CS 165B: Machine Learning
Neural networks, Unsupervised learning

CS 165B
Nov 21, 2023

Instructor: Prof. Yu-Xiang Wang
Last time

• Feature expansion

• Kernel methods

• Punchline: There is a lot more mileage in your linear learners.
Recap: The idea of feature expansion --- “lifting” a feature vector to a higher-dimensional space.

<table>
<thead>
<tr>
<th></th>
<th>$\mathcal{X}$-space is $\mathbb{R}^d$</th>
<th>$\mathcal{Z}$-space is $\mathbb{R}^\tilde{d}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Features</td>
<td>$x = \begin{bmatrix} 1 \ x_1 \ \vdots \ x_d \end{bmatrix}$</td>
<td>$z = \Phi(x) = \begin{bmatrix} 1 \ \Phi_1(x) \ \vdots \ \Phi_{\tilde{d}}(x) \end{bmatrix}$</td>
</tr>
<tr>
<td>Data</td>
<td>$x_1, x_2, \ldots, x_N$</td>
<td>$z_1, z_2, \ldots, z_N$</td>
</tr>
<tr>
<td>Label</td>
<td>$y_1, y_2, \ldots, y_N$</td>
<td>$y_1, y_2, \ldots, y_N$</td>
</tr>
<tr>
<td>Weight</td>
<td>no weights</td>
<td>$\tilde{w} = \begin{bmatrix} w_0 \ w_1 \ \vdots \ w_{\tilde{d}} \end{bmatrix}$</td>
</tr>
</tbody>
</table>

Model $g(z) = \text{sign}(\tilde{w}^T \Phi(x))$
Recap: Examples of feature expansion

• Polynomial expansion

• Decision-stumps (recovers random forest and boosting)

• Radial basis feature expansion

• Generic RKHS with kernel $k$
Recap: The general idea is that in higher dimensions it is easier for the data to be linearly separable.
Recap: Illustration of how a kernel-SVM works as we adjust the kernel bandwidth

Kernel: $k(x, x') = e^{-\gamma \|x - x'\|^2}$  Feature map: $\phi(x) = e^{-\gamma \|x - \cdot\|^2}$
Checkpoint: Kernel methods

- They are essentially linear models --- linear in the expanded feature space

- Systematic way to tune the kernel-bandwidth, polynomial order, allows us to reduce “approximation error” and its tradeoff with “generalization error”.

- Drawbacks:
  - Need to specify the kernel
  - Computationally efficient but not scalable (O(n^3))!
Today

• Feature learning and neural networks.

• Unsupervised learning
Neural networks: Example: AlexNet (2012)

Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network’s input is 150,528-dimensional, and the number of neurons in the network’s remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

**Imagenet classification with deep convolutional neural networks**

A Krizhevsky, I Sutskever... - Advances in neural ..., 2012 - proceedings.neurips.cc

... a large, deep convolutional neural network to classify the 1.2 million high-resolution images in the ImageNet ... The neural network, which has 60 million parameters and 650,000 neurons, ...
LeNet (1998)
Rumelhart, Hinton, Williams (1986)

- One layer of a feedforward neural networks

Learning representations by back-propagating errors
DE Rumelhart, GE Hinton, RJ Williams - nature, 1986 - nature.com

... their states are completely determined by the input vector; they do not learn representations.)

The learning procedure must decide under what circumstances the hidden units should be ...
It goes back even further…

• 1943 Pitts and McCulloch: Perceptron model to mimic the brain
• 1956: Rosenblatt’s Perceptron Implementation
• 1960s:
  • Ivakhnenko and Lapa: Multi-layer Perceptron (going deeper)
  • Dreyfus: Backpropagation for training (not yet the same as SGD)
  • Amari: Use SGD for training MLPs (separating non-linearly separable patterns)
• 1970s:
  • Fukushima: Convolutional Neural Networks for images
• 1982:
  • Werbos: Modern day backpropagation / SGD
From kernels to neural networks

\[ x \rightarrow \phi(x) \rightarrow S(x) = w^T \phi(x) + b \]

Feature map

Linear layer
Two-layer neural networks

- Linear neural network: \( S(x) = w_2^T (W_1 x + b_1) + b_2 \)
  - Still a linear model at the end of the day, so let’s add a nonlinearity \( \sigma \)!

- Two-layer MLP: \( S(x) = w_2^T \sigma(W_1 x + b_1) + b_2 \)
  - Linear model w.r.t. to a learnable feature map

- RBF-kernel: \( S(x) = w_2^T \exp(i (W_1 x + b_1)) + b_2 \)
  - Kernels are infinite width neural networks with fixed weights
  - By Bochner’s theorem, see, e.g., [Rahimi and Recht, 2007]
Results of fitting MLPs on our three examples

```python
from sklearn.neural_network import MLPClassifier
hidden = 100
# Neural network classifier
nn_clf = MLPClassifier(hidden_layer_sizes=(hidden,), activation='relu', max_iter=1000)
```
Modern neural networks are very complicated

Figure 2. Residual learning: a building block.

Deep residual learning for image recognition
K He, X Zhang, S Ren, J Sun - ... and pattern recognition, 2016 - openaccess.thecvf.com

... Deeper neural networks are more difficult to train. We present a residual learning framework to ease the training of networks that are substantially deeper than those used previously. ...
Modern neural networks are very complicated

**Attention is all you need**

A Vaswani, N Shazeer, N Parmar… - Advances in neural …, 2017 - proceedings.neurips.cc

... to attend to all positions in the decoder up to and including that position. **We need** to prevent ...

We implement this inside of scaled dot-product **attention** by masking out (setting to \( -\infty \)) ...

☆ Save  💬 Cite  Cited by 97503  Related articles  All 62 versions  Import into BibTeX  ✯
Solution to this? Brute-force computation with autograd and GPUs

• **Autograd**: basically chain rules, can be automated.
  • Design networks such that every block is differentiable.

• **Faster Computation**:
  • Well-packaged Deep Learning Framework: Write Python wrapper code but running C++ underneath
  • Parallel computing: Numerical linear algebra and GPUs, scientific computing, supercomputing centers.
  • Distributed computing: Cloud computing, Map-Reduce, federated learning

• Popular tools (there are many more of these):

![PyTorch](image1)
![tensorflow](image2)
![JAX: Autograd and XLA](image3)
Demo of autograd and implementation of a neural network

• Implementing a soft-max regression from scratch.

• Implementing a neural network from scratch.

HW4 Q1 gives you an idea of how to use “autograd”.

Learn more CS190I or 290A “deep learning” that are offered from time to time.
Checkpoint: Neural Networks

• Kernels are infinite width neural networks with fixed weights until the last layer, i.e., fixed feature expansion.

• Neural networks learn feature expansion end-to-end.

• Learn autograd and deep learning frameworks, e.g., pytorch.
  • You can create your neural network and start training it within minutes.
Today

• Feature learning and neural networks.

• Unsupervised learning
Unsupervised learning

• Input space: $\mathcal{X}$ Images, videos, text, graphs, proteins, programs, etc…

• Output space: None.

• Hypothesis space: $\mathcal{H}$

• Each hypothesis $h$ is a particular way to summarize the data

• Loss function $\ell : \mathcal{H} \times \mathcal{X} \rightarrow \mathbb{R}$

• Goal of unsupervised learning: Discover simple hypothesis that captures interesting aspects of the data distribution.
  • Often achieved by minimizing the expected loss (i.e., Risk).
The goal of unsupervised learning is to learn structures in data without human annotations.

• Discussion: What kind of structures can you see?
What kind of structures can you see?
What kind of structures can you see?
What kind of structures can you see?
What kind of structures can you see?
What kind of structures can you see?
Two broad categories of unsupervised learning

(1) Clustering

(2) Dimension reduction

- Clustering aims at finding a partition of the data that makes sense.

- Dimension reduction aims at identifying a more compact representation of data
Applications: Indexing text corpus

<table>
<thead>
<tr>
<th>“Arts”</th>
<th>“Budgets”</th>
<th>“Children”</th>
<th>“Education”</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEW</td>
<td>MILLION</td>
<td>CHILDREN</td>
<td>SCHOOL</td>
</tr>
<tr>
<td>FILM</td>
<td>TAX</td>
<td>WOMEN</td>
<td>STUDENTS</td>
</tr>
<tr>
<td>SHOW</td>
<td>PROGRAM</td>
<td>PEOPLE</td>
<td>SCHOOLS</td>
</tr>
<tr>
<td>MUSIC</td>
<td>BUDGET</td>
<td>CHILD</td>
<td>EDUCATION</td>
</tr>
<tr>
<td>MOVIE</td>
<td>BILLION</td>
<td>YEARS</td>
<td>TEACHERS</td>
</tr>
<tr>
<td>PLAY</td>
<td>FEDERAL</td>
<td>FAMILIES</td>
<td>HIGH</td>
</tr>
<tr>
<td>MUSICAL</td>
<td>YEAR</td>
<td>WORK</td>
<td>PUBLIC</td>
</tr>
<tr>
<td>BEST</td>
<td>SPENDING</td>
<td>PARENTS</td>
<td>TEACHER</td>
</tr>
<tr>
<td>ACTOR</td>
<td>NEW</td>
<td>SAYS</td>
<td>BENNETT</td>
</tr>
<tr>
<td>FIRST</td>
<td>STATE</td>
<td>FAMILY</td>
<td>MANIGAT</td>
</tr>
<tr>
<td>YORK</td>
<td>PLAN</td>
<td>WELFARE</td>
<td>NAMPHY</td>
</tr>
<tr>
<td>OPERA</td>
<td>MONEY</td>
<td>MEN</td>
<td>STATE</td>
</tr>
<tr>
<td>THEATER</td>
<td>PROGRAMS</td>
<td>PERCENT</td>
<td>PRESIDENT</td>
</tr>
<tr>
<td>ACTRESS</td>
<td>GOVERNMENT</td>
<td>CARE</td>
<td>ELEMENTARY</td>
</tr>
<tr>
<td>LOVE</td>
<td>CONGRESS</td>
<td>LIFE</td>
<td>HAITI</td>
</tr>
</tbody>
</table>

The William Randolph Hearst Foundation will give $1.25 million to Lincoln Center, Metropolitan Opera Co., New York Philharmonic and Juilliard School. “Our board felt that we had a real opportunity to make a mark on the future of the performing arts with these grants an act every bit as important as our traditional areas of support in health, medical research, education and the social services,” Hearst Foundation President Randolph A. Hearst said Monday in announcing the grants. Lincoln Center’s share will be $200,000 for its new building, which will house young artists and provide new public facilities. The Metropolitan Opera Co. and New York Philharmonic will receive $400,000 each. The Juilliard School, where music and the performing arts are taught, will get $250,000. The Hearst Foundation, a leading supporter of the Lincoln Center Consolidated Corporate Fund, will make its usual annual $100,000 donation, too.
Application: Motion segmentation and subspace clustering

- Turns out that the tracked point trajectories of each rigid moving body captured on a video fall into a 4-dimensional subspace.
- To cluster these points, it suffices to find the subspace membership.
Applications: learn useful vector space representation of language

• So you can do algebra on them..

Source: https://towardsdatascience.com/creating-word-embeddings-coding-the-word2vec-algorithm-in-python-using-deep-learning-b337d0ba17a8
Application: Image / video compression

Source: http://preservationtutorial.library.cornell.edu/intro/intro-07.html
How do you learn the structure you see?

- Come up with a loss function to minimize?
- Come up a probabilistic model that generates the data?
The problem of k-means clustering

\[ \arg\min_S \sum_{i=1}^{k} \sum_{x \in S_i} \| x - \mu_i \|^2 \]

• Where \( S = \{S_1, S_2, ..., S_k\} \) is a partition of the dataset \( (x_1, x_2, ..., x_n) \),

• And \( \mu_i = \frac{1}{|S_i|} \sum_{x \in S_i} x \), is called a centroid of \( S_i \)

• This optimization problem is \textit{NP-complete}, but we have very practical algorithms. \textit{Kmeans++} is guaranteed to find a solution within a factor of \( O(\log k) \).
K-means clustering with Lloyd’s algorithm

**Algorithm** $\text{KMeans}(D, K)$ – K-means clustering using Euclidean distance $\text{Dis}_2$.

**Input** : data $D \subseteq \mathbb{R}^d$; number of clusters $K \in \mathbb{N}$.

**Output** : $K$ cluster means $\mu_1, \ldots, \mu_K \in \mathbb{R}^d$.

**randomly initialise** $K$ vectors $\mu_1, \ldots, \mu_K \in \mathbb{R}^d$;

**repeat**

assign each $x \in D$ to $\arg\min_j \text{Dis}_2(x, \mu_j)$; \hspace{1cm} 1-Nearest neighbor assignment

for $j = 1$ to $K$ do

$D_j \leftarrow \{x \in D | x \text{ assigned to cluster } j\}$;

$\mu_j = \frac{1}{|D_j|} \sum_{x \in D_j} x$; \hspace{1cm} Re-compute the cluster mean

end

**until** no change in $\mu_1, \ldots, \mu_K$;

**return** $\mu_1, \ldots, \mu_K$;
No change – finished!
Nice demo of k-means variants for you to play with

https://www.naftaliharris.com/blog/visualizing-k-means-clustering/
K-means on our previous examples
Gaussian mixture models

• Assume the data is generated from a mixture of Gaussian distribution

• Data generating process
Fitting Mixture of Gaussian model
Discussion: what can we do for this?

• Hint: we have learned some useful tricks last week.
Checkpoint: Unsupervised Learning

• K-means problem
  • Lloyd’s algorithm for solving k-means problem
  • Which distance function to use?
  • How many cluster centers (centroids) to choose?
  • How to initialize the centroids?
  • **Implement the Lloyd’s algorithm in HW4 Q2**

• Gaussian Mixture models
  • A probabilistic model for clustering.
  • Soft assignment of observed data points.
  • When the covariance matrix is small and isotropic it is very similar to k-means.
  • **What’s the difference from Gaussian Naïve Bayes model?**
Next lecture

• More on unsupervised learning

• Dimension reduction