COMPUTER SCIENCE
INDIANA UNIVERSITY
School of Informatics and Computing
Bloomington

Neural networks

Reminders/Comments

- Thought questions due Wednesday
- Class mini project:
  - pick a dataset from kaggle.com (competition website)
  - project template given for report in canvas
  - additional benefits: good for building a machine learning portfolio and you might be able to solve a real problem
  - suggestion: submit your predictions after your project to see how you do; do not submit numerous times to get high on the public leaderboard as this is only calculated on a small subset of the test

Thought question

- How do you minimize the square error for equations that do not pass the vertical line test? Since the function is non-injective, how could you know what output to compare to?
  - Well-defined functions can only have one value given an input
  - For loss to be a well-defined function, Loss(f(x_i)) must return same value; but due to uncertainty, could see different y for same input x
    - e.g. Loss(f(x_i)) = ll f(x_i) - y_i ll
  - The true error, however, E[L(f(X), Y)] is a well-defined function
  - By minimizing the error summed over many samples, we are approximating this true error that we wish to minimize

Thought question

- I’m having a hard time seeing where SVD exactly fits into machine learning. We used it at one point to compute a pseudo-inverse to solve the optimization problem when features are linearly dependent. Is this its primary use in machine learning? Also, what are the usual considerations ones needs to make when deciding to throw out a very small singular value?

- SVD enables more simple characterization of a linear system
- Since we are learning linear models in many situations, and learning matrices as weights, it generally allows simpler descriptions of these
- Example: SVD is used for matrix factorization, to learn new lower-dimensional representations (e.g. PCA)
- Example: A trace-norm or nuclear-norm regularizer is a common choice for many problems, and requires an SVD to be computes
- Selecting threshold: in “good” systems, often a clear change, where singular value magnitude suddenly drops; good place to cut them off

Thought question

- How is the performance of regression techniques for discrete(categorical) data? Can we use regression prediction model and round the result to the closest number and use it for prediction of categorical data?
  - For symmetric, binary classification we have seen linear regression can do reasonably well (and does so in Assignment 3)
  - How about three values? Say we have prediction task {0, 1, 2}. Let's map this to -1, 0 and 1
  - How about four values? Say we have prediction task {0, 1, 2, 3}. Could map this to -2, -1, 1 and 2

Representation learning

- Generalized linear models enabled many p(y l x) distributions
  - Underneath, still learning a linear representation for E[y l x], which may not have enough representation capacity

- Approach we discussed earlier: augment current features x using polynomials

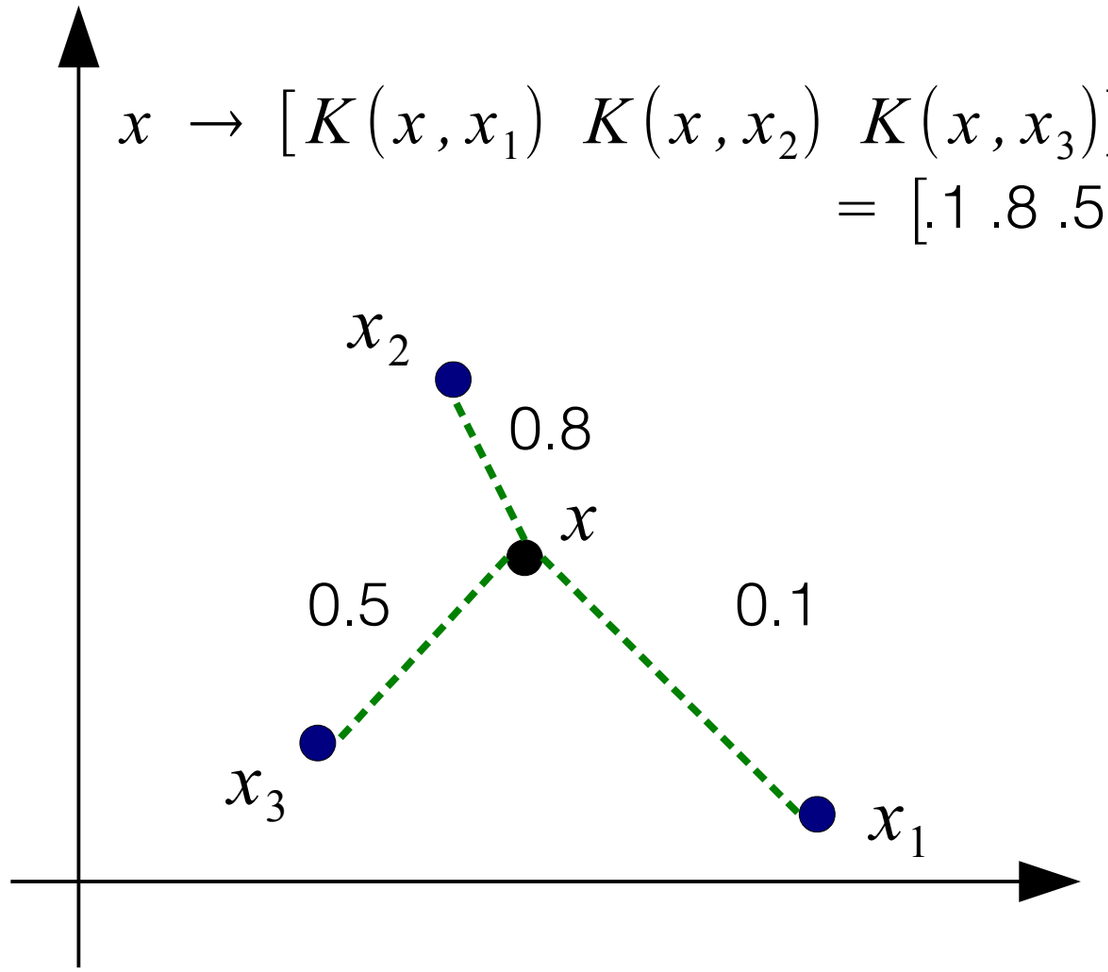
- There are many strategies to augmenting x
  - fixed representations, like polynomials, wavelets
  - learned representations, like neural networks and matrix factorization

Gaussian kernel / Gaussian radial basis function

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{-\|\mathbf{x} - \mathbf{x}'\|_2^2}{\sigma^2}\right)$$

$$f(\mathbf{x}) = \sum_{i=1}^{k} w_i k(\mathbf{x}, \mathbf{x}_i)$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \rightarrow \begin{bmatrix} k(\mathbf{x}, \mathbf{x}_1) \\ \vdots \\ k(\mathbf{x}, \mathbf{x}_k) \end{bmatrix}$$



Other fixed representations

- Fourier basis
- Wavelets

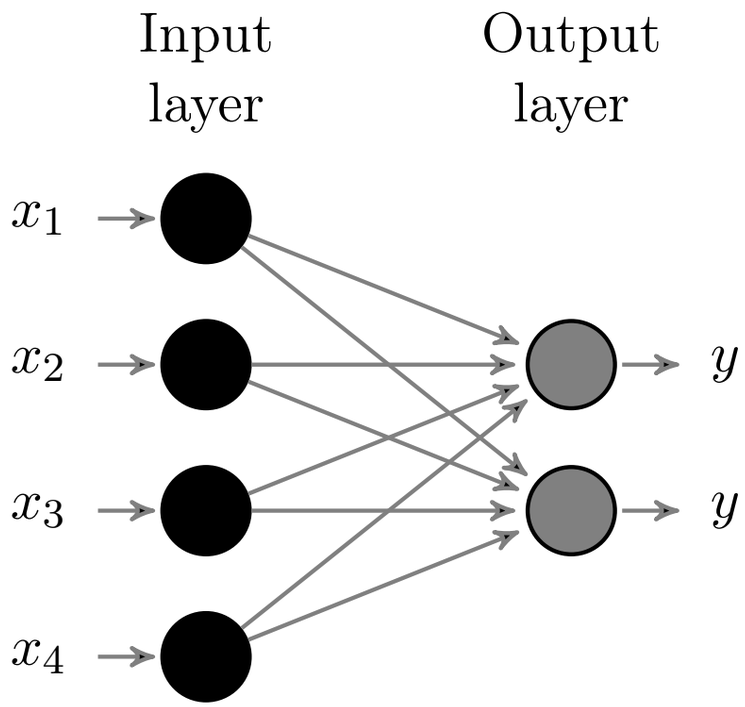
- Tile coding (also called CMAC for cerebellar model articulation controller)

- For many of these fixed representations, try to cover the space with the transformation

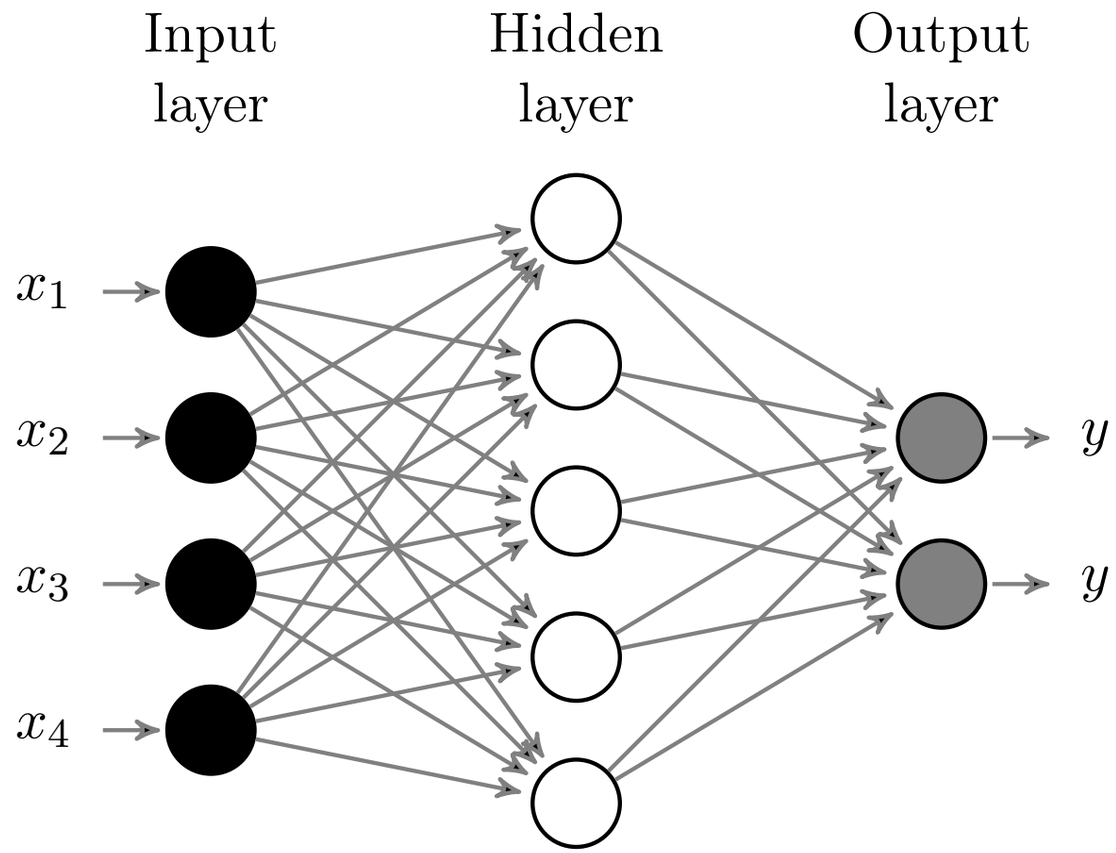
Learning representations

- One way to learn a representation is to learn the parameters to the previously mentioned fixed representations
  - e.g. could learn bandwidth sigma to Gaussian RBF
  - e.g., learning centers for RBFs and kernels
- There are, however, strategies for learning a representation more from scratch; we will focus on two main ones
  - Neural networks
  - Matrix factorization techniques (regularized factor models)

Generalized linear model vs. neural network

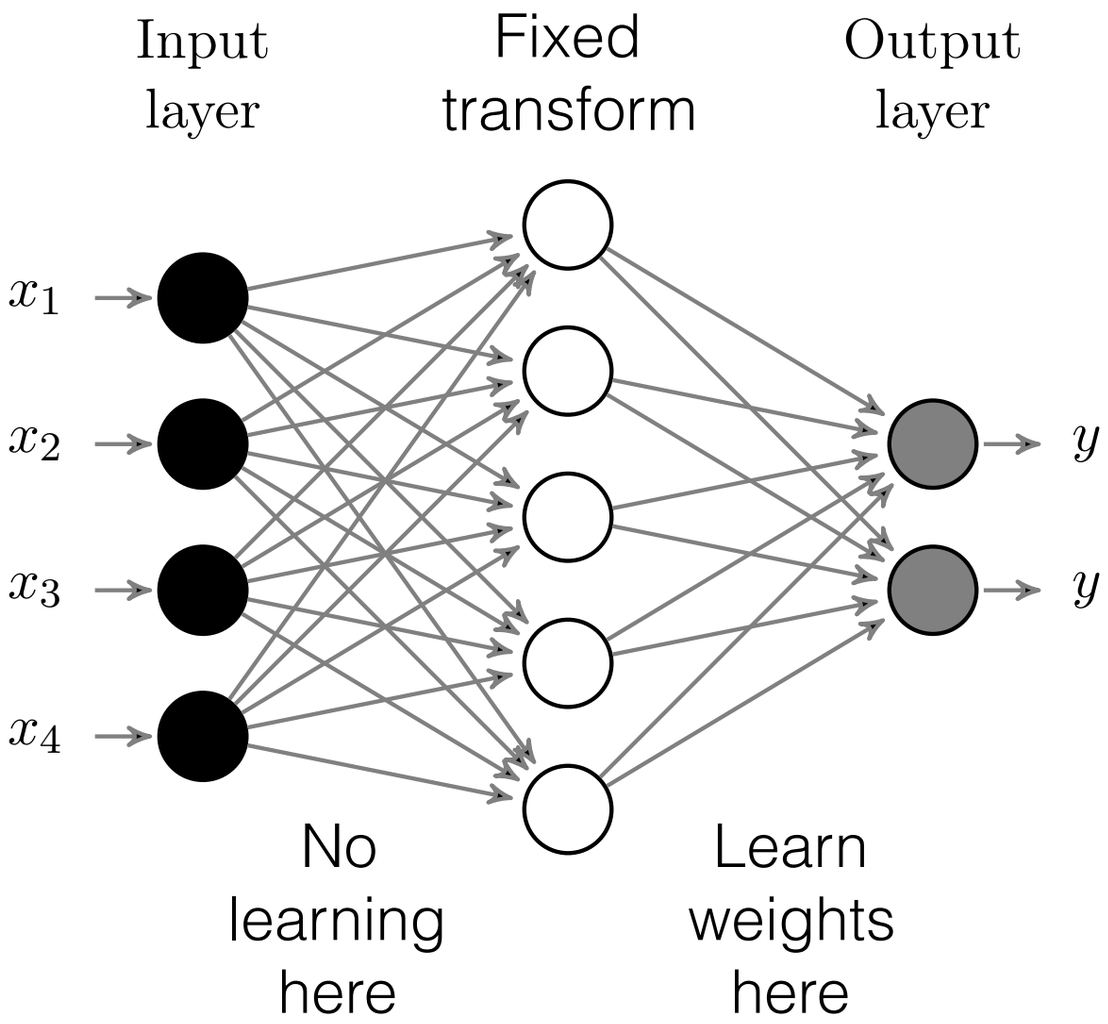


GLM
(e.g. logistic regression)

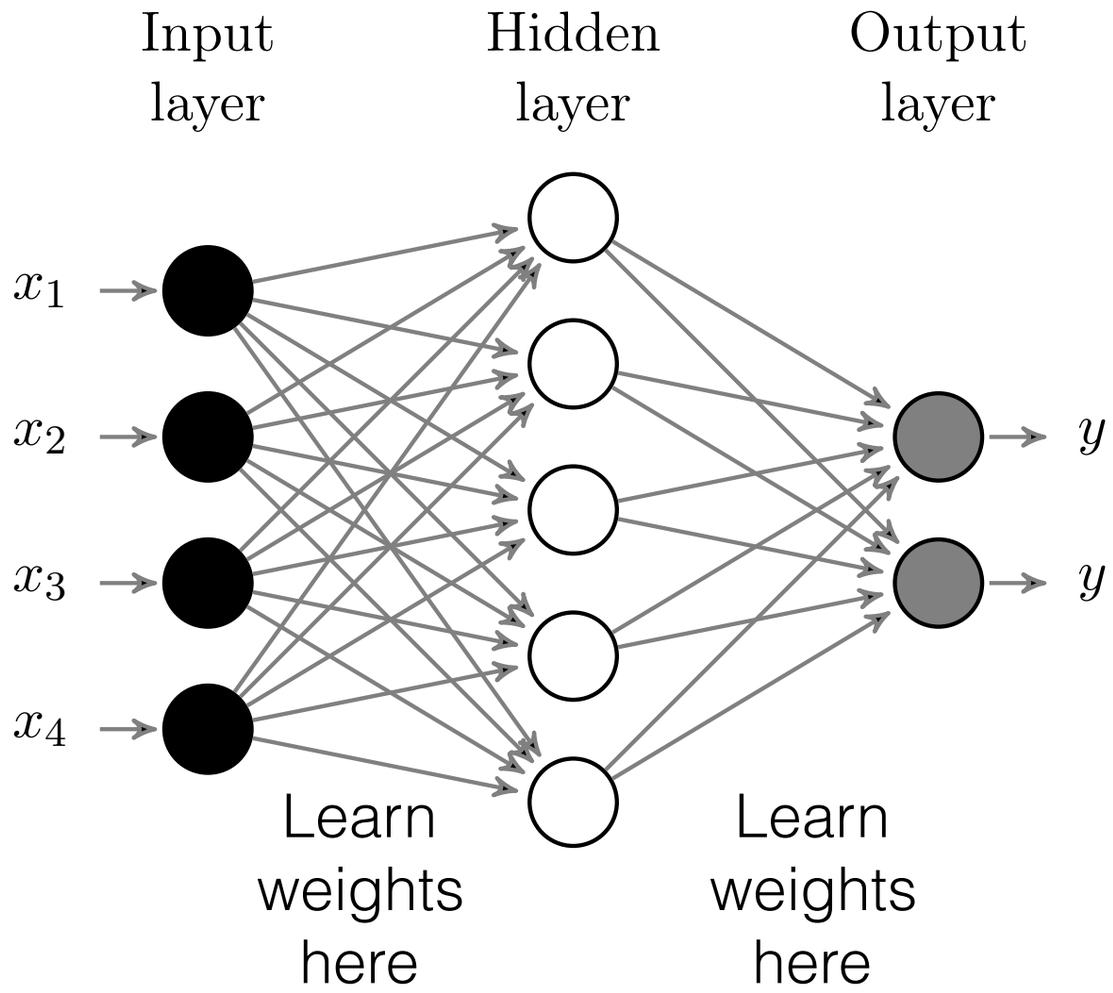


Two-layer neural network

GLM with fixed representation vs. neural network



GLM with augmented fix representation



Two-layer neural network

Whiteboard

- Learning parameters for a neural networks
  - gradient descent rule (i.e., backpropagation)

- Next time:
  - examples of backpropagation with other transfers
  - matrix factorization for representation learning