in computer vision," *Advances in Neural Information Processing Systems* **32**, Neural information processing systems foundation (2019).

[91] Jaffe, L., Zelinski, M., and Sakla, W., "Remote sensor design for visual recognition with convolutional neural networks," *IEEE Transactions on Geoscience and Remote Sensing* **57**, 9090–9108 (11 2019).

[92] Diamond, S., Sitzmann, V., Julca-Aguilar, F., Boyd, S., Wetzstein, G., and Heide, F., "Dirty pixels: Towards end-to-end image processing and perception," *ACM Transactions on Graphics* **40** (5 2021).

[93] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S., "Pytorch: An imperative style, high-performance deep learning library," *Advances in Neural Information Processing Systems 32* , 8024–8035, Curran Associates, Inc. (2019).

[94] Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, I., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and Contributors, S. ., "Scipy 1.0: Fundamental algorithms for scientific computing in python," *Nature Methods* **17**, 261–272 (2020).

[95] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X., "Tensorflow: Large-scale machine learning on heterogeneous systems," (2015). Software available from tensorflow.org.

[96] Bergstrom, A. C., Conran, D., and Messinger, D. W., "Gaussian blur and relative edge response," (1 2023).

[97] Burns, P. D. et al., "Slanted-edge mtf for digital camera and scanner analysis," *Is and Ts Pics Conference* , 135–138 (2000).

[98] Bergstrom, A., "relative-edge-response," (1 2022). Available at https://github.com/acb08/relative-edge-response.

[99] Conran, D., "High fidelity psf/mtf simulations for remote sensing imagers - grss-ieee." Available at https://www.grss-ieee.org/events/high-fidelity-psf-mtf-simulations-for-remote-sensing-imagers/.

[100] Weisstein, E. W., "Lorentzian function. from mathworld—a wolfram web resource." Last visited on 12/15/2022.

[101] Basu, S., Ganguly, S., Mukhopadhyay, S., DiBiano, R., Karki, M., and Nemani, R., "Deepsat: a learning framework for satellite imagery," *Proceedings of the 23rd SIGSPATIAL international conference on advances in geographic information systems* , 1–10 (2015).

[102] Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., and Torralba, A., "Places: A 10 million image database for scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **40**, 1452–1464 (6 2018).

[103] RIT, "Research computing services," (2019).

[104] "Durbin-watson statistic." [Online, https://en.wikipedia.org/wiki/Durbin-Watson_statistic, Accessed: 2023-05-10].

[105] Perktold, J., Seabold, S., and Taylor, J., "statsmodels." [Online, https://www.statsmodels.org/dev/generated/statsmodels.stats.stattools.durbin_watson.html, Accessed: 2023-05-10].

[106] Banks, H. T. and Joyner, M. L., "Aic under the framework of least squares estimation," *Applied Mathematics Letters* **74**, 33–45 (12 2017).

[107] Geirhos, R., Narayanappa, K., Mitzkus, B., Thieringer, T., Bethge, M., Wichmann, F. A., and Brendel, W., "Partial success in closing the gap between human and machine vision," *Advances in Neural Information Processing Systems* **34**, 23885–23899 (12 2021).

[108] Mayer, J., Khairy, K., and Howard, J., "Drawing an elephant with four complex parameters," *Citation: American Journal of Physics* **78**, 648 (2010).

[109] Dyson, F., "A meeting with enrico fermi," *Nature* **427**, 297 (1 2004).

[110] Girshick, R., Donahue, J., Darrell, T., and Malik, J., "Rich feature hierarchies for accurate object detection and semantic segmentation," *Proceedings of the IEEE conference on computer vision and pattern recognition* , 580–587 (2014).

[111] Girshick, R., "Fast r-cnn," *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (12 2015).

[112] Ren, S., He, K., Girshick, R., and Sun, J., "Faster r-cnn: Towards real-time object detection with region proposal networks," (6 2015).

[113] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A., "You only look once: Unified, real-time object detection," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* **2016-December**, 779–788 (6 2015).

[114] Hsiang, H., Chen, K. C., Li, P. Y., and Chen, Y. Y., "Analysis of the effect of automotive ethernet camera image quality on object detection models," *2020 International Conference on Artificial Intelligence in Information and Communication, ICAIIC 2020* , 021–026 (2 2020).

[115] Nath, N. and Behzadan, A. H., "Deep generative adversarial network to enhance image quality for fast object detection in construction sites," *Proceedings - Winter Simulation Conference* **2020-December**, 2447–2459 (12 2020).

[116] Choi, J., Chun, D., Kim, H., and Lee, H.-J., "Gaussian yolov3: An accurate and fast object detector using localization uncertainty for autonomous driving," (2019).

[117] Xu, H., Gao, Y., Yu, F., and Darrell, T., "End-to-end learning of driving models from large-scale video datasets," (2017).

[118] Dixit, A., Chidambaram, R. K., and Allam, Z., "Safety and risk analysis of autonomous vehicles using computer vision and neural networks," *Vehicles 2021, Vol. 3, Pages 595-617* **3**, 595–617 (9 2021).

[119] Ren, S., Hu, W., Bradbury, K., Harrison-Atlas, D., Valeri, L. M., Murray, B., and Malof, J. M., "Automated extraction of energy systems information from remotely sensed data: A review and analysis," *Applied Energy* **326**, 119876 (11 2022).

[120] Bergstrom, A. C. and Messinger, D. W., "Understanding the relationship between image quality and convolutional neural network performance," *Pattern Recognition and Tracking XXXIII* **12101**, 10–24 (2022).

[121] Bergstrom, A. C. and Messinger, D. W., "Image quality and computer vision performance: assessing the effects of image distortions and modeling performance relationships using the general image quality equation," *https://doi.org/10.1117/1.JEI.32.2.023018* **32**, 023018 (3 2023).

[122] Kong, L., Ikusan, A., Dai, R., and Zhu, J., "Blind image quality prediction for object detection," *Proceedings - 2nd International Conference on Multimedia Information Processing and Retrieval, MIPR 2019* , 216–221 (4 2019).

[123] Kong, L., Ikusan, A., Dai, R., and Ros, D., "An image quality adjustment framework for object detection on embedded cameras," *International Journal of Multimedia Data Engineering and Management (IJMDEM)* **12**, 39–57 (2021).

[124] Bergstrom, A. C. and Messinger, D. W., "Image quality and object detection performance of convolutional neural networks," *https://doi.org/10.1117/12.2663779* **12527**, 159–177 (6 2023).

[125] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L., "Microsoft coco: Common objects in context," *European conference on computer vision* , 740–755 (2014).

[126] Jocher, G., Chaurasia, A., and Qiu, J., "Yolo by ultralytics," (4 2023).

[127] tanakawho, "Horse show." [Online, http://farm4.staticflickr.com/3250/2883102207_bcba5527a7_z.jpg, accessed April 25, 2023, Attribution-NonCommercial 3.0 Unported (CC BY-NC 3.0), https://creativecommons.org/licenses/by-nc/3.0/)].

[128] Everingham, M., Gool, L. V., Williams, C. K. I., Winn, J., Zisserman, A., Everingham, M., Leuven, L. K. V. G., Williams, B. C., Winn, J., and Zisserman, A., "The pascal visual object classes (voc) challenge," *Int J Comput Vis* **88**, 303–338 (2010).

[129] Pier, J., "White lab on a street in florence." [Online, http://farm5.staticflickr.com/4087/5078192399_aaefdb5074_z.jpg, accessed April 25, 2023, Attribution-NonCommercial 3.0 Unported (CC BY-NC 3.0), https://creativecommons.org/licenses/by-nc/3.0/)].

[130] Zhang, B., Zerubia, J., and Olivo-Marin, J.-C., "Gaussian approximations of fluorescence microscope point-spread function models," *APPLIED OPTICS* **46** (2007).

[131] Bergstrom, A., "image-quality-for-deep-learning," (8 2022). Available at https://github.com/acb08/image-quality-for-deep-learning.

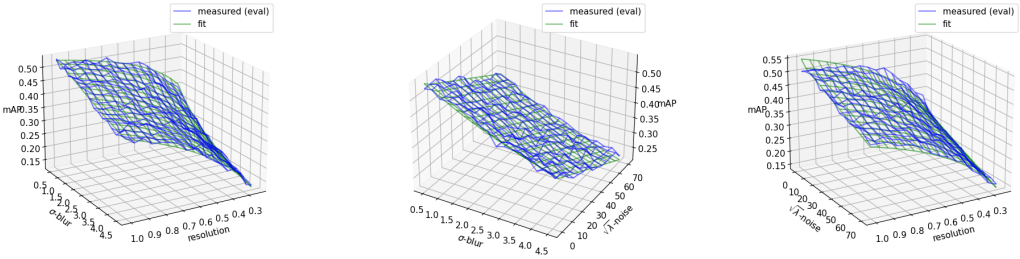
# Chapter 8

# Appendices

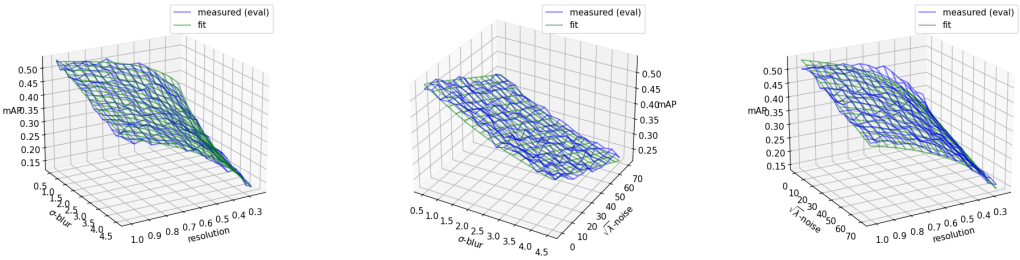
## 8.1 Additional Fit Plots

### 8.1.1 COCO fit plots

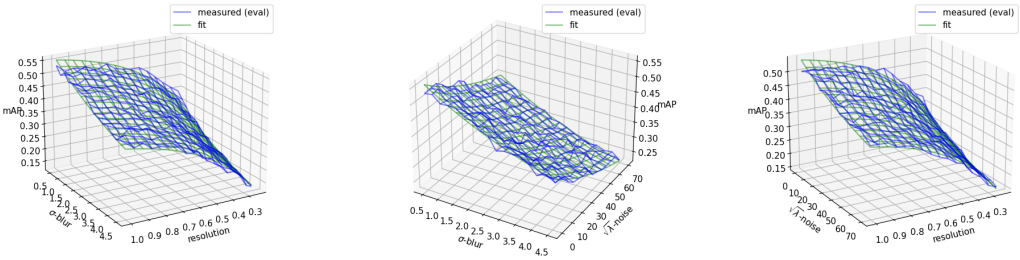
The fitting plots shown here depict the qualitatively similar fits obtained by our four performance prediction models on the COCO dataset.



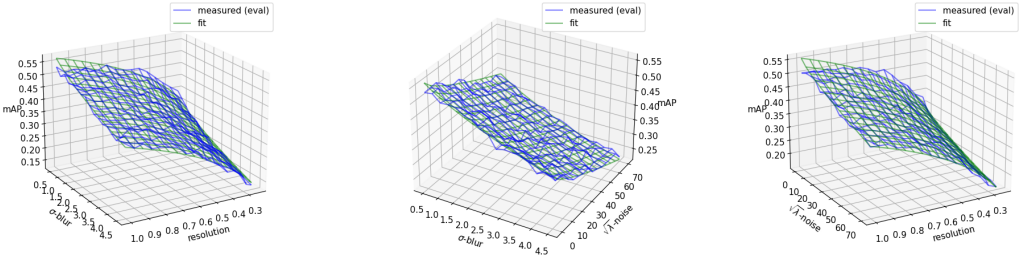

(a) GIQE-5 fits




(b) GIQE-3 fits

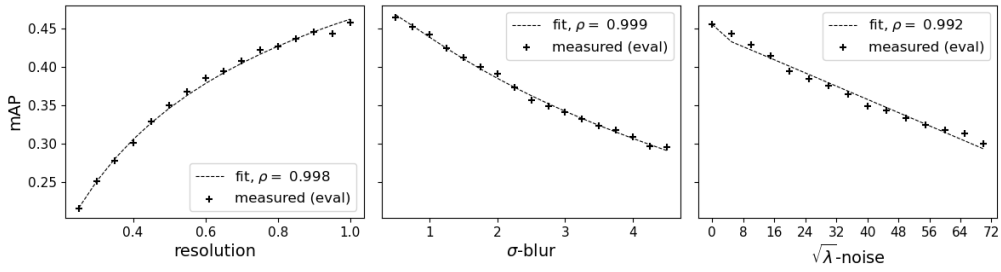



(c) Exponential fits

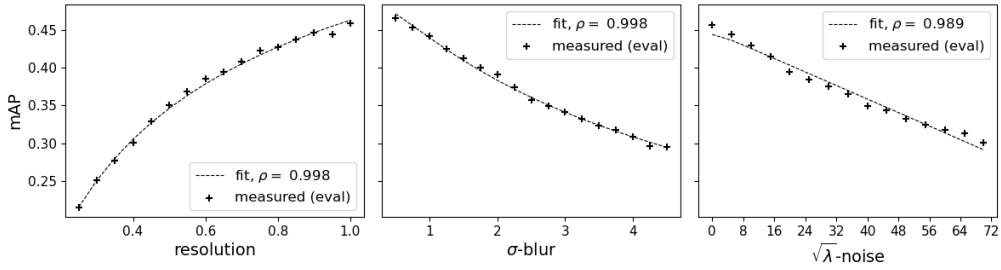



(d) Power law fits

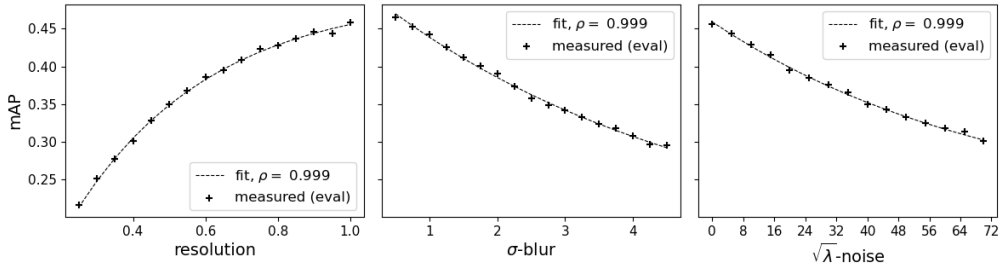
Figure 8.1: 2d views of measured and predicted performance for each of the four performance prediction models on the COCO dataset



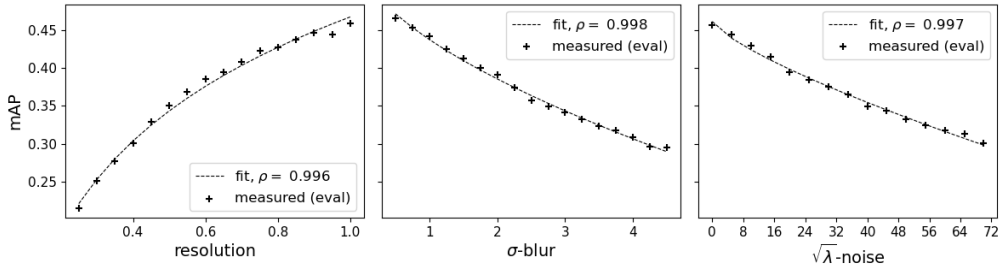

(a) GIQE-5




(b) GIQE-3




(c) Exponential

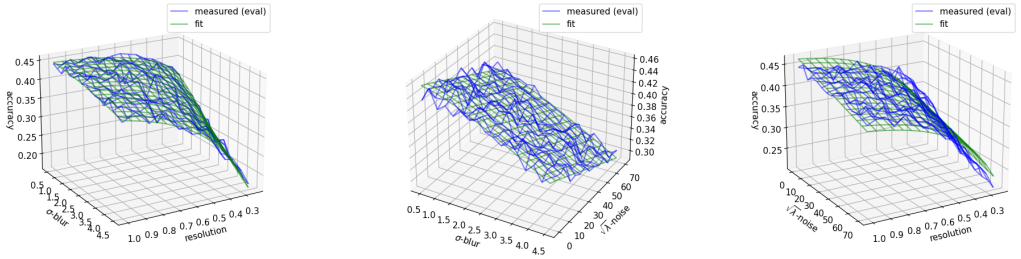



(d) Power law

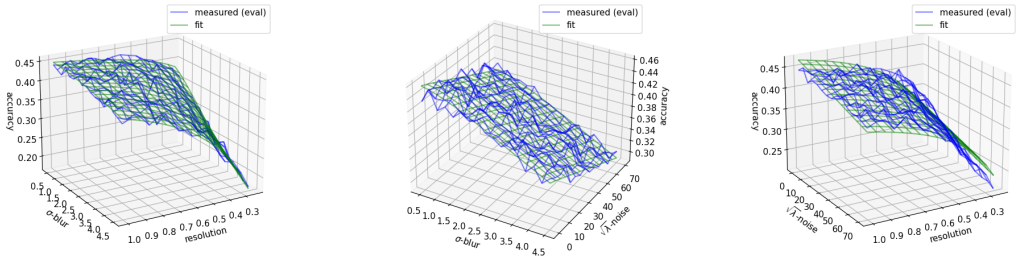
Figure 8.2: 1d views of measured and predicted performance for each of the four performance prediction models on the COCO dataset

### 8.1.2 Places365 RGB fit plots

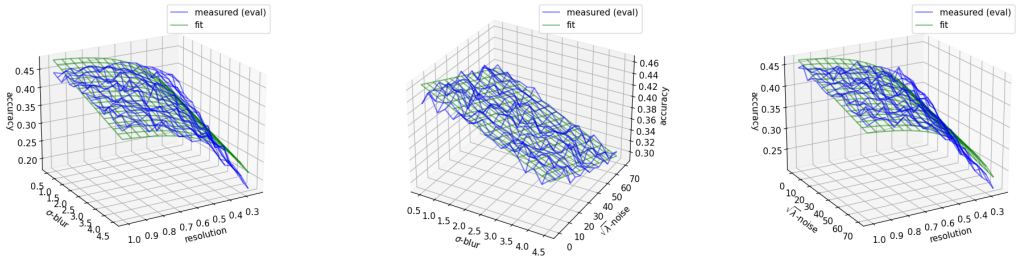
The fitting plots shown here depict fits obtained by our four performance prediction models on the Places365 datasets with the COCO test distortions applied.



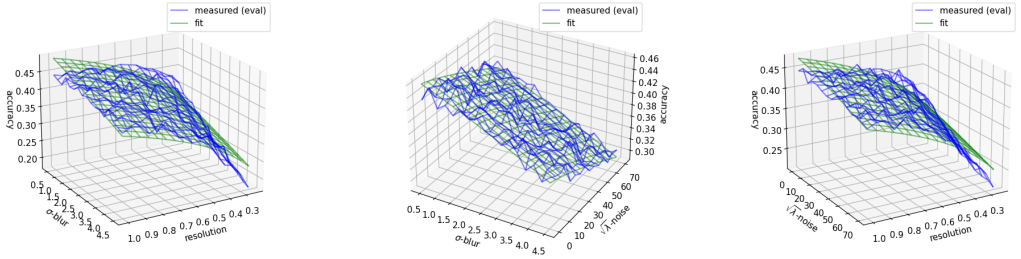

(a) GIQE-5 fits




(b) GIQE-3 fits

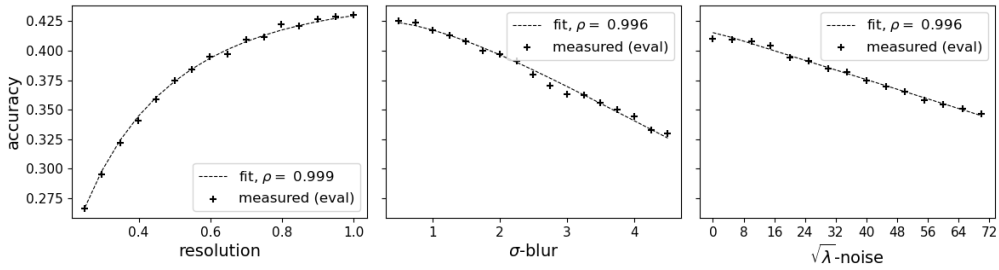



(c) Exponential fits

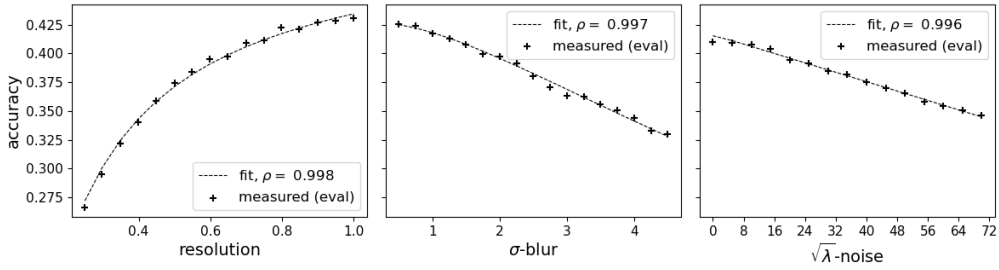



(d) Power law fits

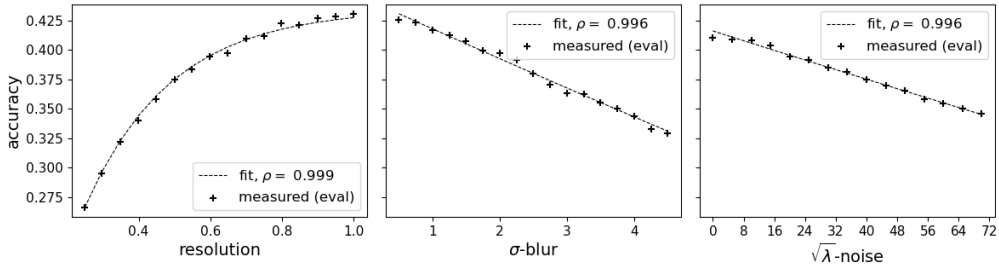
Figure 8.3: 2d views of measured and predicted performance for each of the four performance prediction models on the Places365 RGB dataset



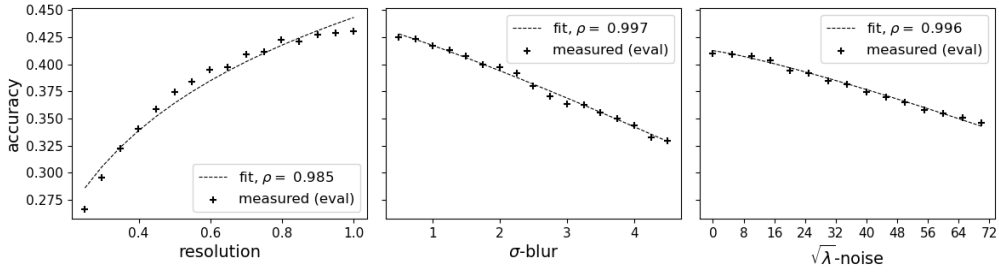

(a) GIQE-5




(b) GIQE-3




(c) Exponential




(d) Power law

Figure 8.4: 1d views of measured and predicted performance for each of the four performance prediction models on the Places365 RGB dataset

## 8.2 Code

With the exception of the code used to create a small number of figures, found primarily in Chapter 2, all of the code used in this research resides in two repositories on GitHub:

- Relative edge response code, mostly used in Chapter 3, can be found at `https://github.com/acb08/relative-edge-response` [98].
- Code used to train and test models, generate datasets, and analyze results can be found at `https://github.com/acb08/image-quality-for-deep-learning` [131].