Lecture 1

Welcome & Intro

Part 1: What This Course Is About

Machine Learning for Structured Data
Vlad Niculae · LTL, UvA · https://vene.ro/mlsd

Machine Learning

Understanding, choosing, designing:

- models
- learning algorithms
- evaluation metrics
- experiment methodology

to learn and evaluate mappings
from inputs x to outputs y.

Machine Learning

Understanding, choosing, designing:

- models
- learning algorithms
- evaluation metrics
- experiment methodology

to learn and evaluate mappings from inputs x to outputs y.

… for Structured Data

structure, *noun: the way in which a complex object's parts are organized in relationship to one another.*

Many objects we want to do ML on have interesting structure:

language, images, shapes, networks…

This course: how to make use of structure in the input and the output.

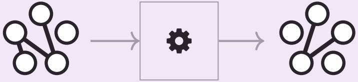
Machine Learning

Understanding, choosing, designing:

- models
- learning algorithms
- evaluation metrics
- experiment methodology

to learn and evaluate mappings from inputs x to outputs y.

... for Structured Data



structure, *noun: the way in which a complex object's parts are organized in relationship to one another.*

Many objects we want to do ML on have interesting structure:

language, images, shapes, networks...

This course: how to make use of structure in the input and the output.

Some things to keep in mind

- Goal: How to make use of structure for ML
  so we expect you to be comfortable with ML basics.

Some things to keep in mind

- Goal: How to make use of structure for ML
  so we expect you to be comfortable with ML basics.
- ML is fast-moving
  popular architectures come and go every few years;
  we'll look a bit deeper, at the timeless underlying principles.

Some things to keep in mind

- Goal: How to make use of structure for ML
  so we expect you to be comfortable with ML basics.
- ML is fast-moving
  popular architectures come and go every few years;
  we'll look a bit deeper, at the timeless underlying principles.
- Structure is common in many domains: we will explore several.
  Language, Vision, Biology, Material Science, Social Science…

Some things to keep in mind

- Goal: How to make use of structure for ML
  so we expect you to be comfortable with ML basics.
- ML is fast-moving
  popular architectures come and go every few years;
  we'll look a bit deeper, at the timeless underlying principles.
- Structure is common in many domains: we will explore several.
  Language, Vision, Biology, Material Science, Social Science…
- Notation: There *will* be differences between classes, books, blogs. Don't assume the same symbol always means the same thing. If in doubt, ask.

Machine Learning Recap

Definition: Supervised ML Task

Find an accurate mapping from x to y
from a labeled dataset $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(N)}, y^{(N)})\}$

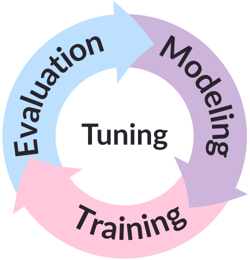
Terminology and notation.

symbol	explanation	example
$x \in \mathcal{X}$	*input object*	measurements of a penguin: (flipper length, bill length, bill depth) $[181, 39.1, 18.7] \in \mathcal{X} = \mathbb{R}^3$
$y \in \mathcal{Y}$	*output label*: the desired true ("gold") output	penguin species $\mathcal{Y} = \{\text{Chinstrap, Gentoo, Adélie}\}$
$\{f_\theta : \theta \in \Theta\}$	model class / architecture / family	linear classifier $f_\theta(x) = Wx + b$
$\theta \in \Theta$	*model parameters* (weights)	$\theta = (W, b)$

ML design

Many choices to make when approaching a ML task.

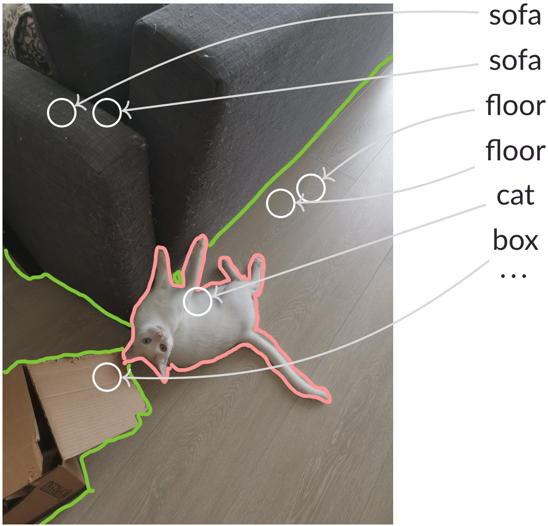
modeling:	• architecture f	(linear model? neural network? decision tree? ...)
	• data encoding	(pixel values? bag-of-words? ...)
	• regularization	($\|\cdot\|_2^2$? dropout? ...)
training:	• training objective / loss	(logistic? hinge? perceptron? ...)
	• learning algorithm	(SGD? Adam? L-BFGS? ...)
evaluation:	• metrics	(accuracy? precision? F_1? ...)
	• visualizations / reports	
tuning:	• validation split / cross-validation	



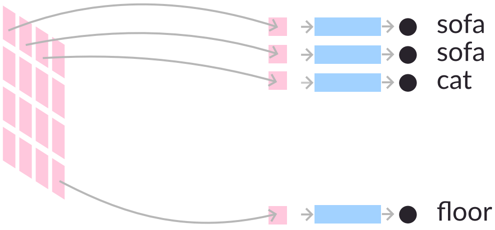
Evaluate on test, deploy.

Case Study: Semantic Image Segmentation

Classify every pixel according to the object it is a part of.



Case Study: Semantic Image Segmentation

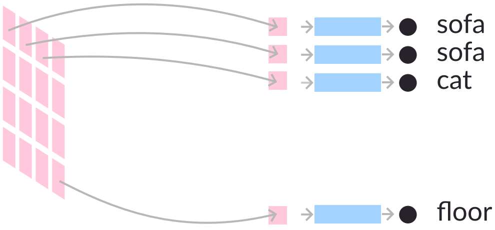


How to model this?

A first idea

$x \in \mathbb{R}^3$ a pixel RGB, e.g., $x = (255, 60, 30)$

$y \in \{$ cat, sofa, floor, box, $\ldots \}$



How to model this?

A first idea

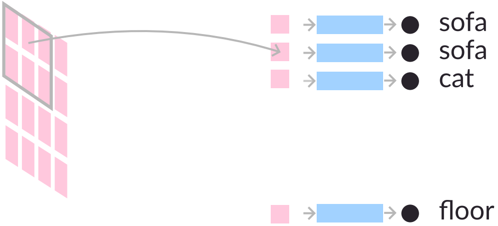
$x \in \mathbb{R}^3$ a pixel RGB, e.g., $x = (255, 60, 30)$

$y \in \{$ cat, sofa, floor, box, $\dots \}$

Acts as if pixels are “IID”:
(independent & identically distributed)

What does this mean, and does it apply?

Case Study: Semantic Image Segmentation



How to model this?

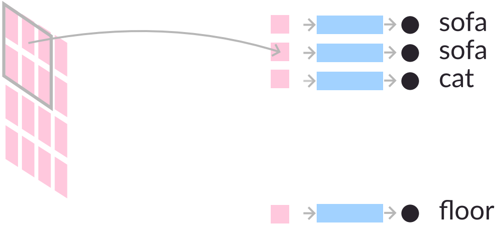
Idea 2: Some structured input context

$x \in \mathbb{R}^{d \times d \times 3}$ a ~~pixel~~ patch of pixels

$y \in \{$ cat, sofa, $\ldots \}$ label of patch center

Structured context helps resolve ambiguous pixels.

Case Study: Semantic Image Segmentation



How to model this?

Idea 2: Some structured input context

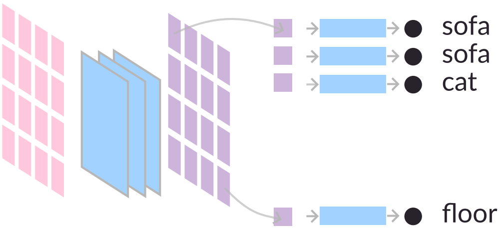
$x \in \mathbb{R}^{d \times d \times 3}$ a ~~pixel~~ patch of pixels

$y \in \{$ cat, sofa, $\dots \}$ label of patch center

Structured context helps resolve ambiguous pixels.

But, only interactions are between nearby pixels.

Case Study: Semantic Image Segmentation



(convolutional network encoder)

How to model this?

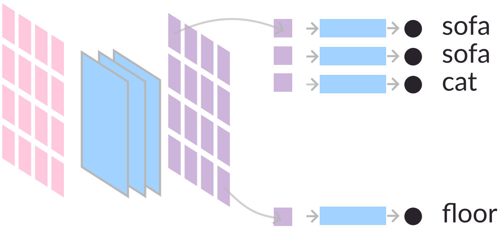
Idea 3: Structured input context – to the max

$x \in \mathbb{R}^{W \times H \times 3}$, an entire image.

encode the image with a structure-aware deep network
(extract patches, recombine, extract again...)

$y \in \{$ cat, sofa, floor, box, $\dots\}^{W \times H}$

Case Study: Semantic Image Segmentation



(convolutional network encoder)

How to model this?

Idea 3: Structured input context – to the max

$x \in \mathbb{R}^{W \times H \times 3}$, an entire image.

encode the image with a structure-aware deep network (extract patches, recombine, extract again...)

$y \in \{ \text{cat, sofa, floor, box, } \ldots \}^{W \times H}$

Make predictions independently for each pixel, but based on *rich representations* of each pixel, that are informed by wider context.

The richer we want the context to be, the larger & more expensive the network needs to be.

Outputs can have structure, too!

$$
y = \begin{pmatrix}
. & . & . & . & . & . & . & . \\
. & . & . & . & . & . & . & . \\
. & . & \mathrm{c} & \mathrm{c} & . & . & . & . \\
. & . & \mathrm{c} & \mathrm{c} & \mathrm{c} & . & . & . \\
. & . & \mathrm{c} & . & . & . & . & . \\
. & . & . & . & . & . & . & . \\
. & . & . & . & . & . & . & \mathrm{b} \\
. & . & . & . & . & . & \mathrm{b} & \mathrm{b}
\end{pmatrix}
$$

- Adjacent labels likely to be the same.
- Nearby labels help <u>disambiguate</u> each other.

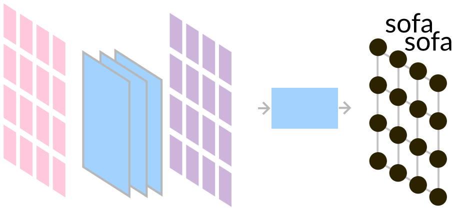
(image from Amaury Guichon’s instagram)

mirror / knife?

drum / cake?

(image from Amaury Guichon’s instagram)

Case Study: Semantic Image Segmentation



(Markov Random Field)

How to model this?

Idea 4: Using output structure

$x \in \mathbb{R}^{W \times H \times 3}$, an entire image.

encode as we want (CNN, simple patches...)

$y \in \{ \text{cat, sofa, floor, box, } \dots \}^{W \times H}$

Predict ~~independently~~ **jointly** over the entire image.

Labels **self-correct** to agree with neighbors.

Which of these models do you know how to train?

1. Pixel-to-label

Which of these models do you know how to train?

1. Pixel-to-label
   preprocess images into a **pixel**-level dataset, apply any clf

Which of these models do you know how to train?

1. Pixel-to-label
   preprocess images into a **pixel**-level dataset, apply any clf
2. Patch-to-label

Which of these models do you know how to train?

1. Pixel-to-label
   preprocess images into a **pixel**-level dataset, apply any clf
2. Patch-to-label
   preprocess images into a **patch**-level dataset, apply any clf

Which of these models do you know how to train?

1. Pixel-to-label
   preprocess images into a **pixel**-level dataset, apply any clf
2. Patch-to-label
   preprocess images into a **patch**-level dataset, apply any clf
3. Convolutional net encoder?

Which of these models do you know how to train?

1. Pixel-to-label
   preprocess images into a **pixel**-level dataset, apply any clf
2. Patch-to-label
   preprocess images into a **patch**-level dataset, apply any clf
3. Convolutional net encoder?
   (covered in first half of this course)

Which of these models do you know how to train?

1. Pixel-to-label
   preprocess images into a **pixel**-level dataset, apply any clf
2. Patch-to-label
   preprocess images into a **patch**-level dataset, apply any clf
3. Convolutional net encoder?
   (covered in first half of this course)
4. Markov Random Field? (interdependent outputs)

Which of these models do you know how to train?

1. Pixel-to-label
   preprocess images into a **pixel**-level dataset, apply any clf
2. Patch-to-label
   preprocess images into a **patch**-level dataset, apply any clf
3. Convolutional net encoder?
   (covered in first half of this course)
4. Markov Random Field? (interdependent outputs)
   (covered in second half of this course)

How to *evaluate*?

$y^{(k)} \in L^{W\times H}$, $L = \{\underline{\text{F}}\text{loor}, \underline{\text{C}}\text{at}, \ldots\}$, collection of labels for entire image.

predicted $\hat{y}^{(k)} = \begin{pmatrix} F & F & F & F \\ F & F & C & F \\ F & C & F & F \\ F & F & F & F \end{pmatrix}$, true $y^{(k)} = \begin{pmatrix} F & F & F & F \\ F & C & C & F \\ F & C & C & F \\ F & F & F & F \end{pmatrix}$.

How to *evaluate*?

$y^{(k)} \in L^{W \times H}, L = \{\underline{\text{F}}\text{loor}, \underline{\text{C}}\text{at}, \ldots\}$, collection of labels for entire image.

predicted $\hat{y}^{(k)} = \begin{pmatrix} F & F & F & F \\ F & F & C & F \\ F & C & F & F \\ F & F & F & F \end{pmatrix}$, true $y^{(k)} = \begin{pmatrix} F & F & F & F \\ F & C & C & F \\ F & C & C & F \\ F & F & F & F \end{pmatrix}$.

- zero-one accuracy (unstructured standard): $\frac{1}{N} \sum_{k=1}^{N} \mathbb{I}[\hat{y}^{(k)} = y^{(k)}]$

Notation: $\mathbb{I}[q] = \begin{cases} 1, & \text{if } q \text{ is true} \\ 0, & \text{otherwise} \end{cases}$ "Iverson bracket"

How to *evaluate*?

$y^{(k)} \in L^{W\times H}$, $L = \{\underline{\text{F}}\text{loor}, \underline{\text{C}}\text{at}, \ldots\}$, collection of labels for entire image.

predicted $\hat{y}^{(k)} = \begin{pmatrix} F & F & F & F \\ F & F & C & F \\ F & C & F & F \\ F & F & F & F \end{pmatrix}$, true $y^{(k)} = \begin{pmatrix} F & F & F & F \\ F & C & C & F \\ F & C & C & F \\ F & F & F & F \end{pmatrix}$.

- zero-one accuracy (unstructured standard): $\frac{1}{N}\sum_{k=1}^{N} \mathbb{1}[\hat{y}^{(k)} = y^{(k)}]$
- Hamming score: $\frac{1}{N}\sum_{k=1}^{N} \frac{1}{WH}\sum_{i,j} \mathbb{1}[\hat{y}_{ij}^{(k)} = y_{ij}^{(k)}]$

Notation: $\mathbb{1}[q] = \begin{cases} 1, & \text{if } q \text{ is true} \\ 0, & \text{otherwise} \end{cases}$ "Iverson bracket"

How to *evaluate*?

$y^{(k)} \in L^{W\times H}$, $L = \{\underline{\text{F}}\text{loor}, \underline{\text{C}}\text{at}, \dots\}$, collection of labels for entire image.

predicted $\hat{y}^{(k)} = \begin{pmatrix} F & F & F & F \\ F & F & C & F \\ F & C & F & F \\ F & F & F & F \end{pmatrix}$, true $y^{(k)} = \begin{pmatrix} F & F & F & F \\ F & C & C & F \\ F & C & C & F \\ F & F & F & F \end{pmatrix}$.

- zero-one accuracy (unstructured standard): $\frac{1}{N}\sum_{k=1}^{N} \mathbb{I}[\hat{y}^{(k)} = y^{(k)}]$
- Hamming score: $\frac{1}{N}\sum_{k=1}^{N} \frac{1}{WH}\sum_{i,j} \mathbb{I}[\hat{y}_{ij}^{(k)} = y_{ij}^{(k)}]$
- Problem-specific costs (e. g. , intersection-over-union, overlap%...)

Notation: $\mathbb{I}[q] = \begin{cases} 1, & \text{if } q \text{ is true} \\ 0, & \text{otherwise} \end{cases}$ "Iverson bracket"

How to *evaluate*?

$y^{(k)} \in L^{W\times H}$, $L = \{\underline{F}\text{loor}, \underline{C}\text{at}, \dots\}$, collection of labels for entire image.

predicted $\hat{y}^{(k)} = \begin{pmatrix} F & F & F & F \\ F & F & C & F \\ F & C & F & F \\ F & F & F & F \end{pmatrix}$, true $y^{(k)} = \begin{pmatrix} F & F & F & F \\ F & C & C & F \\ F & C & C & F \\ F & F & F & F \end{pmatrix}$.

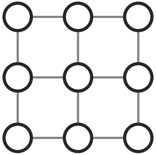
- zero-one accuracy (unstructured standard): $\frac{1}{N}\sum_{k=1}^{N} [\![\hat{y}^{(k)} = y^{(k)}]\!]$
- Hamming score: $\frac{1}{N}\sum_{k=1}^{N} \frac{1}{WH}\sum_{i,j} [\![\hat{y}^{(k)}_{ij} = y^{(k)}_{ij}]\!]$
- Problem-specific costs (e. g. , intersection-over-union, overlap%…)

Notation: $[\![q]\!] = \begin{cases} 1, & \text{if } q \text{ is true} \\ 0, & \text{otherwise} \end{cases}$ "Iverson bracket"

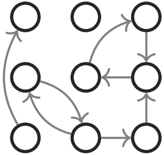
Structured evaluation needs more consideration than unstructured.

A few examples of structure

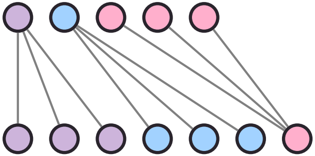
Sequence



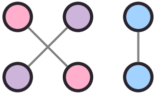
Grid



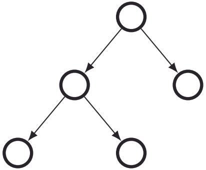
Graph



Alignments

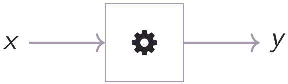


Permutations



Hierarchy

Machine Learning

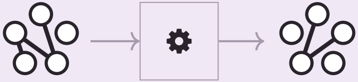


Understanding, choosing, designing:

- models
- learning algorithms
- evaluation metrics
- experiment methodology

to learn and evaluate mappings from inputs x to outputs y.

... for Structured Data



structure, *noun: the way in which a complex object's parts are organized in relationship to one another.*

Many objects we want to do ML on have interesting structure:

language, images, shapes, networks…

This course: how to make use of structure in the input and the output.