Introduction to Machine Learning for NLP I

Benjamin Roth, Nina Poerner, Marina Speranskaya

CIS LMU München

Outline

Course Overview

- Foundations of machine learning
  - loss functions
  - linear regression
  - logistic regression
  - gradient-based optimization
  - neural networks and backpropagation
- Deep learning tools in Python
  - Numpy
  - Pytorch
  - Keras
  - (some) Tensorflow?
- Architectures for NLP
  - CNNs, RNNs, Self-Attention (Transformer)
- Applications
  - Word Embeddings
  - Sentiment Analysis
  - Relation extraction
  - Practical project (NLP related, optional)

Lecture Times, Tutorials

- Course homepage:
  `dl-nlp.github.io`
- This is where exercise sheets and lecture slides are posted
- 9-11 is supposed to be the lecture slot, and 11-12 the tutorial slot ...
- ... but we will not stick to that allocation
- We will sometimes have longer Q&A-style/interactive "tutorial" sessions, sometimes more lectures (see next slide)
- Tutor: Marina Speranskaya
  - Will discuss exercise sheets in the tutorials
  - Will help you with the project

Plan

	9-11 slot	11-12 slot	Ex. sheet
10/16	Overview / ML Intro I		Reading: Linear algebra
10/23	Linear algebra Q&A / ML II	ML II	Reading: Probability
10/30	Probability Q&A / ML III	Numpy	Numpy
11/6	Pytorch Intro	Pytorch	Pytorch
11/13	Word2Vec	Numpy Q&A	Pytorch/Word2Vec

	9-11 slot	11-12 slot	Ex. sheet
11/20	RNNs, Pytorch Q&A	Word2Vec Q&A	Reading: LSTM/GRU
11/27	LSTM discussion	Keras	Keras/Tagging
12/4	CNN	Attention / BERT	Keras/CNN
12/11	Attention / BERT	Keras/Tagging Q&A	
12/18	Project announcement	Keras/CNN Q&A	–

	9-11 slot	11-12 slot	Ex. sheet
1/8	Exam	–	–
1/15	Regularization	Help with projects	–
1/22	Hyperparameters	Help with projects	–
1/29	Project Q&A	Projects Q&A	–
2/5	Project presentations	presentations	–

Formalities

- This class is graded by a written exam (Klausur) in the week after Christmas
- Additional bonus points can be earned by:
  - Exercise sheets (before Christmas)
  - Project and presentation (after Christmas)
- If you got more than 50% of possible bonus points, they count for up to 10% of the exam.
- Formula:

$$g_{\text{final}} = \min(M, g_{\text{exam}} + \frac{M}{10} \cdot \max(0, 2 \cdot (g_{\text{bonus}} - 0.5)))$$

$$g_{\text{bonus}} = \frac{1}{3} g_{\text{project}} + \frac{2}{3} g_{\text{exercises}}$$

- where M is the maximum possible number of points.

Work load

- 6 ECTS, 14 weeks
  $\Rightarrow$ avg work load $\sim$ 13hrs / week (3 in class, 10 at home)
  - in the first weeks, spend enough time to read and prepare so that you are not lost later
  - from beginning of November to Christmas: programming assignments - coding takes time, and can be frustating (but rewarding)!

Exam

- Will cover material from the lectures and reading assignments up to Christmas
- So even if you do not hand the reading assignments, it is a good idea to read them.
- Mostly conceptual questions, no code (no need to learn pytorch function names by heart!)

Exercise sheets

- Optional (bonus points!)
- Exercise sheets 1, 2 and 5 are reading assignments with questions
- Other sheets are programming exercises
- Format: jupyter notebooks
- All exercise sheets contribute equally

Project

- Optional (bonus points!)
- Project topic & data will be distributed before Christmas
- You should work in groups of 2 or 3
- All groups work on the same data
- Must hand in code
- Grading scheme TBA: probably a combination of results (good performance on test set), code quality, creativity
- Optional project presentation in the last week before Easter (may give bonus points, TBA)

Good project code ...

- ... shows that you master the techniques taught in the lectures and exercises.
- ... shows that you can make **"own decisions"**: e.g. adapt model / task / training data etc if necessary.
- ... is well-structured and easy to understand (telling variable names, meaningful modularization – avoid: code duplication, dead code)
- ... is correct
- ... is within the scope of this lecture (time-wise should not exceed $4 \times 10h$)

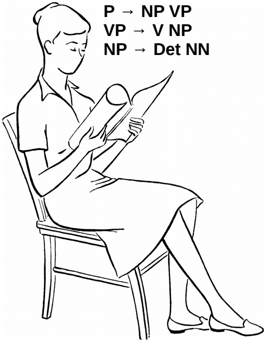
A good project presentation ...

- ... is short (10 min. per team)
- ... is targeted to your fellow students, who do not know details beforehand
- ... contains **interesting stuff**: unexpected observations? conclusions / recommendations? did you deviate from some common practice?
- ... demonstrates that all team members worked together on the project

Outline

Machine Learning for NLP

- What are the alternatives?
- Advantages and disadvantages?
  - Accuracy
  - Coverage
  - Resources required (data, expertise, human labour)
  - Reliability/Robustness
  - Explainability



Outline

A Definition

"A computer program is said to learn from **experience** E with respect to some class of **tasks** T and **performance measure** P, if its performance at tasks in T, as measured by P, improves with experience E."
(Mitchell 1997)

A Definition

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."
(Mitchell 1997)

- Learning: Attaining the ability to perform a task.
- A set of examples (*"experience"*) represents a more general task.
- Examples are described by *features*:
  sets of numerical properties that can be represented as vectors $\mathbf{x} \in \mathbb{R}^n$.

Outline

Data

"A computer program is said to learn from **experience** E [...], if its performance [...] improves with **experience** E."

- Dataset: collection of examples
- Design matrix

$$\mathbf{X} \in \mathbb{R}^{n \times m}$$

  - n: number of examples
  - m: number of features
  - Example: $X_{i,j}$ count of feature j (e.g. a stem form) in document i, intensity of j'th pixel in image i
- Unsupervised learning:
  - Model $\mathbf{X}$, or find interesting properties of $\mathbf{X}$.
  - Example: Clustering (find groups of similar images/documents)
  - Training data: only $\mathbf{X}$.
- Supervised learning:
  - Predict *specific* additional properties from $\mathbf{X}$.
  - E.g., sentiment classification: Predict sentiment (1–5) of amazon review
  - Training data: Label vector $\mathbf{y} \in \mathbb{R}^n$ together with $\mathbf{X}$

Outline

Machine Learning Tasks

"A computer program is said to learn [...] with respect to some class of **tasks T** [...] if its performance at **tasks in T** [...] improves [...]"

Types of Tasks:

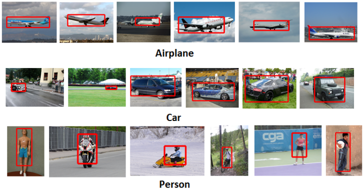
- Classification
- Regression
- Structured Prediction
- Anomaly Detection
- synthesis and sampling
- Imputation of missing values
- Denoising
- Clustering
- Reinforcement learning
- ...

Task: Classification

- Which of k classes does an example belong to?

$$f : \mathbb{R}^n \to \{1 \ldots k\}$$

- Typical example: Categorize image patches
  - Feature vector: color intensities for each pixel; derived features.
  - Output categories: Predefined set of labels



- Typical example: Spam Classification
  - Feature vector: High-dimensional, sparse vector.
    Each dimension indicates occurrence of a particular word, or other email-specific information.
  - Output categories: *"spam"* vs. *'ham"*

Task: Classification

Identifying civilians killed by police with distantly supervised entity-event extraction

Katherine A. Keith, Abram Handler, Michael Pinkham,
Cara Magliozzi, Joshua McDuffie, and **Brendan O'Connor**
College of Information and Computer Sciences
University of Massachusetts Amherst

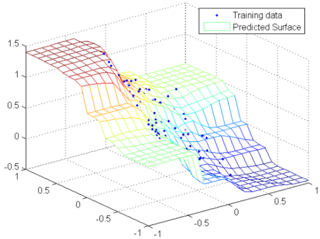
- EMNLP 2017: Given a **person name** in a sentence that contains **keywords** related to **police** ("officer", "police" ...) and to **killing** ("killed", "shot"), was the person a civilian killed by police?

Text	Person killed by police?
Alton Sterling was killed by police.	True
Officers shot and killed **Philando Castile**.	True
Officer **Andrew Hanson** was shot.	False
Police report **Megan Short** was fatally shot in apparent murder-suicide.	False

Task: Regression

- Predict a numerical value given some input.

$$f : \mathbb{R}^n \rightarrow \mathbb{R}$$



- Typical examples:
  - Predict the risk of an insurance customer.
  - Predict the value of a stock.

Task: Regression

Towards an Automatic Turing Test: Learning to Evaluate Dialogue Responses

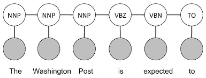
Ryan Lowe[♡*] **Michael Noseworthy**[♡*] **Iulian V. Serban**[◇]
Nicolas A.-Gontier[♡] **Yoshua Bengio**[◇‡] **Joelle Pineau**[♡‡]

- ACL 2017: Given a response in a multi-turn dialogue, predict the value (on a scale from 1 to 5) how natural a response is.

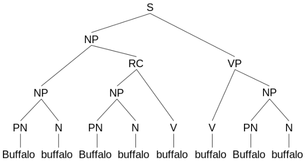
Context	Reference response	Model responses	Human score	ADEM score
photo to see my television debut go to - some. some on <url> - hehe <url> → it really was you? i thought ppl were recognizing someone who looked like you! were the oysters worth the wait?	yeah it was me . haha i 'd kinda forgotten about it it was filmed a while ago	1) i'm not sure. i just don't know what to do with it.	3	1.602
		2) you heard the horsepower productions remix of lee scratch perry's 'exercising' off his 'mighty upsetter' album?	1	1.513
		3) you wont chug a fuzzy peach navel	1	1.744
		4) they were!	5	3.274
just beat call of duty!! → want a cookie? → yes!! → come get it	im in kenmore at the moment	1) i'm gonna get a new phone some moro	1	1.848
		2) no way man.	5	4.265
		3) wow i just got a free pizza coupon! get yours before theres no more! <url>	1	0.921
		4) i'm going to go to the mall.	1	2.634
am i out of twitter jail yet? testing → yeah. i posted bail → thanks. i am a right chatter tweetbox on sundays. same happened last sunday lol	any news on meeting our user ? i go to the us on friday and i don 't want to miss anything arranged	1) i'm not sure if i'm going to be able to get it.	3	1.912
		2) good to see another mac user in the leadership ranks	4	1.417
		3) awww poor baby hope u get to feeling better soon. maybe some many work days at piedmont	2	1.123
		4) did you tweet too much?	5	2.539

Task: Structured Prediction

- Predict a multi-valued output with special inter-dependencies and constraints.
- Typical examples:
  - Part-of-speech tagging



  - Syntactic parsing



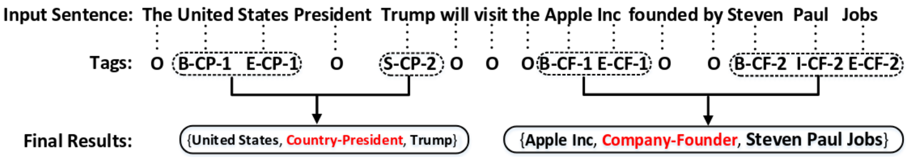
  - Machine Translation
- Often involves search and problem-specific algorithms.

Task: Structured Prediction

**Joint Extraction of Entities and Relations
Based on a Novel Tagging Scheme**

Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao,Peng Zhou, Bo Xu
Institute of Automation, Chinese Academy of Sciences, 100190, Beijing, P.R. China

- ACL 2017: jointly find all relations relations of interest in a sentence by tagging arguments and combining them.



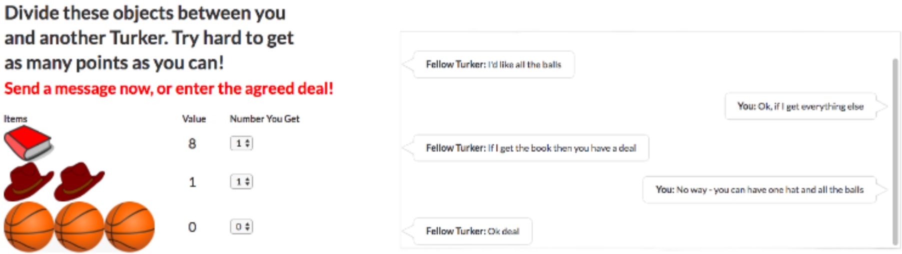
Task: Reinforcement Learning

- In **reinforcement learning**, the model (also called **agent**) needs to select a serious of actions, but only observes the outcome (**reward**) at the end.
- The goal is to predict actions that will maximize the outcome.

Deal or No Deal? End-to-End Learning for Negotiation Dialogues

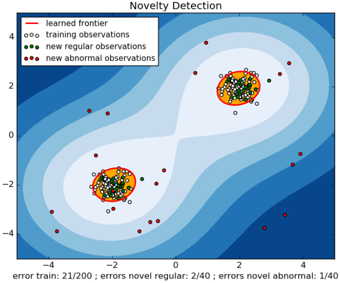
Mike Lewis[1], **Denis Yarats**[1], **Yann N. Dauphin**[1], **Devi Parikh**[2,1] and **Dhruv Batra**[2,1]
[1]Facebook AI Research [2]Georgia Institute of Technology

- EMNLP 2017: The computer negotiates with humans in natural language in order to maximize its points in a game.



Task: Anomaly Detection

- Detect atypical items or events.
- Common approach: Estimate density and identify items that have low probability.



- Examples:
  - Quality assurance
  - Detection of criminal activity
- Often items categorized as outliers are sent to humans for further scrutiny.

Task: Anomaly Detection

Using Automated Metaphor Identification to Aid in Detection and Prediction of First-Episode Schizophrenia

E. Darío Gutiérrez[1] Philip R. Corlett[2] Cheryl M. Corcoran[3] Guillermo A. Cecchi[1]

- ACL 2017: Schizophrenia patients can be detected by their non-standard use of mataphors, and more extreme sentiment expressions.

Supervised and Unsupervised Learning

- Unsupervised learning: Learn interesting properties, such as probability distribution $p(x)$
- Supervised learning: learn mapping from x to y, typically by estimating $p(y|x)$
- Supervised learning in an unsupervised way:

$$p(y|\mathbf{x}) = \frac{p(\mathbf{x}, y)}{\sum_{y'} p(\mathbf{x}, y')}$$

Outline

Performance Measures

"A computer program is said to learn [...] with respect to some [...] **performance measure** P, if its performance [...] **as measured by** P, improves [...]"

- Quantitative measure of algorithm performance.
- Task-specific.

Discrete vs. Continuous Loss Functions

- **Discrete** Loss Functions
  - Accuracy (how many samples were correctly labeled?)
  - Error Rate (1 - accuracy)
  - Precision / Recall
  - Accuracy may be inappropriate for skewed label distributions, where relevant category is rare

$$\text{F1-score} = \frac{2 \cdot \text{Prec} \cdot \text{Rec}}{\text{Prec} + \text{Rec}}$$

- Discrete loss functions cannot indicate   a wrong decision is
- They are not differentiable (hard to optimize)
- Often algorithms are optimized using a continuous loss (e.g. hinge loss) and evaluated using another loss (e.g. F1-Score).

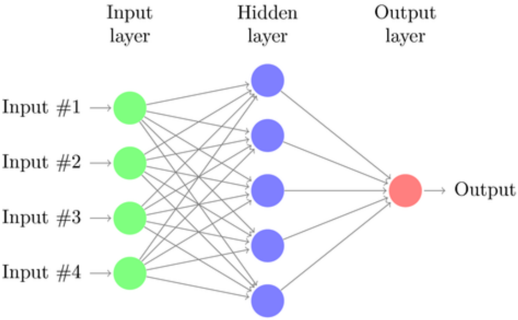
Examples for Continuous Loss Functions

- Squared error (regression): $(y - f(\mathbf{x}))^2$
- Hinge loss (classification):
  - $\max(0, 1 - f(\mathbf{x}) \cdot y)$
  - (assume that $y \in \{-1, 1\}$)
- ...
- These loss functions are differentiable. So we can use them for gradient descent (more on that later).

Outline

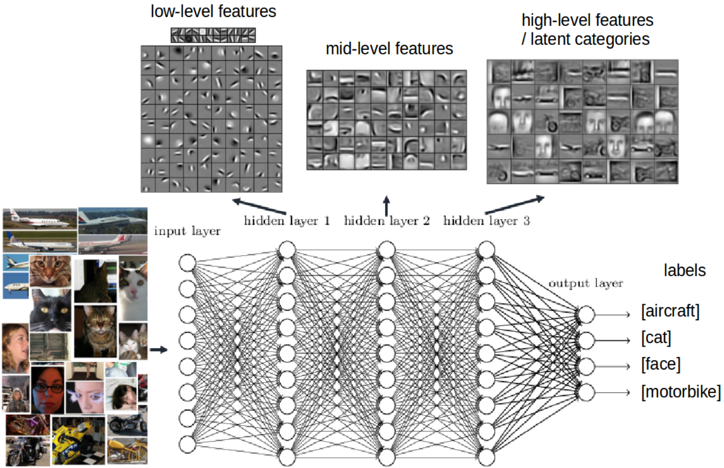
Deep Learning

- Learn complex functions, that are (recursively) composed of simpler functions.
- Many parameters have to be estimated.



Deep Learning

- Main Advantage: Feature learning
  - Models learn to capture *most essential* properties of data (according to some performance measure) as intermediate representations.
  - No need to hand-craft feature extraction algorithms



Neural Networks

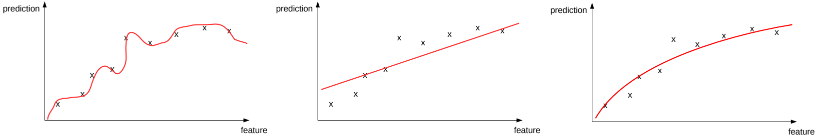
- First training methods for deep nonlinear NNs appeared in the 1960s (Ivakhnenko and others).
- Increasing interest in NN technology (again) since around 10 years ago (*"Neural Network Renaissance"*):
  Orders of magnitude more data and faster computers now.
- Many successes:
  - Image recognition and captioning
  - Speech regonition
  - NLP and Machine translation
  - Game playing (AlphaGO)
  - ...

Machine Learning

- Deep Learning builds on general Machine Learning concepts

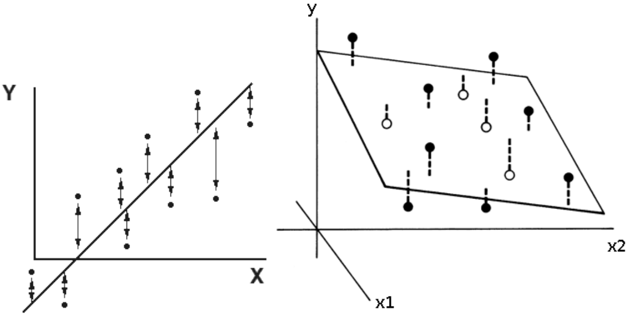
$$\text{argmin}_{\boldsymbol{\theta} \in \mathcal{H}} \sum_{i=1}^{m} \mathcal{L}(f(\mathbf{x}_i; \boldsymbol{\theta}), y_i)$$

- Fitting data vs. generalizing from data



Outline

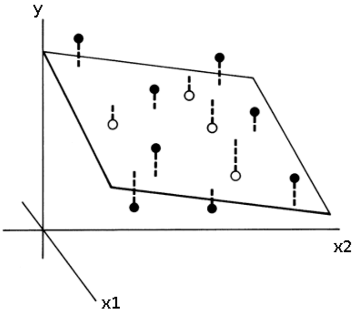
Linear Regression



- For one instance:
  - Input: vector $\mathbf{x} \in \mathbb{R}^n$
  - Output: scalar $y \in \mathbb{R}$
    (actual output: y; predicted output: $\hat{y}$)
  - Linear function

$$\hat{y} = \mathbf{w}^T \mathbf{x} = \sum_{j=1}^{n} w_j x_j$$

Linear Regression



- Linear function:

$$\hat{y} = \mathbf{w}^T\mathbf{x} = \sum_{j=1}^{n} w_j x_j$$

- Parameter vector $\mathbf{w} \in \mathbb{R}^n$
  Weight w_j decides if value of feature x_j increases or decreases prediction $\hat{y}$.

Linear Regression

- For the whole data set:
  - Use matrix $\mathbf{X}$ and vector $\mathbf{y}$ to stack instances on top of each other.
  - Typically first column contains all **1** for the intercept (bias, shift) term.

$$\mathbf{X} = \begin{bmatrix} 1 & x_{12} & x_{13} & \dots & x_{1n} \\ 1 & x_{22} & x_{23} & \dots & x_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{m2} & x_{m3} & \dots & x_{mn} \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

- For entire data set, predictions are stacked on top of each other:

$$\hat{\mathbf{y}} = \mathbf{X}\mathbf{w}$$

- Estimate parameters using $\mathbf{X}^{(train)}$ and $\mathbf{y}^{(train)}$.
- Make high-level decisions (which features...) using $\mathbf{X}^{(dev)}$ and $\mathbf{y}^{(dev)}$.
- Evaluate resulting model using $\mathbf{X}^{(test)}$ and $\mathbf{y}^{(test)}$.

Simple Example: Housing Prices

- Predict property prices (in 1K Euros) from just one feature: Square feet of property.

$$\mathbf{X} = \begin{bmatrix} 1 & 450 \\ 1 & 900 \\ 1 & 1350 \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} 730 \\ 1300 \\ 1700 \end{bmatrix}$$

- Prediction is:

$$\hat{\mathbf{y}} = \begin{bmatrix} w_1 + 450 w_2 \\ w_1 + 900 w_2 \\ w_1 + 1350 w_2 \end{bmatrix} = \begin{bmatrix} 1 & 450 \\ 1 & 900 \\ 1 & 1350 \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \mathbf{Xw}$$

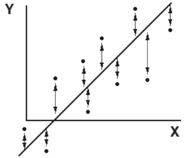
- $\mathbf{w}_1$ will contain costs incurred in any property acquisition
- $\mathbf{w}_2$ will contain remaining average price per square feet.
- Optimal parameters are for the above case:

$$\mathbf{w} = \begin{bmatrix} 273.3 \\ 1.08 \end{bmatrix} \quad \hat{\mathbf{y}} = \begin{bmatrix} 759.1 \\ 1245.1 \\ 1731.1 \end{bmatrix}$$

Linear Regression: Mean Squared Error

- Mean squared error of training (or test) data set is the sum of squared differences between the predictions and labels of all m instances.

$$MSE^{(train)} = \frac{1}{m} \sum_{i=1}^{m} (\hat{y}_i^{(train)} - y_i^{(train)})^2$$



- In matrix notation:

$$MSE^{(train)} = \frac{1}{m} ||\hat{\mathbf{y}}^{(train)} - \mathbf{y}^{(train)}||_2^2$$

$$= \frac{1}{m} ||\mathbf{X}^{(train)}\mathbf{w} - \mathbf{y}^{(train)}||_2^2$$

Outline

Summary

- Machine learning definition
  - Data
  - Task
  - Cost function
- Machine learning tasks
  - Classification
  - Regression
  - ...
- Deep Learning
  - many successes in recent years
  - feature learning instead of feature engineering
  - builds on general machine learning concepts
- Linear regression
  - Output depends linearly on input
  - Cost function: Mean squared error
- Next up: estimating the parameters