Machine learning notes

Contents

Notation

Objects

Variables are generally denoted by italic serif letters. Most constants are denoted by upright serif letters. Random variables are underlined. Vectors and sequences are denoted by lowercase bold letters. Matrices and multidimensional-arrays are denoted by uppercase bold letter. Sets are denoted by uppercase blackboard-bold letters. Latin or Greek letters can be used for any type of object.

a, A, θ	Variable (commonly scalar or function)
$\boldsymbol{a}$, $\boldsymbol{\theta}$	Vector or sequence (commonly column vector)
$\boldsymbol{A}$, $\boldsymbol{\Theta}$	Matrix or multidimensional array
A	Set or multiset
a, A, $\mathrm{\theta}$	Constant
$\mathbf{a}$, $\mathbf{\theta}$	Vector or sequence constant
$\mathbf{A}$, $\mathbf{\Theta}$	Matrix or multidimensional array constant
$\mathbb{A}$	Set constant
$\underline{a}$, $\underline{A}$, $\underline{\theta}$	Random variable
$\underline{\boldsymbol{a}}$, $\underline{\boldsymbol{\theta}}$	Random vector or sequence
$\underline{\boldsymbol{A}}$, $\underline{\boldsymbol{\Theta}}$	Random matrix or multidimensional array
$\underline{\mathit{A}}$	Random set or multiset
a, riječ	Textual label not representing an object

Constants

$\{\}$	Enpty set
e	The number that satisfies $\frac{\mathrm{d}}{\mathrm{d}x}\mathrm{e}^x = \mathrm{e}^x$
$\mathbf{0}$	Null-vector
$\mathbf{e}_i$	i-th canonical basis vector
$\mathbf{1}$	The sum of all canonical basis vectors
$\mathbf{I}$, $\mathbf{I}_n$	Identity matrix (with n rows/columns)
$\mathbb{N}, \mathbb{Z}, \mathbb{R}, \mathbb{C}$	A standard set

$\mathbb{R}_{\geq 0}$, $\mathbb{R}_{>0}$	The set of all non-negative/positive real numbers

Defining sets and arrays

$a..b$	Shorthand notation for $a, .., b$
$\{a..b\}$	A subset of integers from a to b
$\{f(a)\colon P(a)\}$, $\{f(a)\}_{P(a)}$	A set with elements defined by a function f and a predicate P
$\{f(a)\}_a$	A set with elements defined by a function f and variables a from an implicitly defined set
$\{a_1..a_n\}$, $\{a_i\}_{i=1..n}$	A set with n elements
$[x_1, .., x_n]$	A row vector
$[a_i]_i$, $[a_{i,j}]_{i,j}$, $[a_{i,j,k}]_{i,j,k}$	A multidimensional array with an implicit or undefined number of elements
$[a, b)$	A semi-closed interval

Subscript and superscript

U donjem i gornjem indeksu oznake mogu biti oznake drugih matematičkih objekata ili slova ili riječi koje ne predstavljaju matematičke objekte. Redni brojevi elemenata vektora ili višedimenzionalnih nizova se, ako nije određeno drugačije, pišu u donjem indeksu oznake vektora u uglatim zagradama. Npr. i-ti element vektora $\boldsymbol{a} = [a_1, .., a_n]^\mathsf{T}$ je $\boldsymbol{a}_{[i]} = a_i$. Indeksi kod n-dimenzionalnih nizova mogu biti i vektori iz $\mathbb{N}^n$, ili kombinacije vektora manje dimenzije sa skalarima.

a^g_d	Varijabla s oznakama u donjem i gornjem indeksu
$\boldsymbol{a}_{[i]}$	i-ti element vektora $\boldsymbol{a}$
$\boldsymbol{a}_{[i_1:i_2]}$	Vektor kojeg čine elementi $\boldsymbol{a}_{[i_1]}, \boldsymbol{a}_{[i_1+1]}, .., \boldsymbol{a}_{[i_2]}$
$\boldsymbol{a}_{[(i_1..i_n)]}$	Vektor kojeg čine elementi $\boldsymbol{a}_{[i_1]}, \boldsymbol{a}_{[i_2]}, .., \boldsymbol{a}_{[i_n]}$
$\boldsymbol{A}_{[i,j]}$	Element i, j matrice $\boldsymbol{A}$
$\boldsymbol{A}_{[i,:]}$	i-ti redak matrice $\boldsymbol{A}$
$\boldsymbol{A}_{[:,i_1:i_2,j]}$	2-D odsječak 3-D niza $\boldsymbol{A}$
$\boldsymbol{A}_{[\boldsymbol{i}]}$	Element $\boldsymbol{A}_{[\boldsymbol{i}_{[1]},..,\boldsymbol{i}_{[n]}]}$ n-D niza

$\boldsymbol{A}_{[\boldsymbol{i}_1:\boldsymbol{i}_2]}$	Podniz $\boldsymbol{A}_{\left[\boldsymbol{i}_{1\,[1]}:\boldsymbol{i}_{2\,[1]},..,\boldsymbol{i}_{1\,[n]}:\boldsymbol{i}_{2\,[n]}\right]}$ n-D niza
$\boldsymbol{A}_{[\boldsymbol{i}_1:\boldsymbol{i}_2,:]}$	Podniz $\boldsymbol{A}_{\left[\boldsymbol{i}_{1\,[1]}:\boldsymbol{i}_{2\,[1]},..,\boldsymbol{i}_{1\,[n-1]}:\boldsymbol{i}_{2\,[n-1]},:\right]}$ n-D niza

Operacije linearne algebre i operacije s nizovima

$\langle \boldsymbol{a} \vert \boldsymbol{b} \rangle$, $\boldsymbol{a}^\mathsf{T}\boldsymbol{b}$	Skalarni produkt
$\boldsymbol{a}\boldsymbol{b}^\mathsf{T}$	Vanjski produkt
$\boldsymbol{a} \odot \boldsymbol{b}$	Umnožak po elementima; Hadamardov produkt
$\boldsymbol{a} \oslash \boldsymbol{b}$	Dijeljenje po elementima
$\boldsymbol{a}^{\odot b}$	Potenciranje po elementima
$\boldsymbol{A}\boldsymbol{B}$	Matrično množenje
$\boldsymbol{A}^{-1}$	Inverz matrice
$\boldsymbol{A}^\mathsf{T}$	Transponiranje
$\mathrm{diag}(\boldsymbol{a})$	Dijagonalna matrica kojoj dijagonalu čini vektor $\boldsymbol{a}$
$\det(\boldsymbol{A})$	Determinanta matrice $\boldsymbol{A}$
$\left\|\boldsymbol{a}\right\|_2$	L^2-norma vektora $\boldsymbol{a}$
$\left\|\boldsymbol{a}\right\|_p$	L^p-norma vektora $\boldsymbol{a}$
$\left\|\boldsymbol{A}\right\|_p$	Matrična L^p-norma matrice $\boldsymbol{A}$
$\left\|\boldsymbol{A}\right\|_\mathsf{F}$	Frobeniusova norma matrice $\boldsymbol{A}$
$\boldsymbol{a} +\!\!+ \boldsymbol{b}$	Konkatenacija vektora (stupaca) $\boldsymbol{a} \in \mathbb{R}^n$ i $\boldsymbol{b} \in \mathbb{R}^m$ u vektor iz $\mathbb{R}^{n+m}$
$\boldsymbol{A} +\!\!+ \boldsymbol{B}$	Konkatenacija nizova po prvoj dimenziji
$\mathrm{vec}(\boldsymbol{A})$	Funkcija koja preslikava niz iz $\mathbb{R}^{d_1 \times \cdots \times d_n}$ u $\mathbb{R}^{d_1 \ldots d_n}$
$\dim(\boldsymbol{a})$	Dimenzija vektora
$\dim(\boldsymbol{A})$	Vektor dimenzija niza; $[d_1, .., d_n]$ za $\boldsymbol{A} \in \mathbb{R}^{d_1 \times \cdots \times d_n}$

Diferencijalni račun

$\frac{\mathrm{d}y}{\mathrm{d}x}$, $\frac{\mathrm{d}}{\mathrm{d}x}f(x)$	Derivacija $y = f(x)$ po x

$\frac{\partial y}{\partial x}$, $\frac{\partial}{\partial x} f(x)$	Parcijalna derivacija $y = f(x)$ po x
$\nabla_{\boldsymbol{x}} y$, $\nabla_{\boldsymbol{x}} f(x)$, $\left(\frac{\partial y}{\partial \boldsymbol{x}}\right)^{\mathsf{T}}$	Gradijent $y = f(\boldsymbol{x})$ po $\boldsymbol{x}$
$\nabla_{\boldsymbol{X}} y$, $\nabla_{\boldsymbol{X}} f(x)$	Gradijent $y = f(\boldsymbol{x})$ po $\boldsymbol{X}$
$\frac{\partial^2 y}{\partial \boldsymbol{x} \partial \boldsymbol{x}^{\mathsf{T}}}$, $\boldsymbol{H}_f(\boldsymbol{x})$, $\boldsymbol{H}$	Hessijan iz $\mathbb{R}^{n \times n}$ za $f \colon \mathbb{R}^n \to \mathbb{R}$ i $y = f(\boldsymbol{x})$
$\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}}$, $\boldsymbol{J}_f(\boldsymbol{x})$, $\boldsymbol{J}$	Jakobijeva matrica iz $\mathbb{R}^{m \times n}$ za $f \colon \mathbb{R}^n \to \mathbb{R}^m$ i $\boldsymbol{y} = f(\boldsymbol{x})$
$\int_{\mathbb{A}} f(x) \,\mathrm{d}x$, $\int_{x \in \mathbb{A}} f(x)$	Određeni integral funkcije $f(x)$ po $x \in \mathbb{A}$
$\int f(x) \,\mathrm{d}x$, $\int_x f(x)$	Određeni integral funkcije $f(x)$ po $x \in \mathbb{A}$, gdje je $\mathbb{A}$ implicitan

Teorija vjerojatnosti

Svakoj slučajnoj varijabli $\underline{a}$ jednoznačno je dodijeljena jedna razdioba $\mathrm{p}(\underline{a})$ (ili $\mathrm{P}(\underline{a})$) i funkcija gustoće vjerojatnosti (koja može biti poopćena funkcija) $p_{\underline{a}}(a) = \mathrm{p}(\underline{a} = a)$. $\mathrm{P}(A)$ označava vjerojatnost događaja A, a $P_{\underline{a}}$ funkciju vjerojatnosti slučajne varijable $\underline{a}$. Mogući su i kraći zapisi $\mathrm{p}(a)$ i $\mathrm{P}(a)$, gdje se po slovu koje označava vrijednost pretpostavlja slučajna varijabla označena istim slovom bez serifa. Mogu se koristiti i druge oznake za funkciju vjerojatnosti ili funkciju gustoće vjerojatnosti.

$(\underline{a} \mid \underline{b} = b)$, $(\underline{a} \mid b)$	Uvjetna slučajna varijabla
$(\underline{a}, \underline{b})$	Združena slučajna varijabla
$\underline{a} \perp \underline{b}$	*Slučajne varijable $\underline{a}$ i $\underline{b}$ su nezavisne*
$\underline{a} \not\perp \underline{b}$	*Slučajne varijable $\underline{a}$ i $\underline{b}$ su zavisne*
$\underline{a} \perp \underline{b} \mid \underline{c}$	*Slučajne varijable $\underline{a}$ i $\underline{b}$ su uvjetno nezavisne uz poznat ishod slučajne varijable $\underline{c}$*
$\underline{a} \not\perp \underline{b} \mid \underline{c}$	*Slučajne varijable $\underline{a}$ i $\underline{b}$ su uvjetno zavisne uz poznat ishod slučajne varijable $\underline{c}$*
p, q	Razdioba ili funkcija gustoće vjerojatnosti
$\mathbb{A}$	Događaj
$\{R(\underline{a})\}$	Događaj definiran predikatorm slučajne varijable $\underline{a}$

$\mathrm{P}(\{R(\underline{a})\})$, $\mathrm{P}(R(\underline{a}))$	Vjerojatnost događaja $\{R(\underline{a})\}$
$\mathrm{P}(\underline{a})$, $\mathrm{p}(\underline{a})$, $\mathcal{D}$	Razdioba slučajne varijable $\underline{a}$; P ako je $\underline{a}$ diskretna slučajna varijabla, a p ako nije ili ako se ne zna
$\mathrm{P}(\underline{a} = a)$, $P_{\underline{a}}(a)$, $\mathrm{P}(a)$	Vjerojatnost događaja $\{\underline{a} = a\}$
$\mathrm{p}(\underline{a} = a)$, $p_{\underline{a}}(a)$, $\mathrm{p}(a)$	Gustoća vjerojatnosti događaja $\{\underline{a} = a\}$
$p_{\underline{a}\mid b}(a)$, $\mathrm{p}(a \mid b)$	Gustoća vjerojatnosti događaja $\{\underline{a} = a \mid \underline{b} = b\}$
$p_{\underline{a},\underline{b}}(a, b)$, $\mathrm{p}(a, b)$	Gustoća vjerojatnosti događaja $\{\underline{a} = a, \underline{b} = b\}$
$\underline{a} \sim q$, $\mathrm{p}(\underline{a}) = q$	*Slučajna varijabla $\underline{a}$ ima razdiobu q*
$\underline{a} \sim \mathit{A}$	*Slučajna varijabla $\underline{a}$ ima takvu razdiobu da svi elementi (multi)skupa A imaju vjerojatnost proporcionalnu višestrukosti ($\frac{1}{\lvert\mathit{A}\rvert}$ za običan skup)*
$a \sim q$	*a se izvlači iz razidiobe q*
$a \sim \underline{a}$, $a \sim \mathrm{p}(\underline{a})$	*a se izvlači iz razidobe $\mathrm{p}(\underline{a})$*
$\mathop{\mathbf{E}}_{a\sim\underline{a}} f(a)$, $\mathop{\mathbf{E}}_{\underline{a}} f(a)$	Očekivanje funkcije slučajne varijable $\underline{a}$
$\mathop{\mathbf{D}}_{a\sim\underline{a}} f(a)$, $\mathop{\mathbf{D}}_{\underline{a}} f(a)$	Disperzija (varijanca) funkcije slučajne varijable $\underline{a}$
$\mathrm{Cov}(\underline{a}, \underline{b})$	Kovarijanca
$\mathcal{N}(\mu, \sigma^2)$	Normalna razdioba s učekivanjem μ i varijancom σ^2
$\mathcal{U}(\mathit{A})$	Uniformna razdioba nad skupom A

Information theory

$\mathrm{I}(\mathit{A})$	Information content of event A
$\mathrm{H}(\underline{a})$	Entropy
$\mathrm{h}(\underline{a})$	Differnetial entropy
$\mathrm{I}(\underline{a}; \underline{b})$	Mutual information
$\mathrm{H}(\underline{a} \mid \underline{b})$	Conditional entropy
$\mathrm{H}_{\underline{b}}(\underline{a})$	Cross entropy
$\mathrm{D}_{\underline{b}}(\underline{a})$	Relative entropy (Kullback-Leibler divergence)

Grafovi

$\mathrm{pa}_G(a)$	Skup čvorova roditelja čvora a u grafu G
$\mathrm{ch}_G(a)$	Skup čvorova djece čvora a u grafu G
$\mathrm{pred}_G(a)$	Skup čvorova prethodnika čvora a u grafu G
$\mathrm{succ}_G(a)$	Skup čvorova nasljednika čvora a u grafu G

Other

$\mathbb{A} \to \mathbb{B}$	A set of functions with domain $\mathbb{A}$ and codomain $\mathbb{B}$
$f \in (\mathbb{A} \to \mathbb{B})$	Function definition; a function mapping from its domain $\mathbb{A}$ to its codomain $\mathbb{B}$
$x \mapsto f(x)$	Function definition; a function mapping x from its domain to $f(x)$ in its codomain
$f + g$	Sum of functions
fg	Product of functions
$f * g$	Convolution of functions
$\langle f \vert g \rangle$	Scalar product of functions
$\vert\mathbb{A}\vert$	Set cardinality
$\delta(\cdot)$	Dirac delta
$[\![\cdot]\!]$	Iverson bracket; $[\![P]\!] = \begin{cases} 1, & P \equiv \top \\ 0, & P \equiv \bot \end{cases}$
$f[\mathbb{A}]$	Image of $\mathbb{A}$ (subset of the domain) under f; $f[\mathbb{A}] \coloneqq \{f(a) \mid a \in \mathbb{A}\}$
$f^{-1}[\mathbb{B}]$	Inverse image (preimage) of $\mathbb{B}$ (subset of the codomain) under f; $f^{-1}[\mathbb{B}] \coloneqq \{a \mid f(a) \in \mathbb{B}\}$

Chapter 1

Introduction

1.1 Probability

Probability of an event $\mathrm{P}(E)$

The distribution of a random variable $\underline{x}$ is denoted $\mathrm{p}_{\underline{x}}$. If $\underline{x}$ is known to be discrete, its distribution can be denoted with $\mathrm{P}_{\underline{x}}$.

$$\mathrm{P}_{\underline{x}}(x) = \mathrm{P}(\underline{x} = x). \tag{1.1}$$

$$\mathrm{p}_{\underline{x}}(x) = \lim_{\epsilon \to 0} \frac{\mathrm{P}(\underline{x} \in B_\epsilon(\boldsymbol{x}))}{\int_{x' \in B_\epsilon(\boldsymbol{x})} \mathrm{d}x'}. \tag{1.2}$$

$$\mathrm{P}_{\underline{x}}(x \in A) = \int_{x \in A} \mathrm{dp}_{\underline{x}}(x). \tag{1.3}$$

1.1.1 Functions of random variables

For any function $f \in (\mathbb{A} \to \mathbb{B})$, we will use the same symbol to denote an equivalent function that maps random variables taking values in $\mathbb{A}$ to random variables taking values in $\mathbb{B}$:

$$\mathrm{P}(f(\underline{a}) \in \mathbb{B}_1) \coloneqq \mathrm{P}\big(\underline{a} \in f^{-1}[\mathbb{B}_1]\big), \tag{1.4}$$

where $f^{-1}[\mathbb{B}_1] \coloneqq \{a \mid f(a) \in \mathbb{B}_1\}$ is what we call the preimage of $\mathbb{B}_1 \subseteq \mathbb{B}$ under f.

Conditional expectation:

$$\mathbf{E}(\underline{x} \mid \underline{y}) = (y \mapsto \mathbf{E}(\underline{x} \mid y))(\underline{y}). \tag{1.5}$$

The conditional expectation $E(\underline{x} \mid \underline{y})$ is a function of the random variable $\underline{y}$ and, thus, is a random variable as well.

1.2 Statistics

1.2.1 Monte Carlo integration

Monte Carlo integration (approximation) is a method of approximating integrals that can be expressed as expectations of some random variables.

Let $u \in A \to \mathbb{R}$ be a function. Let $I \coloneqq \int u(x)\,\mathrm{d}x$ be an integral that is hard to compute. If we express u as the product of a function f and a probability density function (distribution) p, $u(x) = f(x)p(x)$, the integral I can be expressed as the expectation of $f(\underline{x})$, where $\underline{x}$ is distributed according to p:

$$I = \int f(x)\,\mathrm{d}x = \int f(x)p(x)\,\mathrm{d}x = \mathop{\mathbf{E}}_{x\sim p}(f(x)). \tag{1.6}$$

This expectation can be approximated with the following estimator:

$$\hat{\underline{I}}_n \coloneqq \frac{1}{n}\sum_{i=1..n} f(\underline{x}_i), \tag{1.7}$$

where $\underline{x}_i \sim p$. The estimator $\hat{\underline{I}}_n$ is unbiased if $\underline{x}_i$ are independent and it is valid if variances of $u(\underline{x}_i)$ are bounded.

1.2.2 Rejection sampling

1.2.3 Importance sampling

$I \coloneqq \int_{x\in B} f(x)\,\mathrm{d}x$

1.3 Information theory

1.3.1 Information-theoretic measures

functionals

Basic concepts

Information content of an event – optimal message length for the event $\{\underline{x} = x\}$:

$$\mathrm{I}(\underline{x} = x) = -\ln \mathrm{P}(x). \tag{1.8}$$

Entropy (Shannon entropy) of a random variable (or a distribution) – expected message length for optimally encoded elementary events of $\underline{x}$:

$$\mathrm{H}(\underline{x}) = \mathop{\mathbf{E}}_{\underline{x}} \mathrm{I}(\underline{x} = x) = -\mathbf{E} \ln \mathrm{P}(\underline{x}). \tag{1.9}$$

The same formula applies for joint distributions, e.g. for the distribution $\mathrm{P}(\underline{x}, \underline{y})$, we denote entropy (also called **joint entropy**) by $\mathrm{H}(\underline{x}, \underline{y})$. Entropy is a common measure of uncertainty.

Cross entropy – expected message length if the optimal code for $\mathrm{P}_{\underline{y}}$ is used, but $\mathrm{P}_{\underline{x}}$ is sampled.

$$\mathrm{H}_{\underline{y}}(\underline{x}) = \mathop{\mathbf{E}}_{\underline{x}} \mathrm{I}(\underline{y} = x) = -\mathop{\mathbf{E}}_{\underline{x}} \ln \mathrm{P}(\underline{y} = x). \tag{1.10}$$

Relative entropy (Kullback–Leibler (KL) divergence) – difference of cross entropy and entropy, measures how much $\mathrm{P}_{\underline{y}}$ differs from $\mathrm{P}_{\underline{x}}$:

$$\mathrm{D}_{\underline{y}}(\underline{x}) = \mathrm{H}_{\underline{y}}(\underline{x}) - \mathrm{H}(\underline{x}) = \mathop{\mathbf{E}}_{\underline{x}}(\mathrm{I}(\underline{y} = x) - \mathrm{I}(\underline{x} = x)) = \mathop{\mathbf{E}}_{\underline{x}} \ln \frac{\mathrm{P}(x)}{\mathrm{P}(\underline{y} = x)}. \tag{1.11}$$

Mutual information of random variables is the expectation of how much knowing the outcome of one of them gives information (or reduces the uncertainty) about the other:

$$\mathrm{I}(\underline{x}; \underline{y}) = \mathrm{H}(\underline{x}) - \mathrm{H}(\underline{x} \mid \underline{y}) = \mathrm{H}(\underline{y}) - \mathrm{H}(\underline{y} \mid \underline{x}) = \mathrm{H}(\underline{x}) + \mathrm{H}(\underline{y}) - \mathrm{H}(\underline{x}, \underline{y}). \tag{1.12}$$

If there is a common condition, we can use $\mathrm{I}(\underline{x}; \underline{y} \mid z)$ as a shorter notation for $\mathrm{I}((\underline{x} \mid z); (\underline{y} \mid z))$. If we want to express mutual information between e.g. $\underline{x}$ and $\underline{y} \mid z$, we do it like this: $\mathrm{I}(\underline{x}; (\underline{y} \mid z))$, without ambiguity.

Mutual information can also be expressed as relative entropy:

$$\mathrm{I}(\underline{x}; \underline{y}) = \mathrm{D}_{\mathrm{P}_{\underline{x}}\,\mathrm{P}_{\underline{y}}}(\mathrm{P}_{\underline{x},\underline{y}}) \tag{1.13}$$

$$\mathrm{I}(\underline{x}; \underline{y}) = \mathrm{D}_{\mathrm{P}[\underline{x}]\,\mathrm{P}[\underline{y}]}(\mathrm{P}[\underline{x}, \underline{y}]) \tag{1.14}$$

$$\mathrm{I}(\underline{x}; \underline{y}) = \mathrm{D}_{\mathrm{P}(\underline{x})\,\mathrm{P}(\underline{y})}(\mathrm{P}(\underline{x}, \underline{y})) \tag{1.15}$$

$$\mathrm{I}(\underline{x}; \underline{y}) = \mathrm{D}_{[\underline{x}][\underline{y}]}([\underline{x}, \underline{y}]). \tag{1.16}$$

Conditional measures

Conditional counterparts of the information-theoretic measures have random variables in the condition-part of the expression that represents the argument of a measure. e.g. **Conditional entropy** is defined like this:

$$\mathrm{H}(\underline{x} \mid \underline{y}) = \mathop{\mathbf{E}}_{\underline{y}} \mathrm{H}(\underline{x} \mid y). \tag{1.17}$$

Similarly, conditional cross-entropy can be defined like this:

$$\mathrm{H}_{\underline{y}}(\underline{x} \mid \underline{z}) = \mathop{\mathbf{E}}_{\underline{z}} \mathrm{H}_{\underline{y}}(\underline{x} \mid z), \tag{1.18}$$

conditional mutual information like this:

$$\mathrm{I}(\underline{x}; \underline{y} \mid \underline{z}) = \mathop{\mathbf{E}}_{\underline{z}} \mathrm{I}(\underline{x}; \underline{y} \mid z). \tag{1.19}$$

Differential counterparts

Information theory and measure theory

https://en.wikipedia.org/wiki/Information_theory_and_measure_theory

1.3.2 Kolmogorov complexity

1.3.3 Minimum description length

1.4 Gradient-based optimization

Notation, where $f \in (\mathbb{A} \to \mathbb{B})$, $x \in \mathbb{A}$, and $y \coloneqq f(x)$ (equivalently $f = (x \to y)$):

$$\mathrm{D}_f = \frac{\partial f(x)}{\partial x} = f' \qquad \text{is a function,} \tag{1.20}$$

$$\mathrm{D}_f(x) = \frac{\partial f(\alpha)}{\partial \alpha}(x) = \left.\frac{\partial f(\alpha)}{\partial \alpha}\right|_x = \frac{\partial f(x)}{\partial x} = f'(x) \qquad \text{is a domain element,} \tag{1.21}$$

$$\mathrm{D}_{x \mapsto y} = \frac{\partial y}{\partial x} = \frac{\partial y}{\partial x}(x') \qquad \text{is a domain element (short),} \tag{1.22}$$

1.4.1 Dependence of gradient on parametrization

Let $\boldsymbol{x}$ be a vector in the input space of a differentiable function $f \in (\mathbb{R}^n \to \mathbb{R})$. Let $\boldsymbol{y} \coloneqq \frac{1}{a}\boldsymbol{x}$. Then $f(a\boldsymbol{y}) = f(\boldsymbol{x})$ and

$$\frac{\partial f(a\boldsymbol{y})}{\partial \boldsymbol{y}} = a\frac{\partial f(a\boldsymbol{y})}{\partial a\boldsymbol{y}} = a\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}. \tag{1.23}$$

Using equation (1.23), we can compare the ratio of the scale of the parameter and the scale of the gradient:

$$\frac{\left\|\frac{\partial f(a\boldsymbol{y})}{\partial \boldsymbol{y}}\right\|}{\|\boldsymbol{y}\|} = \frac{\left\|a\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}\right\|}{\left\|\frac{1}{a}\boldsymbol{x}\right\|} = a^2\frac{\left\|\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}\right\|}{\|\boldsymbol{x}\|}. \tag{1.24}$$

In order to make the gradient invariant to such a parametrization, it could be divided by the square of its p-norm (or multiplied by the square of the norm of the parameter). We can eaily show that the scaled gradient is independent of the parametrization:

$$\frac{\left\|\frac{\partial f(a\boldsymbol{y})}{\partial \boldsymbol{y}}\right\|^{-2}\frac{\partial f(a\boldsymbol{y})}{\partial \boldsymbol{y}}}{\|\boldsymbol{y}\|} = \frac{\left\|a\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}\right\|^{-2}a\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}}{\frac{1}{a}\|\boldsymbol{x}\|} = \frac{\left\|\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}\right\|^{-2}\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}}{\|\boldsymbol{x}\|}. \tag{1.25}$$

An issue that is introduced here is that the scaled gradient can't be $\mathbf{0}$. Moreover, it will grow as the unscaled gradient approaches $\mathbf{0}$, which is often undesirable. Alternatively, instead of dividing by the norm of its norm, the gradient could be multiplied by the square of the norm of the parameter:

$$\frac{\|\boldsymbol{y}\|^2\frac{\partial f(a\boldsymbol{y})}{\partial \boldsymbol{y}}}{\|\boldsymbol{y}\|} = \frac{\left\|\frac{1}{a}\boldsymbol{x}\right\|^2 a\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}}{\frac{1}{a}\|\boldsymbol{x}\|} = \frac{\|\boldsymbol{x}\|^2\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}}{\|\boldsymbol{x}\|}. \tag{1.26}$$

$$\tag{1.27}$$

The gradient update rule for $\boldsymbol{x}$ is:

$$\boldsymbol{x}' = \boldsymbol{x} - \eta\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}^{\mathsf{T}}. \tag{1.28}$$

The gradient update rule for $\boldsymbol{y}$ is:

$$\boldsymbol{y}' = \boldsymbol{y} - \eta \frac{\partial f(a\boldsymbol{y})}{\partial \boldsymbol{y}}^\mathsf{T} \tag{1.29}$$

$$= \frac{1}{a}\boldsymbol{x} - a\eta \frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}^\mathsf{T} \qquad \text{(by equation (1.23))} \tag{1.30}$$

$$= \frac{1}{a}\left(\boldsymbol{x} - a^2\eta \frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}^\mathsf{T}\right) \tag{1.31}$$

$$\neq \frac{1}{a}\left(\boldsymbol{x} - \eta \frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}^\mathsf{T}\right) = \frac{1}{a}\boldsymbol{x}'. \tag{1.32}$$

We can see that, if the parameter $\boldsymbol{x}$ is scaled with $\frac{1}{a}$, the optimization step needs to be multiplied with $\frac{1}{a^2}$. This discrepancy arises due to the property of the Jacobian being covariant to the input space transformation. Let's modify the gradient update rule with !!!inverting!!! the Jacobian:

$$\boldsymbol{x}' = \boldsymbol{x} - \eta \frac{\partial \boldsymbol{x}}{\partial f(\boldsymbol{x})}^\mathsf{T}. \tag{1.33}$$

$$\boldsymbol{y}' = \boldsymbol{y} - \eta \frac{\partial \boldsymbol{y}}{\partial f(a\boldsymbol{y})}^\mathsf{T} \tag{1.34}$$

$$= \frac{1}{a}\boldsymbol{x} - \eta \frac{\partial \frac{1}{a}\boldsymbol{x}}{\partial f(\boldsymbol{x})}^\mathsf{T} \tag{1.35}$$

$$= \frac{1}{a}\boldsymbol{x} - \frac{1}{a}\eta \frac{\partial \boldsymbol{x}}{\partial f(\boldsymbol{x})}^\mathsf{T} \tag{1.36}$$

$$= \frac{1}{a}\left(\boldsymbol{x} - \eta \frac{\partial \boldsymbol{x}}{\partial f(\boldsymbol{x})}^\mathsf{T}\right) = \frac{1}{a}\boldsymbol{x}', \tag{1.37}$$

1.5 Machine learning

An Occam's razor idea. If the model (hypothesis search space) is simpler (smaller), we are more likely to find the correct hypothesis. If the correct hypothesis is complex, there will be more hypotheses consistent with the data and we are less likely to find the correct one anyway.

Chapter 2

Uncertainty

2.1 Uncertainty in machine learning

2.1.1 Expressing uncertainty

The basic and most complete way to express uncertainty are probability distributions. From a probability distribution, other uncertainty measures can be derived. Some common ones are the distribution of a derived random variable (a random variable which is a function of the original one), a parameter or a property of the distribution (e.g. the probability of the most certain value of the random variable or the entropy of the distribution).

2.1.2 Epistemic and aleatory uncertainty

Epistemička nesigurnost (nesigurnost modela) je nesigurnost u model ili parametre. Ona se može smanjiti uz više podataka/informacija. *Epistemička nesigurnost predikcije* dolazi od nesigurnosti u model/parametre.

Kad parametre modela procjenjujemo točkasto, nemamo aposteriornu razdiobu parametara i ne znamo kakva je epistemička nesigurnost (ne možemo ju izraziti), ali ju možemo smanjiti uz više podataka. Kod bayesovske procjene parametara ili kod ansabla možemo procijeniti epistemičku nesigurnost.

Aleatorna nesigurnost (predikcije) je nesigurnost koja dolazi od višeznačnosti podataka i ograničenja modela. Aleatorna nesigurnost se ne može smanjiti uz više podataka, ali bi se mogla smanjiti uz bolje podatke, tj. podatke koji imaju značajke koje sadrže više korisnih informacija, ili model koji pronalazi bolje značajke.

Kod diskriminativnog modela izlazna razdioba $p(y \mid x, \theta_{\mathsf{MAP}})$ izražava aleatornu nesigurnost.

Je li ukupna nesigurnost zbroj epistemičke i aleatorne? Mislim da procjena ukupne nesigurnosti ovisi o tome koliko je dobro procijenjena epistemička nesigurnost. Što je lošija procjena aposteriorne razdiobe parametara, to je procjena epistemičke nesigurnosti lošija.

2.1.3 Nesigurnost i izvanrazdiobni primjeri

Neka je D_{train} razdioba iz koje su došli primjeri za učenje. Diskrimanativni model uči funkciju $p(y \mid x)$. Ako je gustoća vjerojatnosti $D_{\text{train}}(x)$ jako mala (ili 0), moguće je da nije bilo sličnih primjera u skupu za učenje i model može dati bilo kakvu predikciju za taj primjer. Takve primjeri su *izvanrazdiobni primjeri*.

Ipak, pokazano je se da se (kod nekih modela) izvanrazdiobni primjeri često mogu dosta dobro prepoznavati na temelju izlazne razdiobe modela s točkasto procijenjenim parametrima.

2.1.4 Successfulness of epistemic uncertainty estimation with bayesian inference approximation

Chapter 3

Generalization and robustness

3.1 Introduction

Although common machine learning algorithms achieve human-level performance in many tasks, when given out-of-distribution, domain-shifted, corrupted or slightly modified examples from the training data distribution, they can often make overconfident and incorrect predictions [Hendrycks and Gimpel, 2016, Ganin and Lempitsky, 2015, Nguyen et al., 2015, Hendrycks and Dietterich, 2019, Engstrom et al., 2017, Szegedy et al., 2013]. Perhaps most surprisingly, by perturbing input examples (e.g. images) even inperceptibly for humans, common state-of-the-art algorithms can be made to significantly change their predictions both for examples from and outside of the training data distribution [Szegedy et al., 2013, Goodfellow et al., 2014b]. Such perturbed examples are called *adversarial examples*. The existence of adversarial examples indicates that common algorithms are probably performing well for somewhat wrong reasons, without actually *understanding data*.

Although adversarial examples exist on other tasks, image classification algorithms with deep, mostly convolutional, models is the problem considered in most research on adversarial examples. In the following text, the problem of image classification will be considered mostly as well.

Some evidence suggests that there is a trade-off between robustness and generalization [Tsipras et al., 2018, Madry et al., 2017, Su et al., 2018] with current algorithms, which is counter-intuitive because a hypothesis which optimally generalizes would have no adversarial examples. The question remains whether it is feasible or tractable with respect to computation or amount of data to implement such algorithms. A small but interesting step in this direction has been made by [Stutz et al., 2018].

A brief overview over a subset of the research on adversarial examples will be

given with the goals of improving intuition and understanding of adversarial examples, understanding how robustness and generalization of deep models relate, and getting closer to knowing whether it is possible to improve both robustness and generalization on real-world datasets.

3.2 Definitions and notation

To prevent some ambiguities, we define some basic machine learning concepts as they will be used here:

- Hypothesis – a mapping from inputs to predictions. A hypothesis will be denoted $h(\boldsymbol{x})$ or $h(\boldsymbol{x}; \boldsymbol{\theta})$, where $\boldsymbol{x} \in \mathbb{X}$ is an input vector and $\boldsymbol{\theta}$ a fixed vector of model parameters. For discriminative models $h(\boldsymbol{x}) = \mathrm{p}(\underline{y} \mid \boldsymbol{x})$, where $\underline{y} \mid \boldsymbol{x}$ is a conditioned random variable, short for $\underline{y} \mid \underline{\boldsymbol{x}} = \boldsymbol{x}$. Random variables are underlined.
- Model – a set of hypotheses $\mathbb{H}$ or a function h defined up to some free parameters $\boldsymbol{\theta}$: $\mathbb{H} = \{\boldsymbol{x} \mapsto h(\boldsymbol{x}; \boldsymbol{\theta})\}_{\boldsymbol{\theta}}$.
- Machine learning algorithm (short: algorithm) – usually a model together with inductive bias and an objective function.
- Classifier – a hypothesis that outputs a categorical distribution.
- The true hypothesis – the best hypothesis given knowledge of the true underlying distribution. Usually, it can be known only for synthetic data.
- Risk – the expected error on the data distribution. For discriminative models it can often be expressed as

$$R(h, \mathcal{D}) \coloneqq \mathop{\mathbf{E}}_{(\boldsymbol{x}, y) \sim \mathcal{D}} L(y, h(\boldsymbol{x})), \tag{3.1}$$

  where L is the loss function, which is usually proportional to negative log-likelihood $\mathrm{p}(y \mid \boldsymbol{x}, \boldsymbol{\theta})$.
- Empirical risk – the error on the empirical distribution $p_{\mathbb{D}}$ (a uniform distribution over the set of observed examples $\mathbb{D}$): $R_{\mathsf{E}}(h, \mathbb{D}) \coloneqq R(h, p_{\mathbb{D}})$. The objective of a machine learning algorithm is to minimize a combination of empirical risk and structural risk, which usually represents the prior hypothesis probability of the hypothesis (parameters) $\mathrm{p}(\boldsymbol{\theta})$.

Here are some common terms related to adversarial examples defined:

- Attack – an algorithm that generates adversarial examples.
- Defense – an idea used to make an algorithm resistant to adverarial examples.
- Natural example – an example from the distribution that the training data was sampled from.

- Adversarial perturbation – the difference between an adversarial example and the natural example it is based on. An adversarial example is based on a natural example iff the it is found in the neighbourhood of the natural example.
- Targeted attack – an attack with the goal that the hypothesis outputs some desired prediction.
- Non-targeted attack – an attack with the goal that the hypothesis outputs any misprediction.
- Threat model – a set of constraints on attacks. A defense should assume a threat model under which it is meant to work.
- Adversarial robustness (short: robustness) – either the performance of a hypothesis or algorithm under some threat model or how large a perturbations must be in order to turn natural examples into adversarial examples.
- Adversarial training – training that uses adversarial examples generated with an attack besides natural examples.
- Standard training – training that uses original examples from the training set without an attack.

Because an adversarial example can be defined in more ways, depending on the usefulness of the definition, and the definition is important for making precise conclusions, the definitions of an adversarial example will be discussed in section 3.3.

3.3 Adversarial example definitions

A broadly accepted definition of an adversarial example is that it is *an input designed to fool a hypothesis into producing a misprediction*. To make it more precise and useful, the "designed" part can be replaced with the constraint that adversarial examples are close to examples with correct predictions, which serves as a distinction between adversarial examples and examples far from decision boundaries with mispredicted labels causing standard generalization error. We get the following definition.

Definition 1 (adversarial example) *An adversarial example is an input for which the following holds:*

1. *It is close to an input with a correct prediction.*
2. *The hypothesis produces a misprediction.*

By this definition, the set of adversarial examples is a function of the hypothesis, the true hypothesis, and a neighbourhood function. A misprediction can be defined via some distance between distributions and a threshold. In classification,

a misprediction is usually defined as misclassification, i.e. the class with the highest predicted probability being different from true class.

As will be explained, a more practical, but less consistent definition is the following.

Definition 2 (practical adversarial example) *A (practical) adversarial example is an input for which the following holds:*

1. *It is close to an input with a correct prediction.*
2. *The hypothesis produces a different prediction than for the input it is based on (the input from point 1).*

Again, the difference between predictions can be defined via a distance between distributions. In classification, predictions of a hypothesis being different usually means that the class with the highest probability differs.

The difference from the first definition is that in the second definition knowledge of the true label is substituted with the prediction for the input with the correct prediction. This is useful for finding adversarial examples because all constraints can be known, but it is possible for a pair of natural examples differing in the label to have overlapping neighbourhoods and thus to be inside each other's neighbourhoods and thus be adversarial examples of each other. By the second definition, Even the true hypothesis can have adversarial examples close to decision boundaries. The first definition is more consistent in this sense, but then, knowing whether something is an adversarial example requires knowing the true hypothesis (i.e. it requires having solved the problem we are trying to solve with machine learning). In the following text, we will refer to the first definition as *the consistent definition* and the second one as *the practical definition*.

There are also some different or broader definitions of adversarial examples. Some authors consider out-of-distribution examples producing high-confidence predictions [Gal and Smith, 2018] or, most broadly, any inputs that fool the hypothesis [Brown et al., 2018].

3.4 The manifold hypothesis

A useful assumption in many machine learning problems is that the distribution of high-dimensional data is approximately concentrated in low-dimensional manifolds [Goodfellow et al., 2016]. A manifold can be described as a set of points related through neighourhoods that locally resembles a euclidean space. In machine learning, the term tends to be used loosely, usually denoting a connected set of points that can be approximated well with a small number of degrees of freedom, each corresponding to a local direction of variation, embedded in a higher-dimensional space. [Goodfellow et al., 2016] provide the following evidence. Firstly, the fact that the chance a natural example will be sampled from a simple

distribution in a high-dimensional space is negligible indicates that the data distribution is highly concentrated. Secondly, we can imagine transformations that represent connections between similar examples and can be applied to traverse the manifold (e.g. for images).

One of the goals of generative models is often to model the data manifold with a lower-dimensional representation. This assumption provides useful intuition for generative models, dimensionality reduction, the understanding of adversarial examples, and some adversarial defenses.

3.5 Properties of adversarial examples

Here some properties, phenomena and hypotheses regarding adversarial examples are described, some of which are more or less supported by evidence. Besides the hypotheses described here, there is also a quite predictive and well suppoted by evidence – the *non-robust features* hypothesis [Ilyas et al., 2019, Tsipras et al., 2018] described in section 3.8, which might not be in contradiction with most of the hypotheses presented here. A broader overview can be found in e.g. Serban and Poll [2018].

Rareness and closeness to natural examples The existence of adversarial examples shows that near almost every input, there are misclassified inputs nearby. Evidence also shows that they are rare, i.e. they can not be easily found with random search in the neighbourhood of an natural example. This initially lead to the hypothesis that existence of adversarial examples is caused by high nonlinearity of deep models and that they are located in *low-probability pockets* in the data manifold [Szegedy et al., 2013].

Local linearity However, Goodfellow et al. [2014b] have found that it is usually enough to know the locally linear behaviour (gradient with respect to the input) to reliably generate adversarial examples along directions of increasing the linear approximation. This is visualized in figure 3.1. They show that the direction of the perturbation matters more than the position in input space, evidencing against the low-probability pockets hypothesis. They suggest the *linearity hypothesis* whereby high local linearity of models accounts for the existence of adversarial examples. Summing small perturbations in many elements of high-dimensional inputs can produce a large change in the prediction. Su et al. [2017] have shown that even modifying only 1 pixel can often be enough to cause a misclassification, indicating that models can be highly sensitive to some input elements. Tabacof and Valle [2016] have found more evidence that adversarial examples span many-dimensional subsets in the input space.

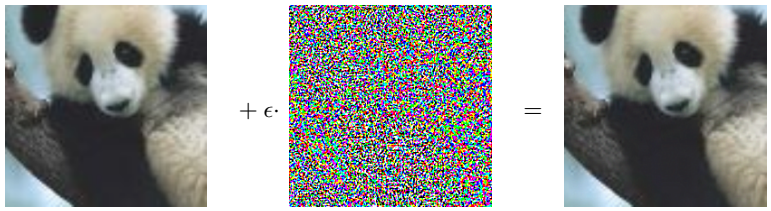


Figure 3.1: Generation of an adversarial example with FGSM, a single step attack. Italic words and numbers represent classes and confidences. The images are from Goodfellow et al. [2014b].

Transferability An interesting property is that adversarial examples generated for one model are often misclassified by other models and models trained on different datasets, i.e. they generalize across models and datasets [Szegedy et al., 2013]. This property of adversarial examples is called transferability. This suggests that most state-of-the art deep models are similarly biased. More analysis of transferebility can be found in Papernot et al. [2016], Liu et al. [2017], Tramèr et al. [2017].

Universal perturbations Moosavi-Dezfooli et al. [2016a] have observed that there exist perturbations that can reliably produce adversarial examples when added to almost any input and hypothesize that they exploit geometric correlations between different parts of decision boundaries.

Boundary tilting Tanay and Griffin [2016] hypothesize that adversarial examples exist when the decision boundary lies close to the submanifold of sampled data. They suggest that adversarial examples might be occurring along low-variance directions of the data where it is close to the manifold and that robustness could be improved with regularization. This is illustrated in figure 3.2.

High-dimensional space manifold geometry Gilmer et al. [2018] hypothesize that the existence of adversarial examples could be a naturally occurring result of the geometry of high-dimensional data manifolds. For a simple dataset with two classes consisting of examples from a pair of high-dimensional concentric spheres, they have observed that most random points in the data distribution are both correctly classified and close to a misclassified point. The authors have also given negative evidence on the hypothesis that adversarial examples are off the data manifold by showing that *on-manifold* adversarial examples can exist as well.

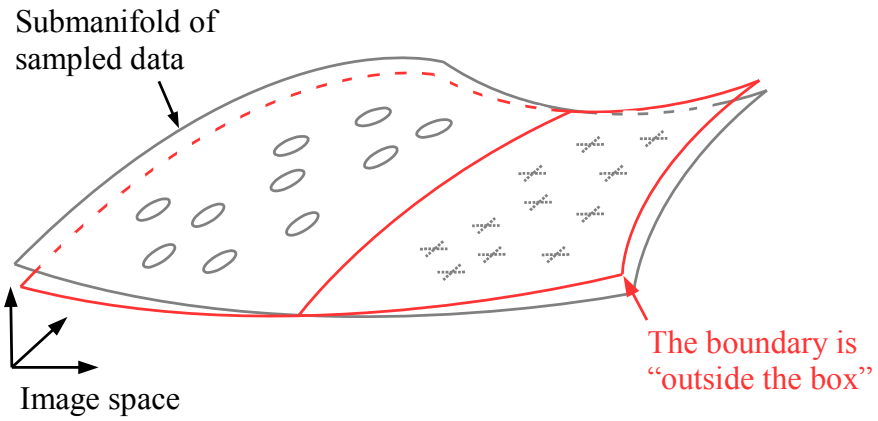


Figure 3.2: An illustration of boundary tilting from Tanay and Griffin [2016].

Adversarial examples of generative models Adversarial examples are not just a phenomenon related to discriminative models. They have been found to exist for some generative models as well [Goodfellow et al., 2014b, Kos et al., 2018]. An example is shown in figure 3.3.

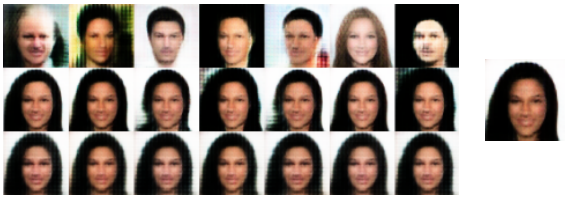


Figure 3.3: Reconstruction outputs for targeted attacks on a VAE-GAN model from Kos et al. [2018]. The rows represent of original image reconstructions (top), reconstructions of adversarial examples generated using an attack in latent space (middle) and a VAE-loss attack (bottom). The target reconstruction is on the right.

True ambiguity of adversarial examples of robust classifers Qualitative assesment of adversarial examples of robust classifiers suggest that they *understand data* much better. Their adversarial examples really are ambigous, i.e. generated perturbations don't look like noise but are semantically meaningful to humans [Tsipras et al., 2018, Li, 2018]. This is illustrated in figures 3.4 and 3.5. An interesting observation is that adversarially trained discriminative classifiers together with an iterative attack can interpolate between and generate quite

realistic examples. Tsipras et al. [2018] suggest a connection between the the saddle point problem of adversarial training and GAN training [Goodfellow et al., 2014a].

Figure 3.4: Original images and adversarial examples generated with a large perturbation using an iterative non-targeted attack on an adversarially trained Restricted ImageNet [Tsipras et al., 2018] classifier from Tsipras et al. [2018].

Further interesting phenomena and hypotheses related to the nature of adversarial examples, adversarial robustness and generalization will be discussed in section 3.8.

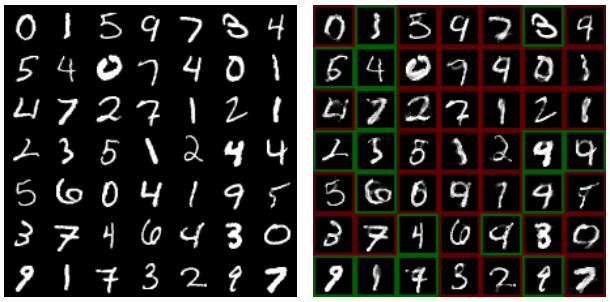


Figure 3.5: Clean images (left) and adversarial examples generated using an iterative non-targeted attack on a generative MNIST [LeCun et al., 2015] classifier with the factorization $\mathrm{p}(\boldsymbol{z})\,\mathrm{p}(\boldsymbol{y} \mid \boldsymbol{z})\,\mathrm{p}(\boldsymbol{x} \mid \boldsymbol{z}, \boldsymbol{y})$ (right) from Li [2018]. The adversarial examples marked in green are successful.

3.6 Finding adversarial examples

Let $\mathbb{X}$ be the input space, and $d \in (\mathbb{X} \times \mathbb{X} \to \mathbb{R}^+)$ a *distance function* for defininining similarity between inputs. For each example $\boldsymbol{x}$, we can also define its *neighbourhood* as $B_\epsilon(\boldsymbol{x}) = \{\boldsymbol{x}' \colon d(\boldsymbol{x}', \boldsymbol{x}) \leq \epsilon\}$, where ϵ is the maximum distance from the example.

Ideally, the neighbourhood of an example $\boldsymbol{x}$ should be the set of *perceptually similar* examples that all belong to the same class as $\boldsymbol{x}$ (their true class may be at most ambiguous), but it is hard to define such a neighbourhood (as it requires knowing the true model). A practical and common way of defining the neighbourhood function for images is to have d be a L^p distance where p is usually ∞ or 2. Note that, if an example is very near the true class boundary, such a neighbourhood may contain examples belonging to another class.

3.6.1 Attack objectives

Finding an adversarial example can be defined as a constrained optimization problem of maximizing some loss with respect to the input with the constraint that the input is in the neighbourhood $B_\epsilon(\boldsymbol{x})$:

$$\tilde{\boldsymbol{x}} = \underset{\boldsymbol{x}' \in B_\epsilon(\boldsymbol{x})}{\arg\max}\, L(y, h(\boldsymbol{x}')), \tag{3.2}$$

where y is the true label. Let $\hat{h}(\boldsymbol{x}) \coloneqq \arg\max_y h(\boldsymbol{x})_{[y]}$ denote the function that assigns the label with the highest probability to an input. An objective can also be to find the $\tilde{\boldsymbol{x}}$ closest to $\boldsymbol{x}$ such that the classifier misclassifies it [Moosavi-Dezfooli

et al., 2016b]:

$$\tilde{\boldsymbol{x}} = \underset{\boldsymbol{x}' \,:\, \boldsymbol{x}' \in B_\epsilon(\boldsymbol{x}) \wedge \hat{h}(\boldsymbol{x}') \neq y}{\arg\min} d(\boldsymbol{x}', \boldsymbol{x}). \tag{3.3}$$

The described objectives, where it only matters that the adversarial example is misclassified, are objectives for *non-targeted adversarial attacks*. There are also *targeted adversarial attacks*, where the objective is to create an adversarial example such that the model classifies it as some desired target. Targeted attack objectives corresponding to equations (3.2) and (3.3) are:

$$\tilde{\boldsymbol{x}} = \underset{\boldsymbol{x}' \in B_\epsilon(\boldsymbol{x})}{\arg\min} L(y_\mathsf{a}, h(\boldsymbol{x}')), \tag{3.4}$$

$$\tilde{\boldsymbol{x}} = \underset{\boldsymbol{x}' \,:\, \boldsymbol{x}' \in B_\epsilon(\boldsymbol{x}) \wedge \hat{h}(\boldsymbol{x}') = y_\mathsf{a}}{\arg\min} d(\boldsymbol{x}', \boldsymbol{x}), \tag{3.5}$$

where y_a denotes the adversarial target label. The difference in equation (3.4) is loss minimization and the adversarial target label instead of the true label. The difference in equation (3.5) is the condition that the predicted labels equals the adversarial target instead of differing from the true label.

non-targeted adversarial examples can also be generated without knowledge of the true label. Instead of the true label y, the predicted label $\hat{h}(\boldsymbol{x})$ can be used in equations (3.2) and (3.3). Such adversarial examples are called *virtual adversarial examples*. Miyato et al. [2017] propose the following attack objective for use in semi-supervised learning:

$$\tilde{\boldsymbol{x}} = \underset{\boldsymbol{x}' \in B_\epsilon(\boldsymbol{x})}{\arg\min} D((\underline{y} \mid \boldsymbol{x}, \boldsymbol{\theta}), (\underline{y} \mid \underline{\boldsymbol{x}} = \boldsymbol{x}', \boldsymbol{\theta})), \tag{3.6}$$

where D is some non-negative function that represents distance between distributions.

For adversarial training, non-targeted attacks should be preferred due to the *label-leaking* phenomenon [Kurakin et al., 2016b] where the learned classifier can overfit to adversarial examples and perform better on them than on natural examples, especially with attacks with a small number of iterations. For robustness evaluation with datasets that have many similar classes, non-targeted attacks can too easily fool the classifier and targeted attacks give more meaningful evaluation results [Athalye et al., 2018].

3.6.2 Common attacks

Being that finding adversarial examples is a constrained optimization problem, general gradient-based and black-box optimization algorithms can be used for attacks. Additionally, sometimes techniques specific to some potential defense and machine learning algorithm have to be used.

Some commonly known gradient-based attacks for finding adversarial examples are the following (using the notation from section 3.6):

- Box-constrained L-BFGS – Szegedy et al. [2013] propose to minimize $c\|\boldsymbol{x} - \tilde{\boldsymbol{x}}\|_2^2 + L(y, h(\tilde{\boldsymbol{x}}))$ with the constraint $\tilde{\boldsymbol{x}} \in [0, 1]$ with L-BFGS, a quasi-Newton optimization method. c is a number obtained via line-search that yields adversarial examples of minimum distance.
- Fast gradient sign method (FGSM) – an attack proposed by Goodfellow et al. [2014b] that requires a single gradient computation:

$$\tilde{\boldsymbol{x}} = \boldsymbol{x} + \epsilon \nabla_{\boldsymbol{x}} L(y, h(\boldsymbol{x})), \tag{3.7}$$

  where ϵ is the L^∞-norm of the perturbation. For L^2-constrined perturbations, Miyato et al. [2017] propose L^2 normalization instead of the sign function.
- DeepFool – an iterative non-targeted attack proposed by Moosavi-Dezfooli et al. [2016b] that in each step finds the optimal solution to a linear approximation of a loss in the L^2 ball $B_\epsilon(\boldsymbol{x})$ using the gradient in the current adversarial input. It is faster and finds smaller perturbations than L-BFGS and stronger than FGSM.
- Projected gradient descent (PGD) [Madry et al., 2017] basic iterative method (BIM) [Kurakin et al., 2016a] – an iterative gradient-based algorithm with random initialization [Madry et al., 2017] of the perturbation from within the $B_\epsilon(\boldsymbol{x})$ at the start and steps in the direction of the gradient sign:

$$\tilde{\boldsymbol{x}}_i = \Pi_{B_\epsilon(\boldsymbol{x})}\big(\tilde{\boldsymbol{x}}_{i-1} + \alpha \operatorname{sgn}\big(\nabla_{\tilde{\boldsymbol{x}}_{i-1}} L(y, h(\tilde{\boldsymbol{x}}_{i-1}))\big)\big). \tag{3.8}$$

  α is the step size, and $\Pi_{B_\epsilon(\boldsymbol{x})}$ is the projection into the L^p ϵ-ball around $\boldsymbol{x}$.
- Carlini-Wagner (CW) attacks – Carlini and Wagner [2017b] propose attacks with similar minimal perturbation objectives as Szegedy et al. [2013] and Moosavi-Dezfooli et al. [2016b]. They modify the loss function and, to enable unconstrained optimization, they introduce change of variables $\boldsymbol{\delta} = \frac{1}{2}(\tanh(\boldsymbol{w}) + \mathbf{1}) - \boldsymbol{x}$, which limits the perturbation $\boldsymbol{\delta}$ to the interval $[0, 1]$. With the PGD (BIM) attack, it is currently probably one of the 2 strongest attacks.

3.7 Improving adversarial robustness

There are different approaches (*defenses*) trying to improve adversarial robustness, most of which have been shown to actually be non-robust, but had appeared robust because they intentionally or unintentionally caused attacks that they were evaluated on not to be able to find adversarial examples [Carlini and Wagner, 2017a, Athalye et al., 2018, Uesato et al., 2018, Carlini and Wagner, 2017b]. Thus, it is important to put as much effort as needed to correctly evaluate

robustness, i.e. get an as low as possible upper bound on robustness. [Carlini et al., 2019] is a recent helpful overview on evaluation of robustness.

A broad overview of many defenses can be found in Serban and Poll [2018]. To give a few examples, some approaches use generative models to approximately project inputs to a learned data manifold (e.g. Samangouei et al. [2018]), some approaches are based on limiting the Lipschitz constant of the model to limit sensitivity to small input perturbations by regularization and model modification (e.g. Qian and Wegman [2018]), some research is looking into ways of guaranteeing robustness (e.g. Cohen et al. [2019]).

3.7.1 Adversarial training and empirical adversarial risk

The only defense currently believed to be effective according to Athalye et al. [2018] is adversarial training [Goodfellow et al., 2014b] with a strong attack [Madry et al., 2017], where the model is trained on adversarial examples as well as natural examples.

Madry et al. [2017] define what can be called *empirical adversarial risk* by allowing the worst-case attack to modify each the input in the empirical risk expression:

$$R_{\mathsf{EA}}(h, \mathbb{D}) \coloneqq \mathop{\mathbf{E}}_{(\boldsymbol{x},y)\sim p_{\mathbb{D}}} \left(\max_{\tilde{\boldsymbol{x}} \in B_\epsilon(\boldsymbol{x})} L(y, h(\tilde{\boldsymbol{x}})) \right). \tag{3.9}$$

They propose PGD for the attack during training and PGD with as large a number of iterations as necessary to approximate the worst-case adversary, i.e. get a better upper bound on robustness.

Still, adversarially trained models are not robust to attacks with weaker constraints than those used for training [Schott et al., 2018]. Furthermore, because adversarial examples are generated and robustness is evaluated according to the practical definition of an adversarial example and using L^p distance as a non-ideal approximation of perceptual similarity, performance is affected [Madry et al., 2017, Tsipras et al., 2018] and there can exist misclassified examples among which are *invariance-based* adversarial examples [Jacobsen et al., 2019].

3.8 Adversarial robustness and generalization

Based on the practical definition of an adversarial example or similar definitions, experimental [Madry et al., 2017, Su et al., 2018] and theoretical evidence [Tsipras et al., 2018] suggests that there is a trade-off between robustness and standard generalization for current models.

Here are some recent discoveries related to adversarial robustness and generalization presented. Robustness to distributional shift and corruptions also

seems to be quite relevant [Gilmer et al., 2019, Hendrycks and Dietterich, 2019], but it will not be discussed here.

3.8.1 A trade-off between robustness and generalization

Madry et al. [2017], Su et al. [2017], Tsipras et al. [2018] and others have empirically observed that adversarial robustness with current algorithms requires more capacity and negatively affects generalization. Su et al. [2017] have observed that older convolutional architectures with no shortcut connections, like AlexNet [Krizhevsky et al., 2012] and VGG [Simonyan and Zisserman, 2014] seem to be inherently more robust than architectures like ResNet [He et al., 2015], DenseNet [Huang et al., 2016] and MobileNet [Howard et al., 2017] and NASNets [Zoph et al., 2018] with standard training. Madry et al. [2017] has observed that with adversarial training, more model capacity is required and natural test set performance is reduced. Furthermore, Tsipras et al. [2018] have, based on the practical definition of an adversarial example, theoretically demonstrated an aspect of the trade-off. Another reason that affects performance suggested by them is that salient features might be harder to learn and that algorithms rely on highly predictive but *non-robust* features.

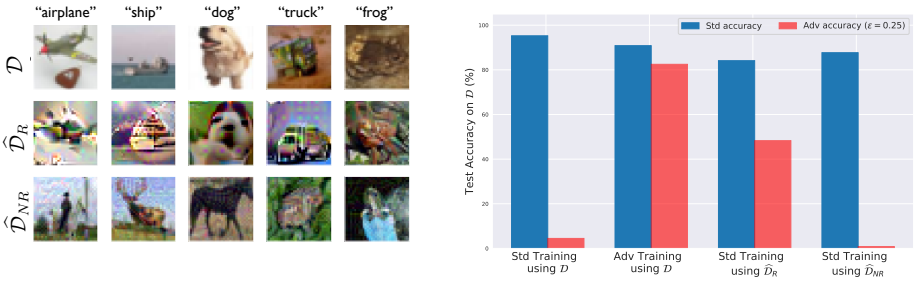
3.8.2 Non-robust features

Based on some ideas from Tsipras et al. [2018], Ilyas et al. [2019] propose an interesting and experimentally well supported hypothesis on the nature of features that well-generalizing non-robust classifiers learn. They show that existence of adversarial examples can be directly attributed to existence of non-robust features, "features derived from patterns in the data that are highly predictive but brittle and incomprehensible to humans".

They demonstrate the predictiveness of non-robust features by:

1. constructing a dataset $\mathbb{D}_{\mathsf{NR}}$ where approximately the only useful features are non-robust features by turning inputs of the original dataset $\mathbb{D}$ into adversarial examples for a classifier that was trained standardly and relabeling them with the adversarial label,
2. training a new classifier on the non-robust dataset $\mathbb{D}_{\mathsf{NR}}$,
3. testing the new classifier on the original test set where it achieves performance close to the original classifier and lower robustness.

In another experiment, they try to remove non-robust-features from inputs with the help of an adversarially trained classifier. A new classifier trained on the dataset with removed non-robust features achieves a bit lower performance and robustness not as high as the adversarially trained classifier but quite significant compared to the standardly trained classifier. Results of these experiments with the CIFAR-10 dataset [Krizhevsky, 2009] are shown in figure 3.6.



(a) (b)

Figure 3.6: (a) Random samples from variants of the CIFAR-10 training set: the original training set; the *robust training set* $\mathbb{D}_{\text{R}}$, restricted to features used by a robust model; and the *non-robust training set* $\mathbb{D}_{\text{NR}}$, restricted to features relevant to a standard model (labels appear incorrect to humans). (b) Standard and robust accuracy on the CIFAR-10 test set ($\mathbb{D}$) for models trained with standard training, adversarial training, and standard training on datasets $\mathbb{D}_{\text{R}}$ (robust) and $\mathbb{D}_{\text{NR}}$ (non-robust). Adapted from Ilyas et al. [2019].

3.8.3 Training with on-manifold adversarial examples

Stutz et al. [2018] challenge the hypothesis that there is a trade-off between robustness and generalization. They hypothesize that most adversarial examples come from directions orthogonal to the learned class manifolds and that training adversarial examples (as per a definition similar to definition 1) limited to the known or learned class manifolds (on-manifold adversarial examples) can improve generalization. They conduct experiments with a synthetic dataset with known class-invariant transformations and datasets with small images that support the hypothesis that generalization can be improved with adversarial training with on-manifold adversarial examples.

Class manifolds to which adversarial examples are restrited and on-manifold and off-manifold adversarial examples are illustrated in figure 3.7.

In one of the experiments Stutz et al. [2018] construct a synthetic dataset with a known manifold (geometric transformations of letters) in order to be able to generate exactly on-manifold adversarial examples by modifying parameters of the geometric transformations. With this dataset, they succeed in improving generalization and on-manifold robustness[1] with adversarial training.

In other experiments they use EMNIST [Cohen et al., 2017], FashionMNIST [Xiao et al., 2017] and CelebA [Liu et al., 2015]. In order to better approximate class manifolds and disable leaving the manifold of a class when an adversarial

[1] By the authors' definition of an adversarial example, which is similar to the consistent definition (definition 1), except for that there is no closeness constraint, making it equivalent to the definition of a misclassified example, on-manifold robustness essentially boils down to generalization.

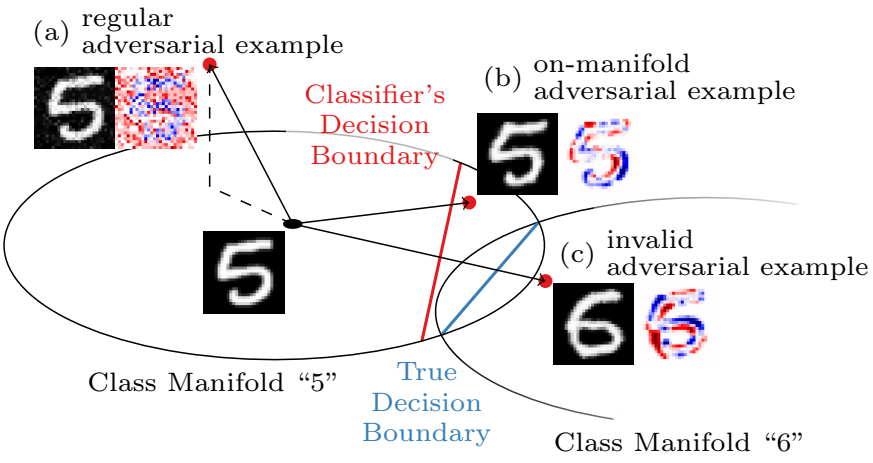


Figure 3.7: An illustration by Stutz et al. [2018] of class manifolds (classes "5" and "6") with a regular (off-manifold) adversarial example and an on-manifold adversarial example.

example is generated for adversarial training, they first train one variational autoencoder (VAE-GAN) per class. They perform training and evaluation analogously to the experiment with synthetic data by allowing the attack to perturb the latent representation of the autoencoder corresponding to the correct class. They measure positive correlation between robustness to on-manifold adversarial examples and generalization. They observe worse quality of on-manifold adversarial examples for the more complex dataset CelebA due to worse approximation quality of their VAE-GAN-s.

3.9 Conclusion

This paper presents an overview of ideas related to the existence adversarial examples, hypotheses on their existence and some of their properties. It describes general principles regarding adversarial attacks, defenses and robustness evaluation and presents examples of such algorithms. Finally, some recent discoveries and hypotheses regarding the relation between robustness and generalization are explored. Some recent results [Stutz et al., 2018] suggest that finding ways of improving both robustness generalization might be an interesting research direction to explore.

.

Adversarially robust generalization requires more data show

With some reasonable assumptions about the model, as the model approaches the true model (in robustness/), its (/on-manifold) adversarial examples would be becoming more close to the true decision boundary, i.e. truly more ambigous (/more often actually changing the class). The true model has no on-manifold adversarial examples, but it might not be defined for off-manifold examples, e.g. if it is a discriminative model.

3.9.1 Robustness evaluation

Athalye et al. [2018] state that, if there are classes that are very similar, using targeted attacks for robustness evaluation can give more meaningful results.

stronger attack targeted, stronger evaluation untargeted

Adversarial training

Kurakin et al. [2016b] note that by using the true label in the loss in untargeted attacks (y in equation (**??**)) can cause

.

Distance metrics for images

Usually, an L^p distance ($d(\boldsymbol{x}', \boldsymbol{x}) = \|\tilde{\boldsymbol{x}} - \boldsymbol{x}\|_p$) is used as a distance metric for adversarial examples.

Scale-invariant norms. Let $\boldsymbol{x}$ denote some image (or perturbation) and $\boldsymbol{x}_\lambda$ the same image with dimensions scaled by λ. $\boldsymbol{x}_\lambda$ has a greater norm because it contains λ^2 the number of pixels of the original image and every pixel is approximately effectively repeated λ^2 times. In terms of $\|\boldsymbol{x}\|_p$, its norm can be approximated like this:

$$\begin{aligned}
\|\boldsymbol{x}_\lambda\|_p &= \left(\sum_{u \in \{0..\lambda H\}} \sum_{v \in \{0..\lambda W\}} \left|\boldsymbol{x}_{\lambda[u,v]}\right|^p \right)^{\frac{1}{p}} && \text{(norm definition)} \\
&\approx \left(\sum_{u \in \{0..H\}} \sum_{v \in \{0..W\}} \lambda^2 \left|\boldsymbol{x}_{[u,v]}\right|^p \right)^{\frac{1}{p}} && (\lambda^2 \text{ element copies}) \\
&= \lambda^{\frac{2}{p}} \left(\sum_{u \in \{0..H\}} \sum_{v \in \{0..W\}} \left|\boldsymbol{x}_{[u,v]}\right|^p \right)^{\frac{1}{p}} \\
&= \lambda^{\frac{2}{p}} \|\boldsymbol{x}\|_p. && \text{(norm definition)}
\end{aligned}$$

If there is a perturbation in 2 different resolutions, the higher-resolution one will have a greater norm by approximately a factor of $(\lambda^2)^{\frac{1}{p}}$, especially if the perturbations don't have too high spatial frequencies. Hence, scale invariant equivalents of L^p norms can be defined by dividing the norm by the scale factor. If we choose the scale factor to be relative to the scale of an image with area 1, λ^2 is n, the total number of pixels.

We can define **scale-invariant norms** like this:

$$\mathrm{m}_p(\boldsymbol{x}) \coloneqq n^{-\frac{1}{p}} \|\boldsymbol{x}\|_p, \tag{3.10}$$

This can also be expressed as the **generalized mean** M_p (also known as **power mean**) of the elementwise absolute value:

$$\mathrm{m}_p(\boldsymbol{x}) = \mathrm{M}_p(|\boldsymbol{x}|) \coloneqq \left(\frac{1}{n} \sum_i \left|\boldsymbol{x}_{[i]}\right|^p \right)^{\frac{1}{p}}. \tag{3.11}$$

Such norms could probably be useful for comparison of norms between different-resolution images and different datasets, and hyperparameter choice for adversarial attacks.

Maybe something similar could be done about objects of different scale?

...

Expectation of scale-invariant norms uniformly distributed random vectors. Let $\underline{\boldsymbol{x}}_n$ be a random vector with n independent elements $\underline{\boldsymbol{x}}_{n\,[i]} \sim \mathcal{U}([-\epsilon, \epsilon])$.

An easy special case is when $p = 1$:

$$\mathbf{E}[\mathrm{m}_1(\underline{\boldsymbol{x}}_n)] = \mathbf{E}\left[\frac{1}{n}\sum_i \left|\underline{\boldsymbol{x}}_{n\,[i]}\right|\right] = \mathbf{E}\left[\left|\underline{\boldsymbol{x}}_{n\,[i]}\right|\right] = \frac{\epsilon}{2}.$$

For very large n, we can approximate it like this:

$$\begin{aligned}
\lim_{n\to\infty} \mathbf{E}[\mathrm{m}_p(\underline{\boldsymbol{x}}_n)] &= \mathbf{E}\left[\lim_{n\to\infty}\left(\frac{1}{n}\sum_i \left|\underline{\boldsymbol{x}}_{n\,[i]}\right|^p\right)^{\frac{1}{p}}\right] && \text{(dominated convergence)} \\
&= \mathbf{E}\left[\mathbf{E}\left[\left|\underline{\boldsymbol{x}}_{n\,[0]}\right|^p\right]^{\frac{1}{p}}\right] && \text{(IID elements)} \\
&= \mathbf{E}\left[\left|\underline{\boldsymbol{x}}_{n\,[0]}\right|^p\right]^{\frac{1}{p}} && \text{(redundant expectation)} \\
&= \frac{1}{2\epsilon}\left(\int_{-\epsilon}^{\epsilon} |x|^p \,\mathrm{d}x\right)^{\frac{1}{p}} && \text{(definition of expectation)} \\
&= \frac{1}{\epsilon}\left(\int_{0}^{\epsilon} x^p \,\mathrm{d}x\right)^{\frac{1}{p}} && \text{(absolute value symmetry)} \\
&= \frac{1}{\epsilon}\left(\frac{\epsilon^{p+1}}{p+1}\right)^{\frac{1}{p}}. && (3.12)
\end{aligned}$$

Specially, for $\epsilon = 1$, it evaluates to

$$\lim_{n\to\infty} \mathbf{E}[\mathrm{m}_p(\underline{\boldsymbol{x}}_n)] = (p+1)^{-\frac{1}{p}}. \tag{3.13}$$

TODO: Show that (3.12) is the upper bound if $p > 1$ and the lower bound if $p < 1$.

By the power mean inequality, which can be proved by applying the Jensen inequality, $\mathrm{m}_p(\underline{\boldsymbol{x}})$ is monotonically increasing in p (for $p > 0$).

For easier analysis, let's first consider a special case: $\boldsymbol{x}_{n[i]} \sim \mathcal{U}([0,1])$.

$$\begin{aligned}\mathbf{E}[\mathrm{m}_p(\boldsymbol{x}_n)] &= \mathbf{E}\left[\left(\frac{1}{n}\sum_i |\boldsymbol{x}_{n[i]}|^p\right)^{\frac{1}{p}}\right] \\ &= \int\cdots\int_{[0,1]^n}\left(\frac{1}{n}\sum_i |\boldsymbol{x}_{[i]}|^p\right)^{\frac{1}{p}} \mathrm{d}\boldsymbol{x} \\ &= n^{-\frac{1}{p}}\int\cdots\int_{[0,1]^n}\left(\sum_i |\boldsymbol{x}_{[i]}|^p\right)^{\frac{1}{p}} \mathrm{d}\boldsymbol{x}\end{aligned}$$

Local p-norm and hiererchical norm. Let $\boldsymbol{k}$ denote a non-negative 2-D kernel with $\|\boldsymbol{k}\|_1 = 1$, e.g. Gaussian cenetered at $(0,0)$. Let $\boldsymbol{x}$ denote an image perturbation and assume that it has a single channel for simplicity. We can define the local p-norm around a pixel (i,j) as

$$\mathrm{LocalNorm}_{p,\boldsymbol{k}}(\boldsymbol{x})_{[i,j]} = (|\boldsymbol{x}|^p * \boldsymbol{k})^{\frac{1}{p}}_{[i,j]}, \tag{3.14}$$

where the absolute value and powering operatons are elementwise.

We can then define a bi-level hierarchical (p_1, p_2)-norm as $\|\mathrm{LocalNorm}_{p_1,\boldsymbol{k}}(\boldsymbol{x})\|_{p_2}$. This can be generalizad to a multi-level norm $(p_1, .., p_n)$-norm by chaining multiple local norms with potentially different kernels until the last, global, p_n-norm.

Why?

Chapter 4

Other

4.1 Generative adversarial networks

4.1.1 Getting the probability of the example from the generator

first paragraph

case 1) normal generator

case 2) invertible generator

4.2 Paper summaries

4.2.1 The Conditional Entropy Bottleneck (Anonymous, 2018)

URL: https://openreview.net/forum?id=rkVOXhAqY7.

4.3 p

Neka je odnos između slučajnih varijabli $\underline{x}$ i $\underline{y}$ definiran funkcijom f koja ishode jedne slučajne varijable deterministički preslikava u ishode druge, što označavamo ovako: $\underline{y} = f(\underline{x})$. Ako su $\underline{x}$ i $\underline{y}$ diskretne slučajne varijable, onda je razdioba

slučajne varijable $\underline{y}$ definirana ovako:

$$P_{\underline{y}}(y) = \sum_{x\colon\, f(x)=y} P_{\underline{x}}(x). \tag{4.1}$$

Ako su $\underline{x}$ i $\underline{y}$ kontinuirane slučajne varijable s vrijednostima iz $\mathbb{R}$ i f je injektivna, može se pokazati [Elezović, 2007] da vrijedi

$$p_{\underline{y}}(y) = p_{\underline{x}}(x)\left|\frac{\mathrm{d}x}{\mathrm{d}y}\right|. \tag{4.2}$$

Neka je $C_{\underline{x}}(x) := \int_{-\infty}^{x} p_{\underline{x}}(x')\,\mathrm{d}x'$. Vrijednosti iz intervala $(x, x+\epsilon)$ na kojem je f monotono rastuća preslikavaju se u interval $(f(x), f(x+\epsilon))$. Granice su obrnute ako je f monotono padajuća na tom intervalu. Budući da $\mathrm{P}(\underline{x} \in (x, x+\epsilon)) = \mathrm{P}(\underline{y} \in (f(x), f(x+\epsilon)))$, vrijedi

$$C_{\underline{x}}(x+\epsilon) - C_{\underline{x}}(x) = C_{\underline{y}}(f(x+\epsilon)) - C_{\underline{y}}(f(x)). \tag{4.3}$$

Ako obje strane jednadžbe dijelimo s ϵ i pustimo $\epsilon \to 0$,

$$\lim_{\epsilon\to 0} \frac{C_{\underline{x}}(x+\epsilon) - C_{\underline{x}}(x)}{\epsilon} = \lim_{\epsilon\to 0} \frac{C_{\underline{y}}(f(x+\epsilon)) - C_{\underline{y}}(f(x))}{\epsilon}. \tag{4.4}$$

Redom, prema definiciji derivacije, pravilu derivacije složene funkcije i definiciji funkcija $C_{\underline{\boldsymbol{x}}}$ i $C_{\underline{\boldsymbol{y}}}$ kao integrala gustoće vjerojatnosti, slijedi:

$$\frac{\mathrm{d}}{\mathrm{d}x} C_{\underline{x}}(x) = \frac{\mathrm{d}}{\mathrm{d}x} C_{\underline{y}}(f(x)), \tag{4.5}$$

$$\frac{\mathrm{d}}{\mathrm{d}x} C_{\underline{x}}(x) = \frac{\mathrm{d}}{\mathrm{d}f(x)} C_{\underline{y}}(f(x)) \frac{\mathrm{d}}{\mathrm{d}x} f(x), \tag{4.6}$$

$$p_{\underline{x}}(x) = p_{\underline{y}}(f(x)) \frac{\mathrm{d}}{\mathrm{d}x} f(x). \tag{4.7}$$

Može se pokazati da je za monotono padajuće intervale desna strana jednadžbe (4.7) pomnožena s -1, iz čega uz jednadžbu (4.7) slijedi

$$p_{\underline{x}}(x) = p_{\underline{y}}(y)\left|\frac{\mathrm{d}y}{\mathrm{d}x}\right|, \tag{4.8}$$

gdje je $f(x)$ zamijenjen s y. Množenjem toga s $\left|\frac{\mathrm{d}x}{\mathrm{d}y}\right| = \left|\frac{\mathrm{d}y}{\mathrm{d}x}\right|^{-1}$ slijedi jednadžba (4.2). To pravilo se može poopćiti i na vektore. Onda vrijedi [Murphy, 2012]

$$p_{\underline{\boldsymbol{y}}}(\boldsymbol{y}) = p_{\underline{\boldsymbol{x}}}(\boldsymbol{x})\left|\det \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{y}}\right|. \tag{4.9}$$

Neka je $\underline{z}$ zbroj slučajnih varijabli $\underline{x}$ i $\underline{y}$. Onda vrijedi

$$p_{\underline{z}}(z) = \int p_{\underline{x},\underline{y}}(x, z - x)\,\mathrm{d}x. \tag{4.10}$$

Ako su $\underline{x}$ i $\underline{y}$ nezavisne, onda to postaje konvolucija:

$$p_{\underline{z}}(z) = \int p_{\underline{x}}(x) p_{\underline{y}}(z - x)\,\mathrm{d}x =: (p_{\underline{x}} * p_{\underline{y}})(z). \tag{4.11}$$

4.3.1 PDF of vector r.v. defined via a function of a vector r.v.

Let $f \in (\mathbb{R}^n \to \mathbb{R}^m)$ and $\underline{\boldsymbol{y}} = f(\underline{\boldsymbol{x}})$. We want to compute the PDF of $\underline{y}$, or, equivalently, the distribution $\mathrm{p}(\underline{\boldsymbol{y}})$.

$$\frac{\partial \underline{\boldsymbol{y}}}{\partial \underline{\boldsymbol{x}}} \in \mathbb{R}^{m \times n} \tag{4.12}$$

For easier analysis, let's assume that $m = 1$, i.e. $\underline{\boldsymbol{y}}$ is a scalar, and denote it with $\underline{y}$. We want to compute its PDF.

4.4 Dense anomaly detection for dense prediction based on reconstruction error

Pretpostavljamo duboki diskriminativni model $h(\boldsymbol{x}; \boldsymbol{\theta})$ s parametrima $\boldsymbol{\theta}$ koji ulaz $\boldsymbol{x}$ preslikava u vektor $\boldsymbol{y}$ koji predstavlja izlaznu razdiobu $\mathrm{p}(\underline{y} \mid \boldsymbol{x}, \boldsymbol{\theta})$.

previsoka sigurnost (postizanje male pogreške na skupu za učenje, kalibracija temperaturnim skaliranjem)

kriva klasifikacija izvanrazdiobnih primjera

neprijateljski primjeri

[Hendrycks and Gimpel, 2016]

[Guo et al., 2017]

[Lee et al., 2017]

[Liang et al., 2017]

Neki pristupi za prepoznavanje anomalija/izvanrazdiobnih primjera (detaljnije opisati i s referencama):

- iz predikcije – očekujemo manju vjerojatnost i veću nesigurnost za izvanrazdioben primjere,

- iz neke skrivene reprezentacije – možemo analizirati razdiobe logita ili nečega drugoga i pomoću toga propoznavati izvanrazdiobne primjere,
- eksplicitnim učenjem razlikovanja razdiobe skupa za učenje od neke pozadinske razdiobe,
- korištenje generativnog modela za generiranje primjera iz područja male gustoće vjerojatnosti i korištenje njih kao izvanrazdiobnih primjera
- korištenjem generativnog modela kod kojeg je moguće izračunati gustoću vjerojatnosti za primjer,
- korištenjem rekonstrukcijske pogreške autoenkodera.

Neki pristup ise mogu kombinirati.

4.4.1 Autoencoders and GAN-s

.

4.4.2 Korištenje rekonstrukcijske pogreške autoenkodera za propoznavanje onoga što model ne zna da ne zna

Pretpostavljamo duboki diskriminativni model $h(\boldsymbol{x};\boldsymbol{\theta})$ s parametrima $\boldsymbol{\theta}$ koji ulaz $\boldsymbol{x}$ preslikava u vektor $\boldsymbol{y}$ koji predstavlja izlaznu razdiobu $\mathrm{p}(\underline{y} \mid \boldsymbol{x}, \boldsymbol{\theta})$.

Korištenje autoenkodera za prepoznavanje izvanrazdiobnih primjera

Sabokrou et al. [2018] za otkrivanje anomalija u slici predlažu korištenje autoenkodera (s jako velikom skrivenom reprezentacijom) kojemu se kod učenja kao ulaz daje zašumljena slika. Uz autoenkoder se dodaje diiskriminator koji se uči a razlikuje izlaz autoenkodera od stvarnih primjera za učenje. Kao gubitak se koristi težinski zbroj kvadratne rekonstrukcijske pogreške i suparničkog gubitka. Kao primjeri se koriste mali izrazani dijelovi većih slika. Za prepoznavanje anomalija koristi se izlaz diskriminatora za rekonstruirani primjer.

Pidhorskyi et al. [2018] isto predlažu pristup s autoenkoderom i superničkim gubitkom. Kod njih gubitak ima 3 komponente: (1) suparnički gubitak koji potiče da primjeri za učenje "pokrivaju" cijelu zadanu (Gaussovu) razdiobu skrivene reprezentacije, (2) suparnički gubitak koji potiče da rekonstruirani primjeri budu iz razdiobe skupa za učenje (kao kod Sabokrou et al. [2018]) i (3) rekonstrukcijski gubitak. Kao mjera za procjenu je li primjer izvan razdiobe se koristi procjena $\mathrm{p}(\boldsymbol{z} \mid \mathbb{D})$ koja ovisi o udaljenosti od "manifolda". Trebam još pručiti kako se točno dobiva.

Pretpostavljamo duboki diskriminativni model $h(\boldsymbol{x};\boldsymbol{\theta})$ s parametrima $\boldsymbol{\theta}$ koji ulaz $\boldsymbol{x}$ preslikava u vektor $\boldsymbol{y}$ koji predstavlja izlaznu razdiobu $\mathrm{p}(\underline{y} \mid \boldsymbol{x}, \boldsymbol{\theta})$. Želimo prepoznavati izvanrazdiobne primjere pomoću autoenkodera.

Neke ideje u vezi autoenkodera:

- Koristiti dekoder s heteroskedastičkom [Kendall and Gal, 2017] nesigurnošću u rekonstrukciju (modelirati $\mathrm{p}(\underline{\boldsymbol{x}} \mid \boldsymbol{z})$) i $-\ln \mathrm{p}(\underline{\boldsymbol{x}} \mid \boldsymbol{z})$ za empirijski gubitak.
- Isprobati rekonstrukciju neke skrivene reprezentacije klasifikatora kako bi se u rekonstrukcijskoj pogrešci naglasile značajke bitne za klasifikaciju (semantički bitne). Možemo h rastaviti na dvije funkcije: $h(\boldsymbol{x}) = (f_2 \circ f_1)(\boldsymbol{x})$ pa onda učime autoenkoder rekonstruirati $f_1(\boldsymbol{x})$. Ako kao kao f_1 koristimo bijekciju, možemo vidjeti kako izgleda rekonstrukcija ulaza koja odgovara rekonstruiranoj reprezentaciji.
- Isprobati klasifikaciju na temelju skrivene reprezentacije autoenkodera $\boldsymbol{z}$, koristiti i klasifikacijski gubitak za učenje kodera, vidjeti kako izgledaju rekonstrukcije. (Isprobati CEB?)
- Minimalna reprezentacija autoenkodera onemogućuje neprijateljske primjere kojima je cilj postići dobru rekonstrukciju anomalije, pogotovo ako pretpostavimo dovoljno dobru funkciju rekonstrukcijske pogreške ili diskriminator.
- Je li dobro poticati da skup za učenje pokriva cijelu razdiobu $\mathrm{p}(\underline{\boldsymbol{z}})$? Onda će različite klase biti odmah jedna uz drugu – malo izmijenimo $\boldsymbol{z}$ i dođemo u područje visoke gustoće za neku drugu klasu. Možda valja učiti razdiobu $\mathrm{p}(\underline{\boldsymbol{z}} \mid \mathbb{D})$ i znati koja su područja niže gustoće (margine) (kako?).
- Dodati šum na ulaz autoenkodera. Možda bi valjalo nešta između gaussovog šuma i "rupa" za popunjavanje.

Osnovni model koji bih htio isprobati (na velikim slikama) je ovakav:

$$\boldsymbol{x} \overset{f_1}{\longmapsto} \boldsymbol{h} \overset{f_2}{\longmapsto} \boldsymbol{y} \quad \text{(klasifikator)}, \tag{4.13}$$

$$\boldsymbol{h} \overset{e}{\longmapsto} \boldsymbol{z} \overset{d}{\longmapsto} \boldsymbol{h}_{\mathrm{r}} \quad \text{(autoenkoder skrivene reprezentacije)}. \tag{4.14}$$

Treba odrediti točan opis modela.

Možemo isprobati i klasifikaciju na temelju rekonstrukcije:

$$\boldsymbol{h}_{\mathrm{r}} \overset{f_2}{\longmapsto} \boldsymbol{y}. \tag{4.15}$$

Možemo isprobati i klasifikaciju na temelju skrivene reprezentacije:

$$\boldsymbol{z} \overset{f_z}{\longmapsto} \boldsymbol{y}, \tag{4.16}$$

i istovremeno učenje klasifikacije i rekonstrukcije.

Bilo bi zanimljivo vidjeti kako izgleda rekonstrukcija ulazne slike na temelju izlaza autoenkodera ovisno o tome koji skriveni sloj se kodira:

$$\boldsymbol{h}_{\mathsf{r}} \overset{f_1^{-1}}{\longmapsto} \boldsymbol{x}_{\mathsf{r}} \quad \text{(inverz prvog dijela klasifikatora s ulazom } \boldsymbol{h}_{\mathsf{r}}\text{)}. \tag{4.17}$$

Rekonstrukciju ulazne slike možemo dobiti ako koristimo neki model koji je bijektivan, npr. i-RevNet.

$$\min_{h} L_{\mathsf{c}}(\boldsymbol{y}, \boldsymbol{y}^*) \tag{4.18}$$

$$\min_{d,e} L_{\mathsf{r}}(\boldsymbol{h}, \boldsymbol{h}_{\mathsf{r}}) \tag{4.19}$$

Što bih još htio isprobati

Kombinaciju Lee et al. [2017] i korištenja izvanrazdiobnih primjera u učenju.

Bibliography

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 274–283, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR. URL http://proceedings.mlr.press/v80/athalye18a.html.

Tom B. Brown, Nicholas Carlini, Chiyuan Zhang, Catherine Olsson, Paul Francis Christiano, and Ian J. Goodfellow. Unrestricted adversarial examples. *CoRR*, abs/1809.08352, 2018. URL http://arxiv.org/abs/1809.08352.

Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, AISec '17, pages 3–14, New York, NY, USA, 2017a. ACM. ISBN 978-1-4503-5202-4. doi: 10.1145/3128572.3140444. URL http://doi.acm.org/10.1145/3128572.3140444.

Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, pages 39–57, 2017b. doi: 10.1109/SP.2017.49. URL https://doi.org/10.1109/SP.2017.49.

Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian J. Goodfellow, Aleksander Madry, and Alexey Kurakin. On evaluating adversarial robustness. *CoRR*, abs/1902.06705, 2019. URL http://arxiv.org/abs/1902.06705.

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. Emnist: an extension of mnist to handwritten letters. *arXiv preprint arXiv:1702.05373*, 2017.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1310–1320, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL http://proceedings.mlr.press/v97/cohen19c.html.

Neven Elezović. *Vjerojatnost i statistika: Slučajne varijable*. Element, 2007.

Logan Engstrom, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. A rotation and a translation suffice: Fooling cnns with simple transformations. *CoRR*, abs/1712.02779, 2017. URL http://arxiv.org/abs/1712.02779.

Y. Gal and L. Smith. Sufficient Conditions for Idealised Models to Have No Adversarial Examples: a Theoretical and Empirical Study with Bayesian Neural Networks. *ArXiv e-prints*, June 2018.

Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 1180–1189. JMLR.org, 2015. URL http://dl.acm.org/citation.cfm?id=3045118.3045244.

Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S. Schoenholz, Maithra Raghu, Martin Wattenberg, and Ian J. Goodfellow. Adversarial spheres. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*, 2018. URL https://openreview.net/forum?id=SkthILkPf.

Justin Gilmer, Nicolas Ford, Nicholas Carlini, and Ekin Cubuk. Adversarial examples are a natural consequence of test error in noise. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2280–2289, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL http://proceedings.mlr.press/v97/gilmer19a.html.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 2672–2680, Cambridge, MA, USA, 2014a. MIT Press. URL http://dl.acm.org/citation.cfm?id=2969033.2969125.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *CoRR*, abs/1412.6572, 2014b. URL http://arxiv.org/abs/1412.6572.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. *CoRR*, abs/1706.04599, 2017. URL http://arxiv.org/abs/1706.04599.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL http://arxiv.org/abs/1512.03385.

Dan Hendrycks and Thomas G. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *CoRR*, abs/1903.12261, 2019. URL http://arxiv.org/abs/1903.12261.

Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *CoRR*, abs/1610.02136, 2016. URL http://arxiv.org/abs/1610.02136.

Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017. URL http://arxiv.org/abs/1704.04861.

Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *CoRR*, abs/1608.06993, 2016. URL http://arxiv.org/abs/1608.06993.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features, 2019. URL http://arxiv.org/abs/1905.02175. cite arxiv:1905.02175.

Jörn-Henrik Jacobsen, Jens Behrmann, Nicholas Carlini, Florian Tramèr, and Nicolas Papernot. Exploiting excessive invariance caused by norm-bounded adversarial robustness. *CoRR*, abs/1903.10484, 2019. URL http://arxiv.org/abs/1903.10484.

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *CoRR*, abs/1703.04977, 2017. URL http://arxiv.org/abs/1703.04977.

J. Kos, I. Fischer, and D. Song. Adversarial examples for generative models. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 36–42, May 2018. doi: 10.1109/SPW.2018.00014.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012. URL http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *CoRR*, abs/1607.02533, 2016a. URL http://arxiv.org/abs/1607.02533.

Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *CoRR*, abs/1611.01236, 2016b. URL http://arxiv.org/abs/1611.01236.

Yann LeCun, Yoshua Bengio, and Geoffrey E. Hinton. Deep learning. *Nature*, 521 (7553):436–444, 2015. doi: 10.1038/nature14539. URL https://doi.org/10.1038/nature14539.

Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. *CoRR*, abs/1711.09325, 2017.

Yingzhen Li. Are generative classifiers more robust to adversarial attacks? *CoRR*, abs/1802.06552, 2018. URL http://arxiv.org/abs/1802.06552.

Shiyu Liang, Yixuan Li, and R. Srikant. Principled detection of out-of-distribution examples in neural networks. *CoRR*, abs/1706.02690, 2017. URL http://arxiv.org/abs/1706.02690.

Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017. URL https://openreview.net/forum?id=Sys6GJqxl.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *CoRR*, abs/1706.06083, 2017. URL http://arxiv.org/abs/1706.06083.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *CoRR*, abs/1704.03976, 2017. URL http://arxiv.org/abs/1704.03976.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. *CoRR*, abs/1610.08401, 2016a. URL http://arxiv.org/abs/1610.08401.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: A simple and accurate method to fool deep neural networks. In *CVPR*, pages 2574–2582. IEEE Computer Society, 2016b.

Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012. ISBN 0262018020, 9780262018029.

Anh Mai Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *CVPR*, pages 427–436. IEEE Computer Society, 2015. ISBN 978-1-4673-6964-0. URL http://dblp.uni-trier.de/db/conf/cvpr/cvpr2015.html#NguyenYC15.

Nicolas Papernot, Patrick D. McDaniel, and Ian J. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *CoRR*, abs/1605.07277, 2016. URL http://dblp.uni-trier.de/db/journals/corr/corr1605.html#PapernotMG16.

Stanislav Pidhorskyi, Ranya Almohsen, and Gianfranco Doretto. Generative probabilistic novelty detection with adversarial autoencoders. In *NeurIPS*, pages 6823–6834, 2018.

Haifeng Qian and Mark N. Wegman. L2-nonexpansive neural networks. *CoRR*, abs/1802.07896, 2018. URL http://arxiv.org/abs/1802.07896.

Mohammad Sabokrou, Mohammad Khalooei, Mahmood Fathy, and Ehsan Adeli. Adversarially learned one-class classifier for novelty detection. In *CVPR*, pages 3379–3388. IEEE Computer Society, 2018.

Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *CoRR*, abs/1805.06605, 2018. URL http://arxiv.org/abs/1805.06605.

Lukas Schott, Jonas Rauber, Matthias Bethge, and Wieland Brendel. Towards the first adversarially robust neural network model on mnist. *CoRR*, abs/1805.09190, 2018. URL http://arxiv.org/abs/1805.09190.

Alexandru Constantin Serban and Erik Poll. Adversarial examples - A complete characterisation of the phenomenon. *CoRR*, abs/1810.01185, 2018. URL http://arxiv.org/abs/1810.01185.

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

David Stutz, Matthias Hein, and Bernt Schiele. Disentangling adversarial robustness and generalization. *CoRR*, abs/1812.00740, 2018.

Dong Su, Huan Zhang, Hongge Chen, Jinfeng Yi, Pin-Yu Chen, and Yupeng Gao. Is robustness the cost of accuracy? - A comprehensive study on the robustness of 18 deep image classification models. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XII*, pages 644–661, 2018. doi: $10.1007/978\text{-}3\text{-}030\text{-}01258\text{-}8\backslash\_39$. URL https://doi.org/10.1007/978-3-030-01258-8_39.

Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *CoRR*, abs/1710.08864, 2017. URL http://arxiv.org/abs/1710.08864.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2013. URL http://arxiv.org/abs/1312.6199.

Pedro Tabacof and Eduardo Valle. Exploring the space of adversarial images. In *2016 International Joint Conference on Neural Networks, IJCNN 2016, Vancouver, BC, Canada, July 24-29, 2016*, pages 426–433, 2016. doi: 10.1109/IJCNN.2016.7727230. URL https://doi.org/10.1109/IJCNN.2016.7727230.

Thomas Tanay and Lewis D. Griffin. A boundary tilting persepective on the phenomenon of adversarial examples. *CoRR*, abs/1608.07690, 2016. URL http://arxiv.org/abs/1608.07690.

Florian Tramèr, Nicolas Papernot, Ian J. Goodfellow, Dan Boneh, and Patrick D. McDaniel. The space of transferable adversarial examples. *CoRR*, abs/1704.03453, 2017. URL http://arxiv.org/abs/1704.03453.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy, 2018. URL http://arxiv.org/abs/1805.12152. cite arxiv:1805.12152.

Jonathan Uesato, Brendan O'Donoghue, Pushmeet Kohli, and Aäron van den Oord. Adversarial risk and the dangers of evaluating against weak attacks. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 5032–5041, 2018. URL http://proceedings.mlr.press/v80/uesato18a.html.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. Learning transferable architectures for scalable image recognition. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 8697–8710, 2018. doi: 10.1109/CVPR.2018.00907. URL http://openaccess.thecvf.com/content_cvpr_2018/html/Zoph_Learning_Transferable_Architectures_CVPR_2018_paper.html.