Key Concepts in Deep Learning

Tripp Deep Learning F23

UNIVERSITY OF WATERLOO

TODAY'S GOAL

By the end of the class, you should be familiar with twelve of the most basic concepts in deep learning.

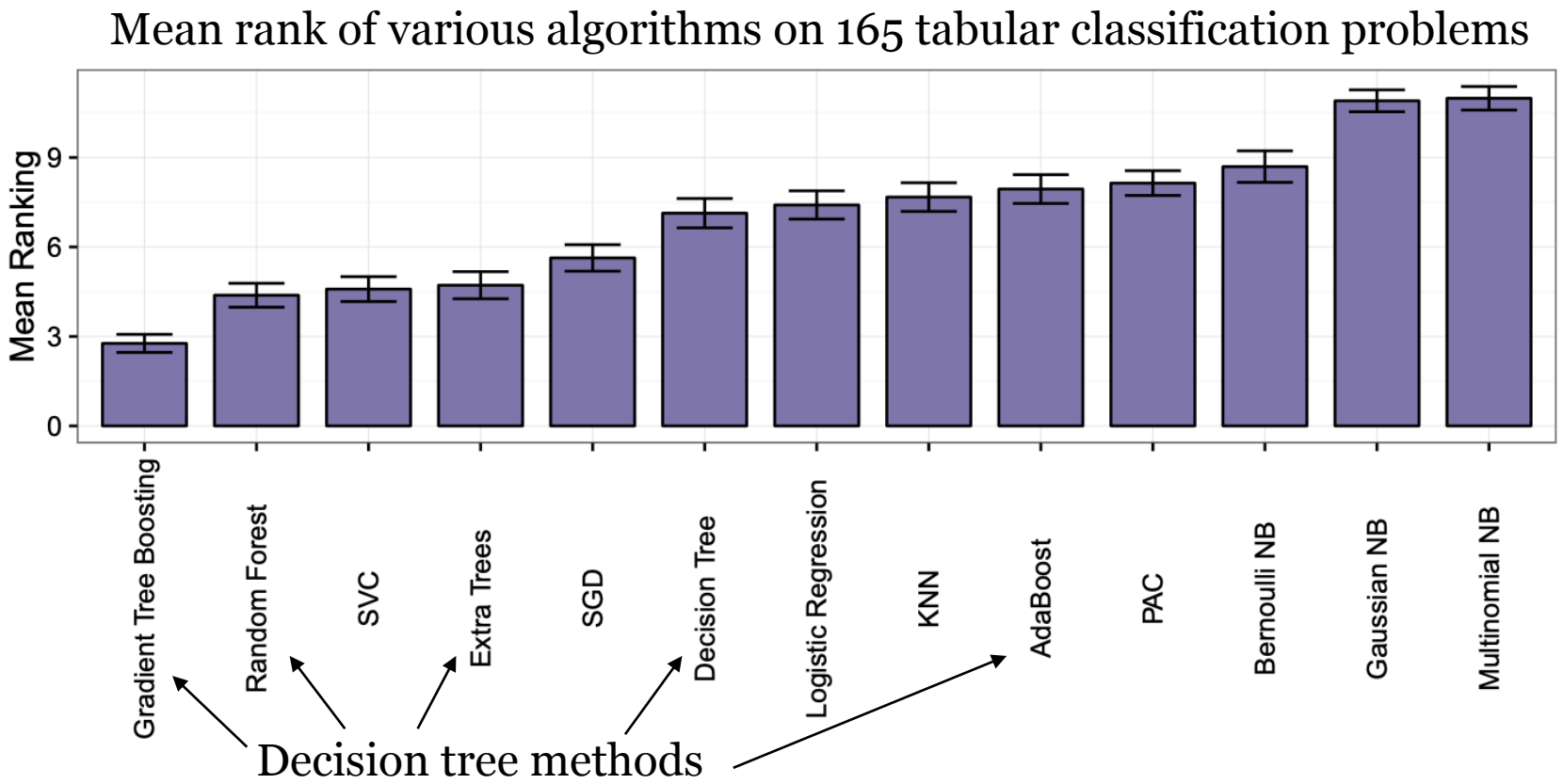
THE BEST PERFORMING MACHINE LEARNING SYSTEMS ARE USUALLY BASED ON DEEP NETWORKS OR DECISION TREES

These approaches tend to work well in different situations

- Deep networks work well when the input is complex and high-dimensional, such as an image or natural-language text
  - Example task: language translation
- Decision tree-based methods work well when the input is tabular
  - Example task: predicting house prices from year, city, # bedrooms, etc.

Decision tree methods

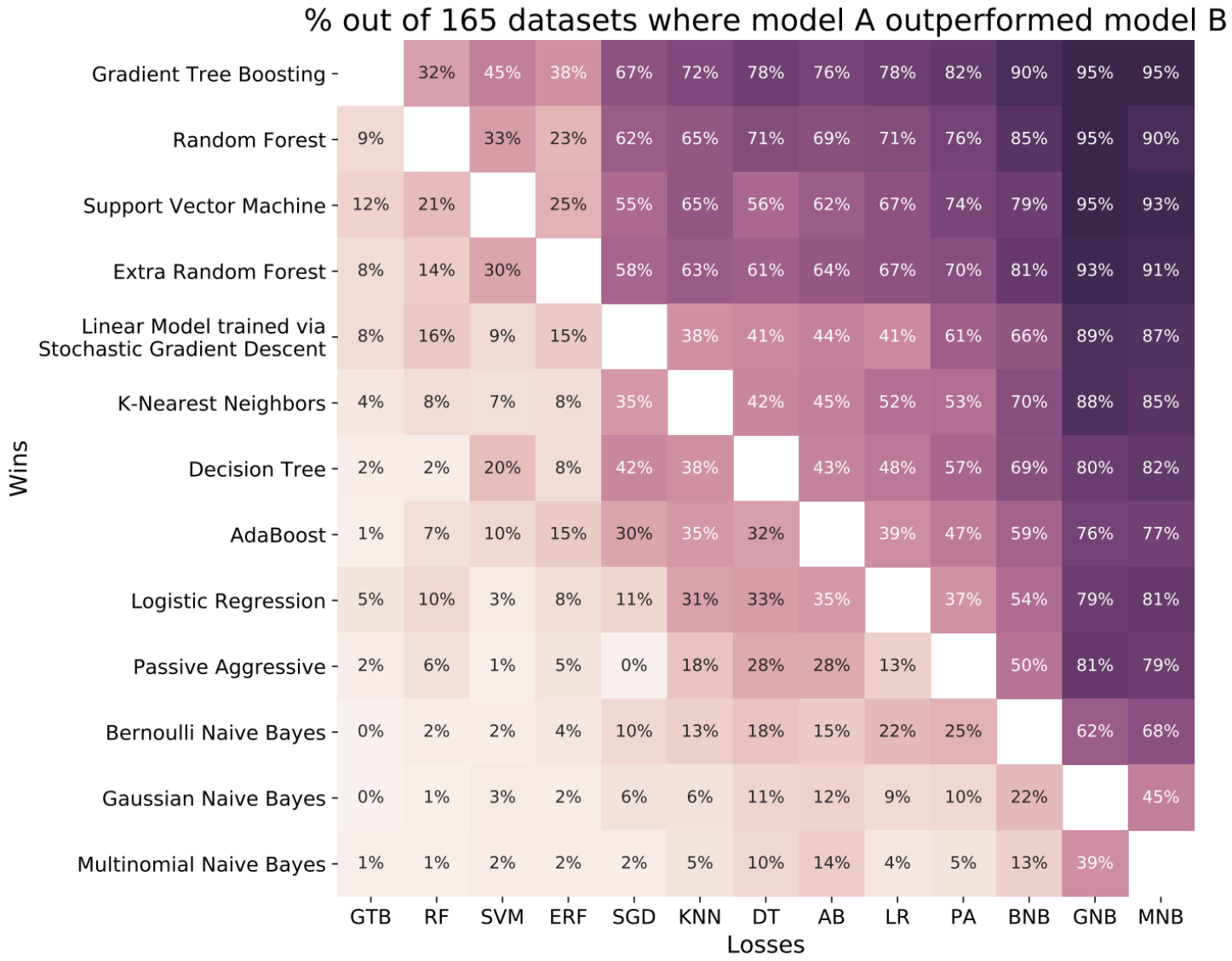
- Normally many trees are combined (e.g., random forests, gradient tree boosting)
- These methods are competitive on tabular data



Olson et al., 2018, Data-driven advice for applying machine learning to bioinformatics problems

UNIVERSITY OF WATERLOO

No single method works best for every problem.



Olson et al., 2018, Data-driven advice for applying machine learning to bioinformatics problems

UNIVERSITY OF WATERLOO

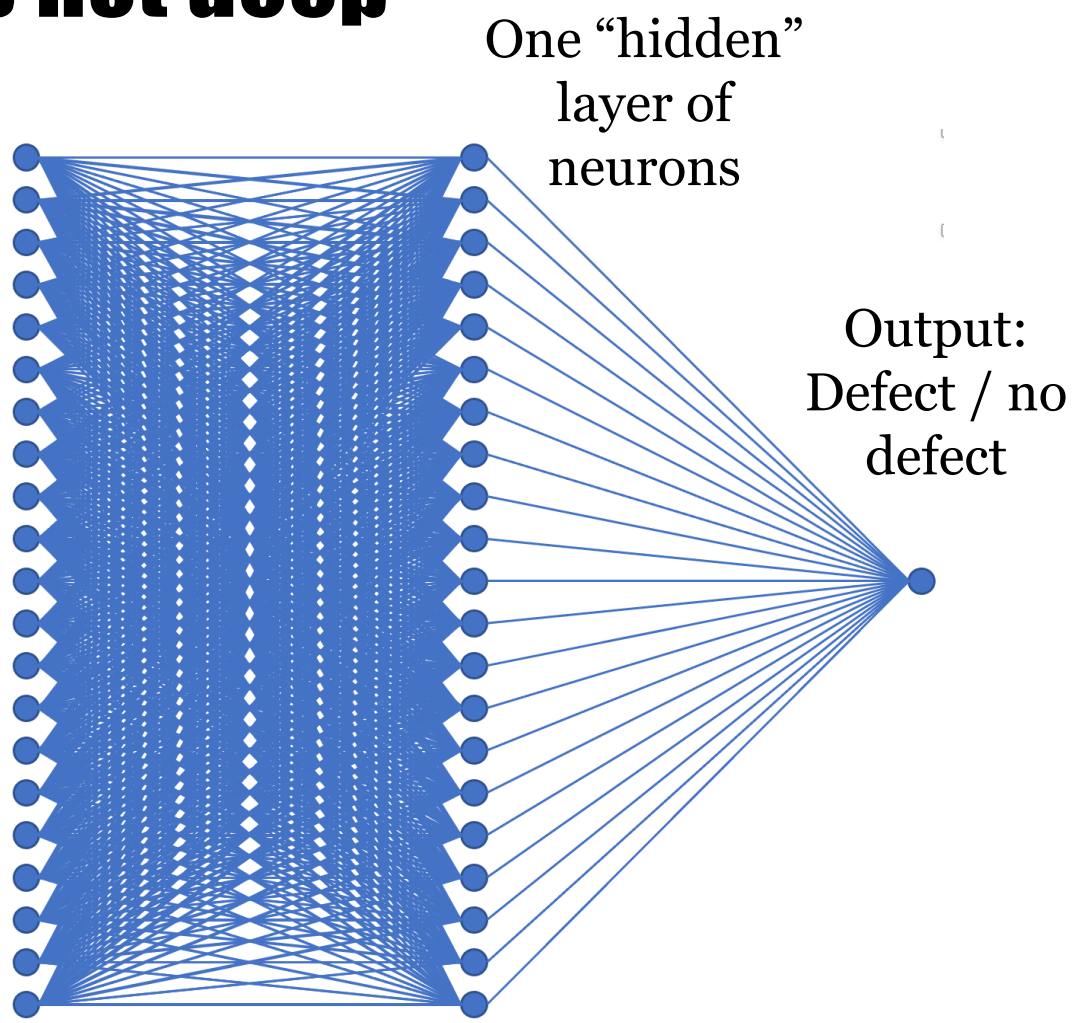
DEEP NETWORKS ARE ARTIFICIAL NEURAL NETWORKS THAT ARE LARGE AND OFTEN COMPLEX

Deep vs. non-deep networks

- A “deep network” is an artificial neural network that has more than a few layers
- There is no crisp boundary between deep and non-deep networks, but there are clear examples of each

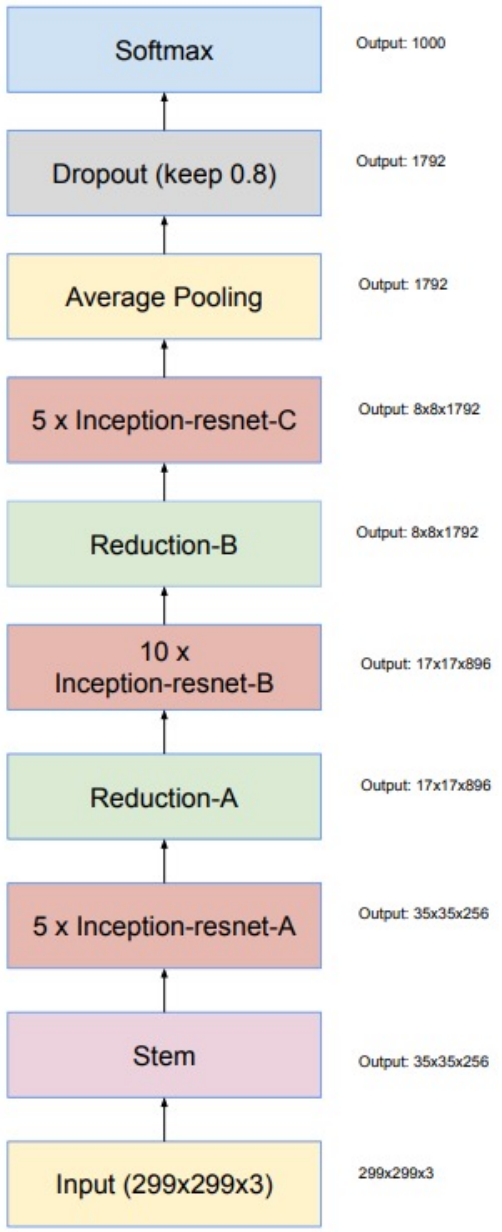
Example of a neural network that is not deep

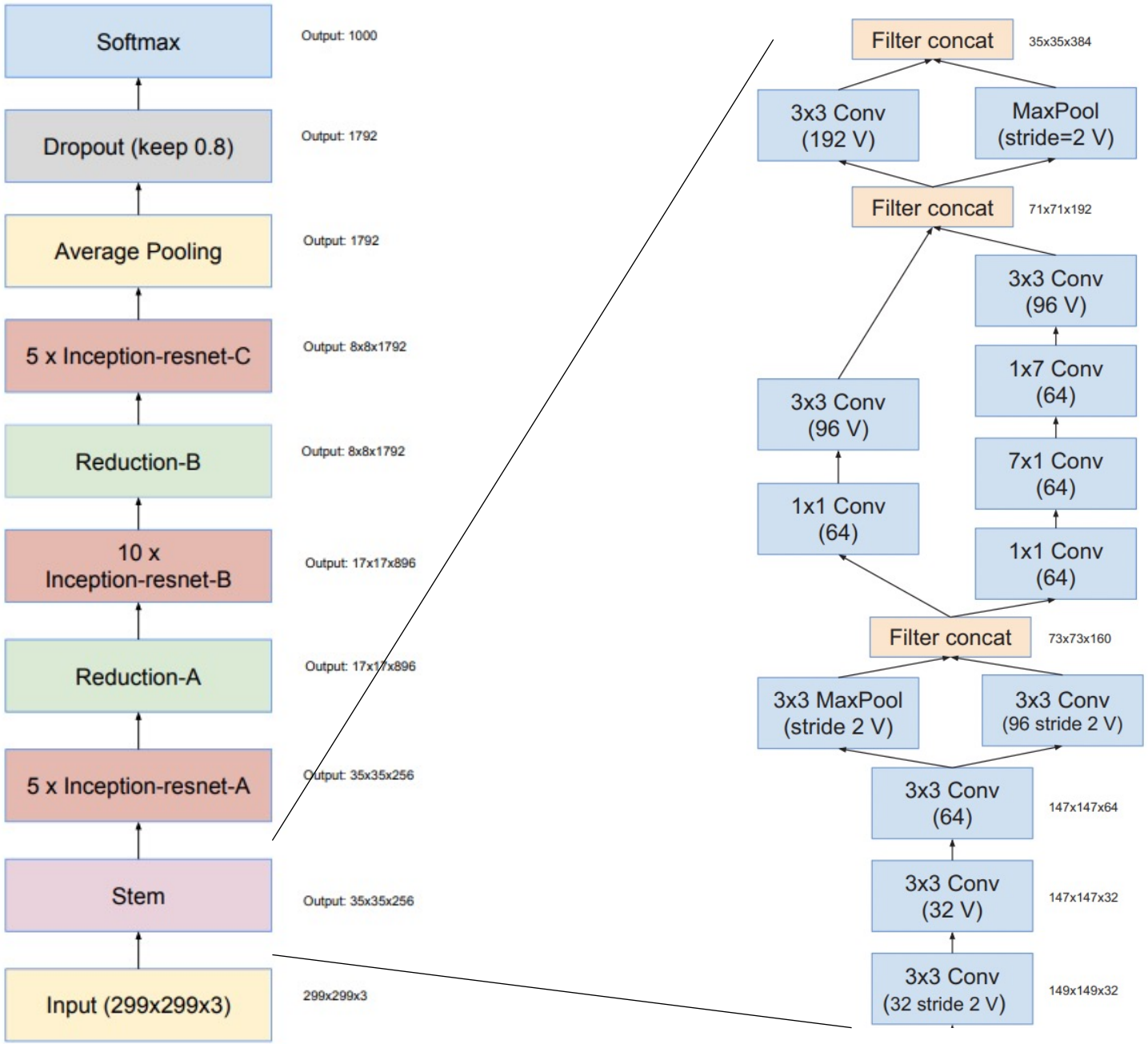
- Sebald, A. V. (1989). Use of neural networks for detection of artifacts in arterial pressure waveforms. In *Images of the Twenty-First Century. Proceedings of the Annual International Engineering in Medicine and Biology Society,* (pp. 2034-2035). IEEE.
- Two layers (plus input)
- 484 parameters (weights & biases)



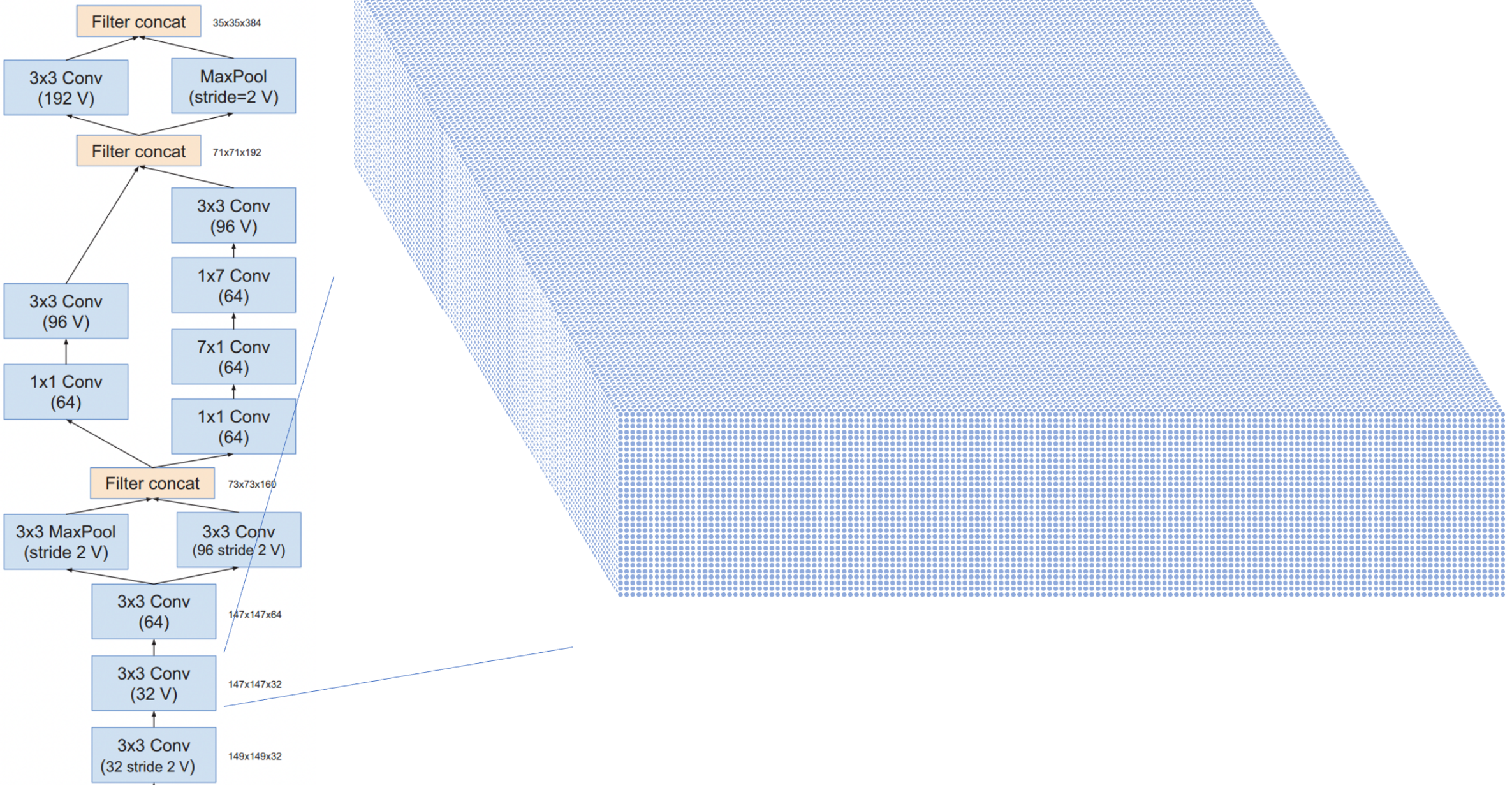
Example of a neural network that is deep

- Inception-v4.
- Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A. (2017, February). Inception-v4, inception-resnet and the impact of residual connections on learning. In Thirty-first AAAI conference on artificial intelligence.
- Many layers, e.g., “stem” box contains 13 layers





UNIVERSITY OF WATERLOO


Filter concat
35x35x384
3x3 Conv
(192 V)
MaxPool
(stride=2 V)
Filter concat
71x71x192
3x3 Conv
(96 V)
1x7 Conv
(64)
3x3 Conv
(96 V)
7x1 Conv
(64)
1x1 Conv
(64)
1x1 Conv
(64)
Filter concat
73x73x160
3x3 MaxPool
(stride 2 V)
3x3 Conv
(96 stride 2 V)
3x3 Conv
(64)
147x147x64
3x3 Conv
(32 V)
147x147x32
3x3 Conv
(32 stride 2 V)
149x149x32

The benefits of depth and complexity

- Sometimes the output we want is not closely related to any of the inputs
- Example: Suppose we wanted to detect melanoma in an image of a skin lesion
  - A particular pixel (say, pixel number 213) could have any colour that can be seen on skin
  - A single network layer can only calculate a simple abstraction, such as whether a particular pixel is at the edge of a transition between two colours
  - More sophisticated, high-level abstractions (e.g., irregularity of a border) provide a better basis for classification
- The many layers of a deep network can learn high-level abstractions, ones that are less and less related to pixel colours but more and more relevant to the task
- This can occur within as few as half a dozen layers, but making a network deeper often makes it more accurate

IMPORTANT ELEMENTS OF A DEEP LEARNING SYSTEM INCLUDE INPUTS, OUTPUTS, ARCHITECTURE, LOSS, PARAMETERS, OPTIMIZATION ALGORITHM, AND HYPERPARAMETERS

Important elements

- Inputs: The information that is fed into the network.
  - Can be signals, images, text, etc.
  - It is always converted into numeric form; e.g., words of text must be converted into vectors that stand for the words
  - Of course, the information must be sufficient for the network to produce the desired output
- Outputs: The inference that the network produces given the input
  - Can be a category label, a continuous value, an image, a sentence, etc.
  - Network outputs are also in numeric form; e.g., to identify a category, a network should output a vector with a large number at the index that belongs to that category and small numbers elsewhere

Important elements

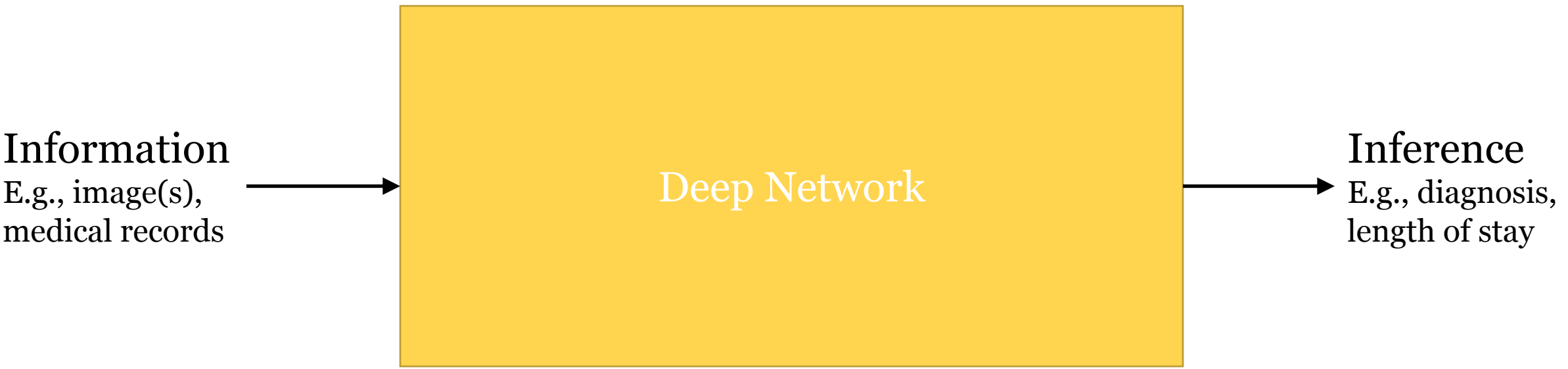
- Architecture: Structural properties of the network
  - E.g., the number of layers, how many neurons are in each layer, what kinds of layers they are (such as fully connected, convolutional), which layers are connected
- Loss (or cost)
  - A function that quantifies how bad the network is at producing correct outputs
  - The loss is reduced during training
- Learned parameters (or just “parameters”)
  - Typically, these include the weights and biases that belong to each neuron
  - Some kinds of layer have other parameters (e.g., batch normalization layers have a scale parameter)

Important elements

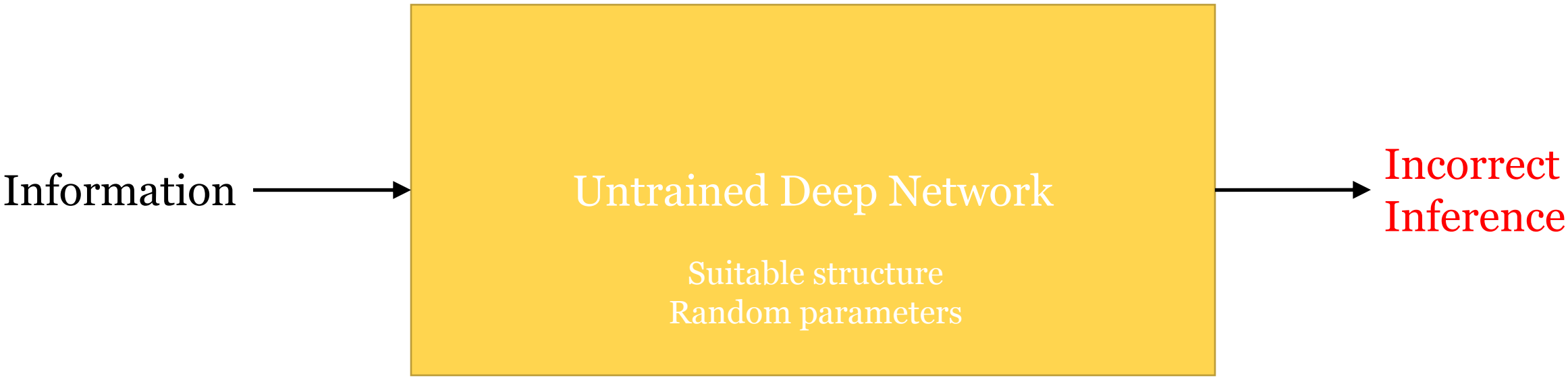
- Optimization algorithm: The algorithm for changing the parameters in order to reduce the loss
  - E.g., stochastic gradient descent
- Hyperparameters
  - These network properties are fixed during training, but the best values are unknown in advance
  - E.g., learning rate, architectural parameters such as numbers of neurons in each layer, loss parameters
  - Trying different hyperparameter combinations may reveal values that perform better

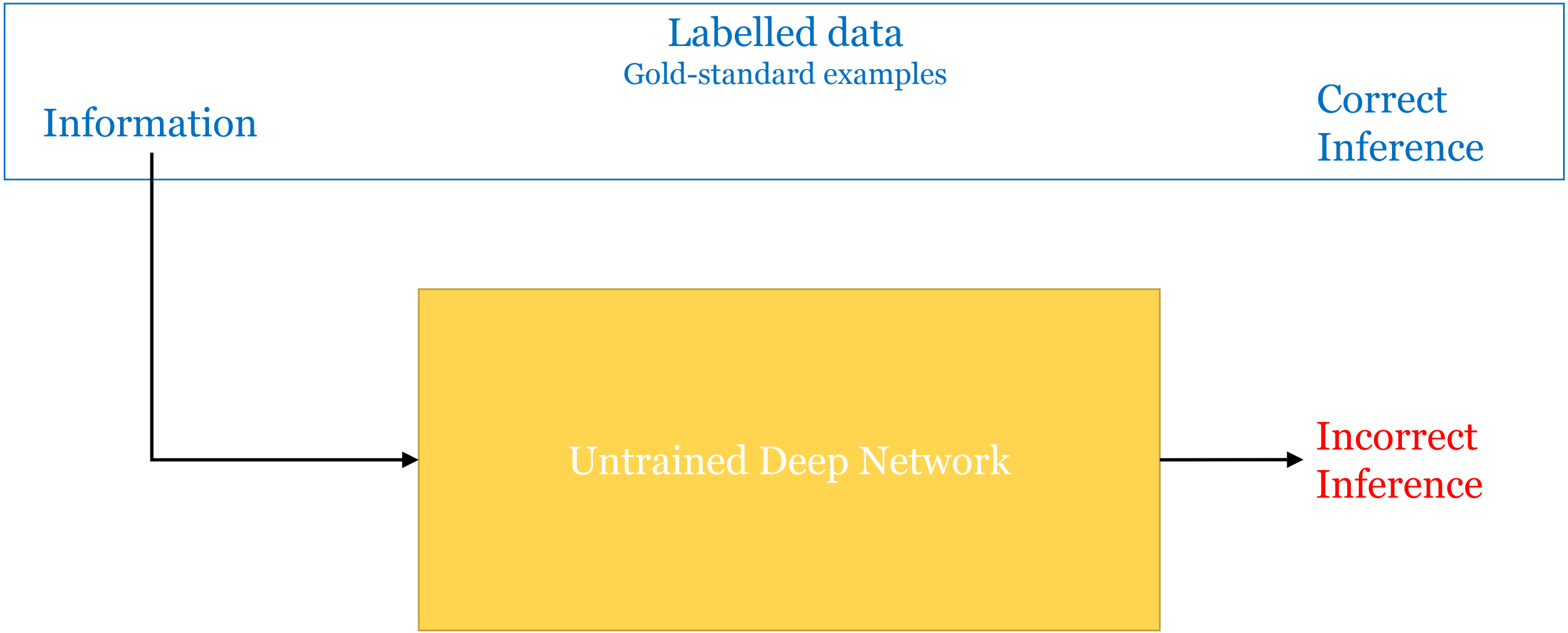
SUPERVISED LEARNING CHANGES A DEEP NETWORK'S PARAMETERS SO THAT IT EMULATES EXAMPLES

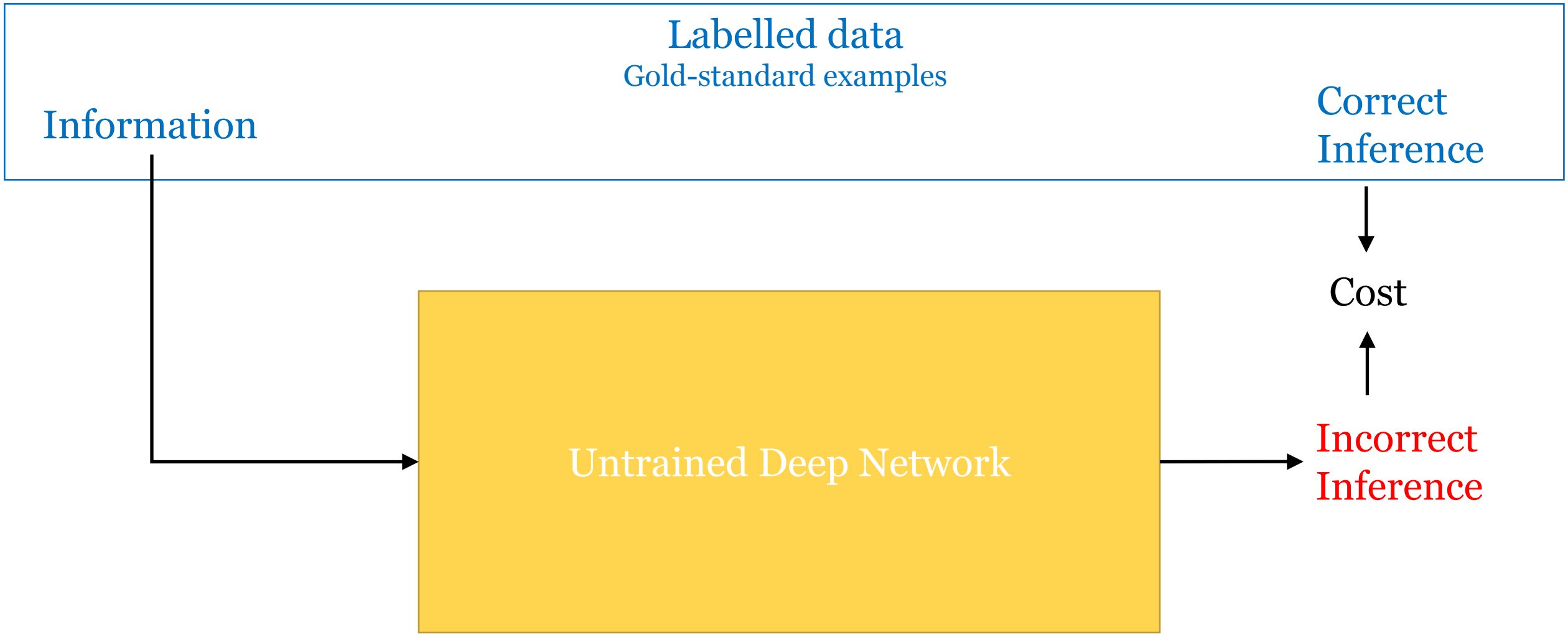
What we want

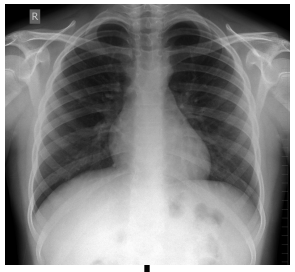

Information
E.g., image(s),
medical records
Deep Network
Inference
E.g., diagnosis,
length of stay

What we start with




Labelled data
Gold-standard examples
Information
Correct Inference
Untrained Deep Network
Incorrect Inference


Labelled data
Gold-standard examples
Information
Correct Inference
Cost
Untrained Deep Network
Incorrect Inference



Labelled data

Gold-standard examples

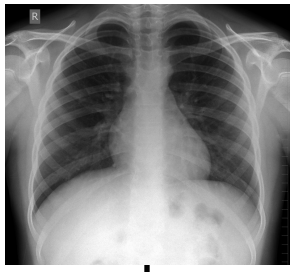
Viral pneumonia: 0
Bacterial pneumonia: 0
Not pneumonia: 1

Cost

.2
.65
0.15

Untrained Deep Network

Case courtesy of Dr Aneta Kecler-Pietrzyk, Radiopaedia.org, rID: 53075



Labelled data

Gold-standard examples

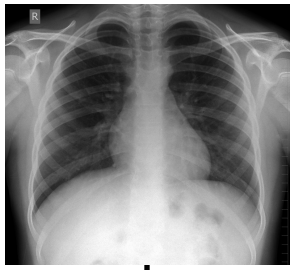
Viral pneumonia: 0
Bacterial pneumonia: 0
Not pneumonia: 1

How to change inference to reduce cost (increase 3rd number, etc.)

Cost

Untrained Deep Network

.2
.65
0.15



Labelled data

Gold-standard examples

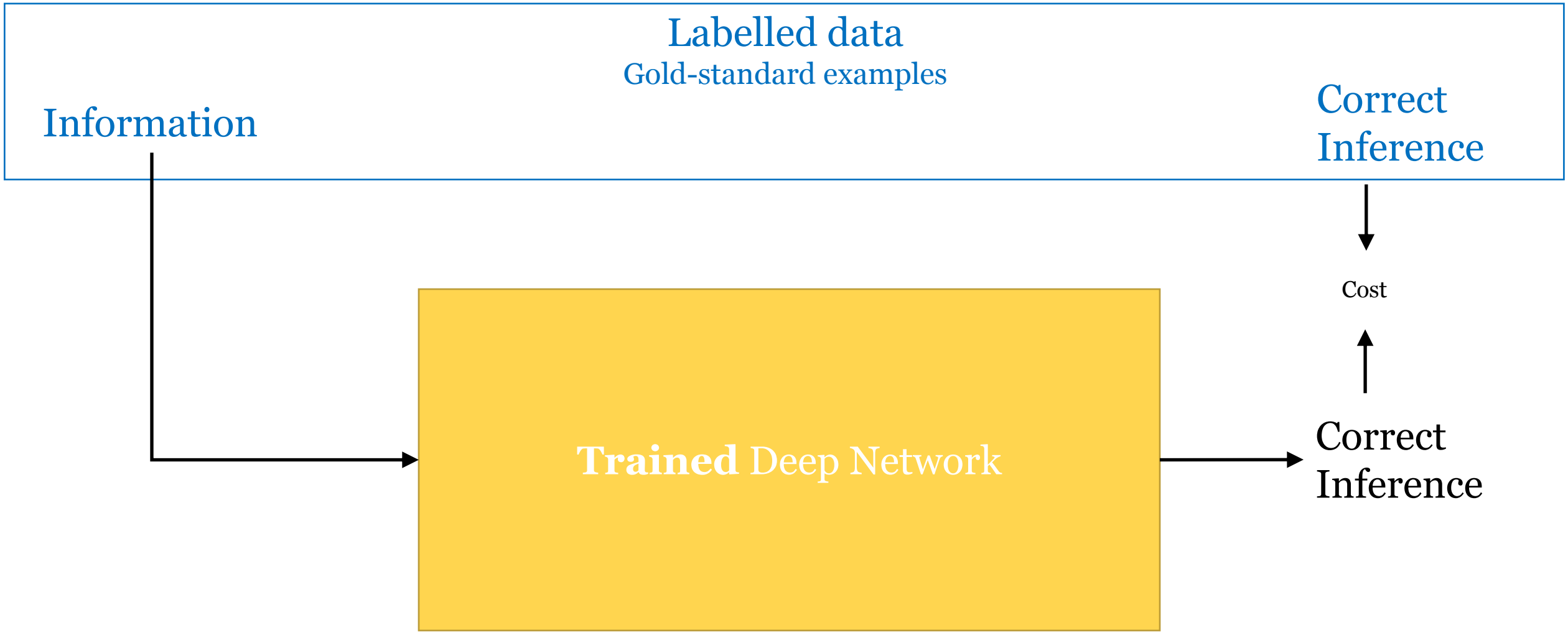
Viral pneumonia: 0
Bacterial pneumonia: 0
Not pneumonia: 1

How to change network parameters to change inference to reduce cost

Cost

Untrained Deep Network

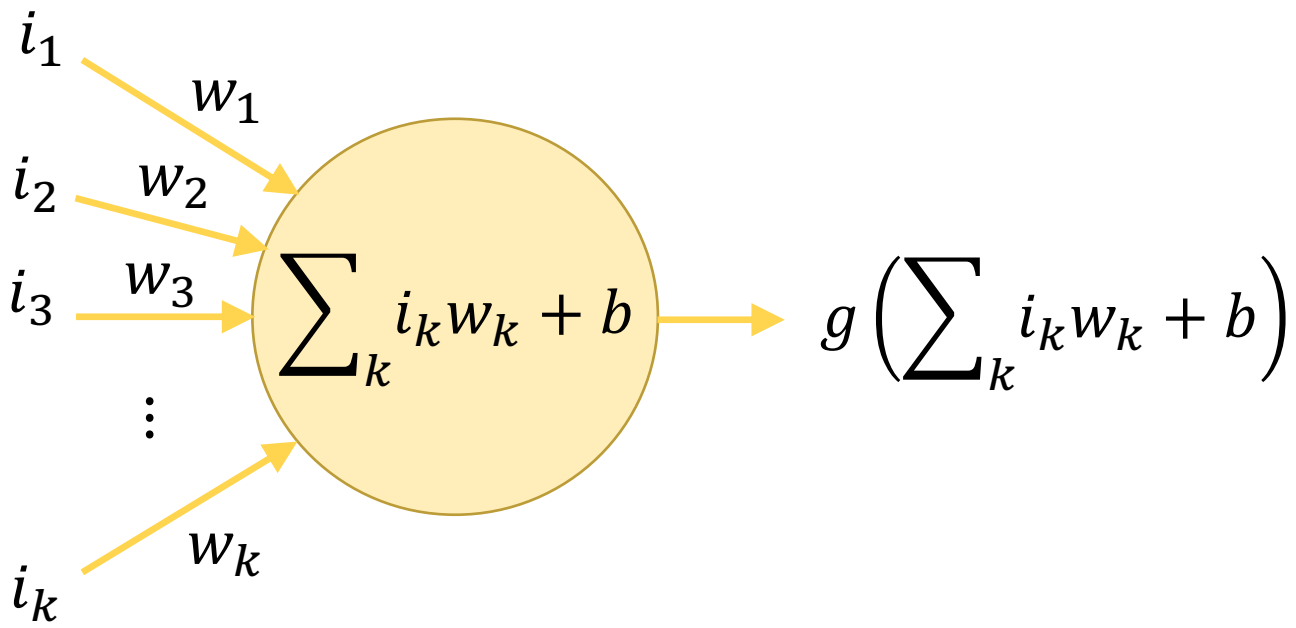
.2
.65
0.15


Labelled data
Gold-standard examples
Information
Correct Inference
Cost
Trained Deep Network
Correct Inference

AN ARTIFICIAL NEURON'S OUTPUT HAS A SIMPLE MATHEMATICAL RELATIONSHIP WITH ITS INPUT

Most artificial neurons are like this

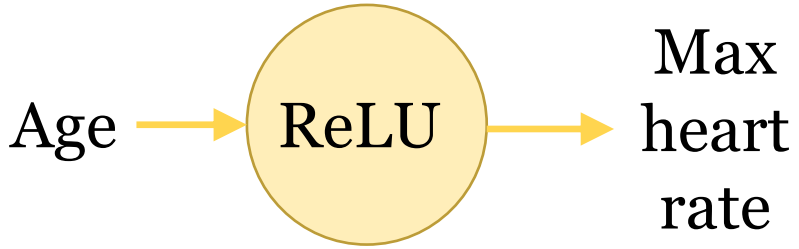
- i are inputs to the neuron
- w are connection weights
- b is a bias
- g is a function that is usually nonlinear. For example, in a rectified linear (ReLU) neuron, $g(\cdot) = \max(0,\cdot)$



TRAINING A NETWORK INVOLVES SETTING THE WEIGHTS AND BIASES SO THAT THE NETWORK DOES SOMETHING USEFUL

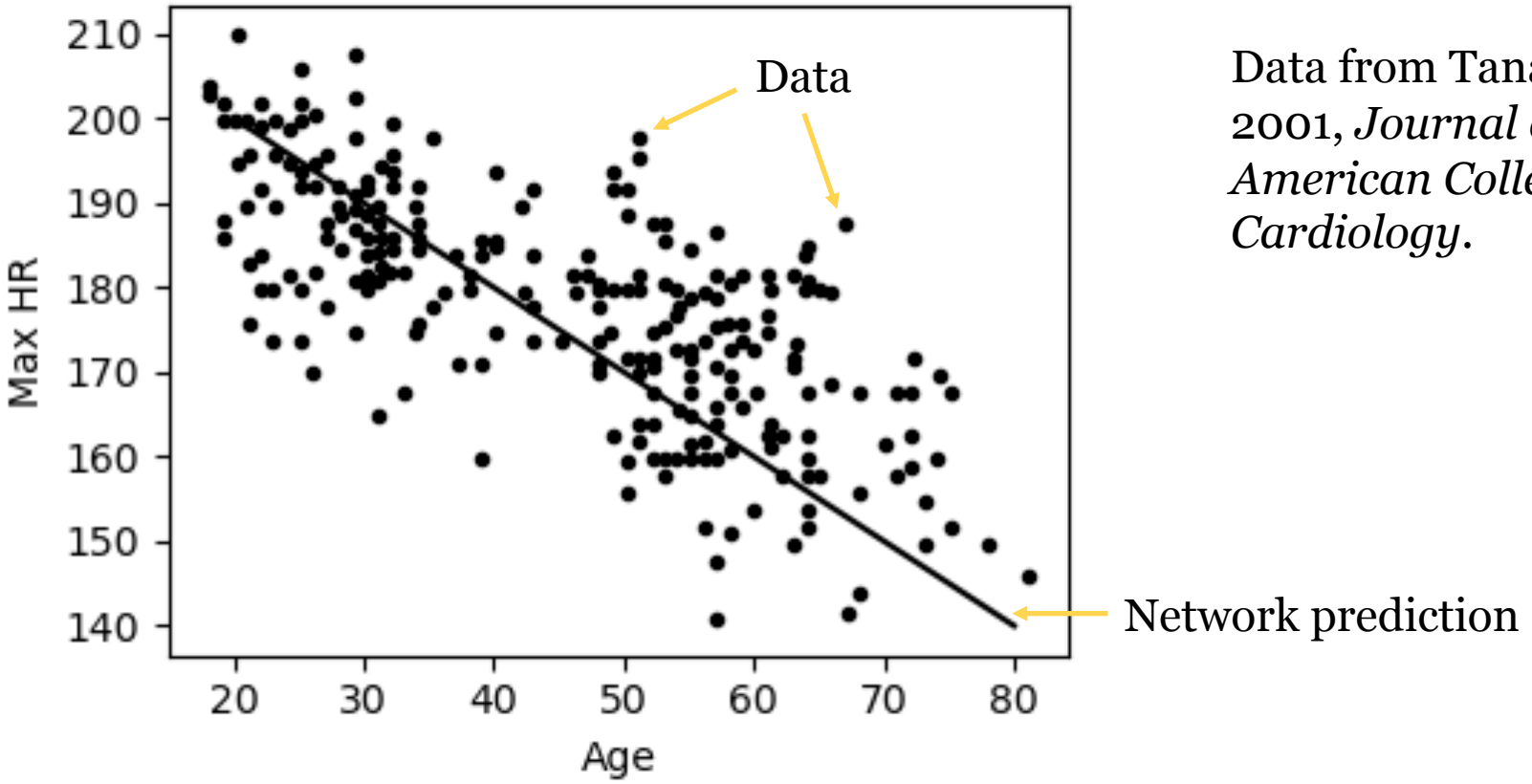
Simple example: Predicting max heart rate from age

- Approximation: max heart rate equals 220 minus person's age
- We just need one neuron to implement this relationship
  - Weight and bias?



Simple example: predicting max heart rate from age

Is the weight too high, too low, or just right?



Data from Tanaka et al., 2001, *Journal of the American College of Cardiology*.

A NETWORK'S ERRORS MUST BE SUMMARIZED AS A SINGLE NUMBER

Loss

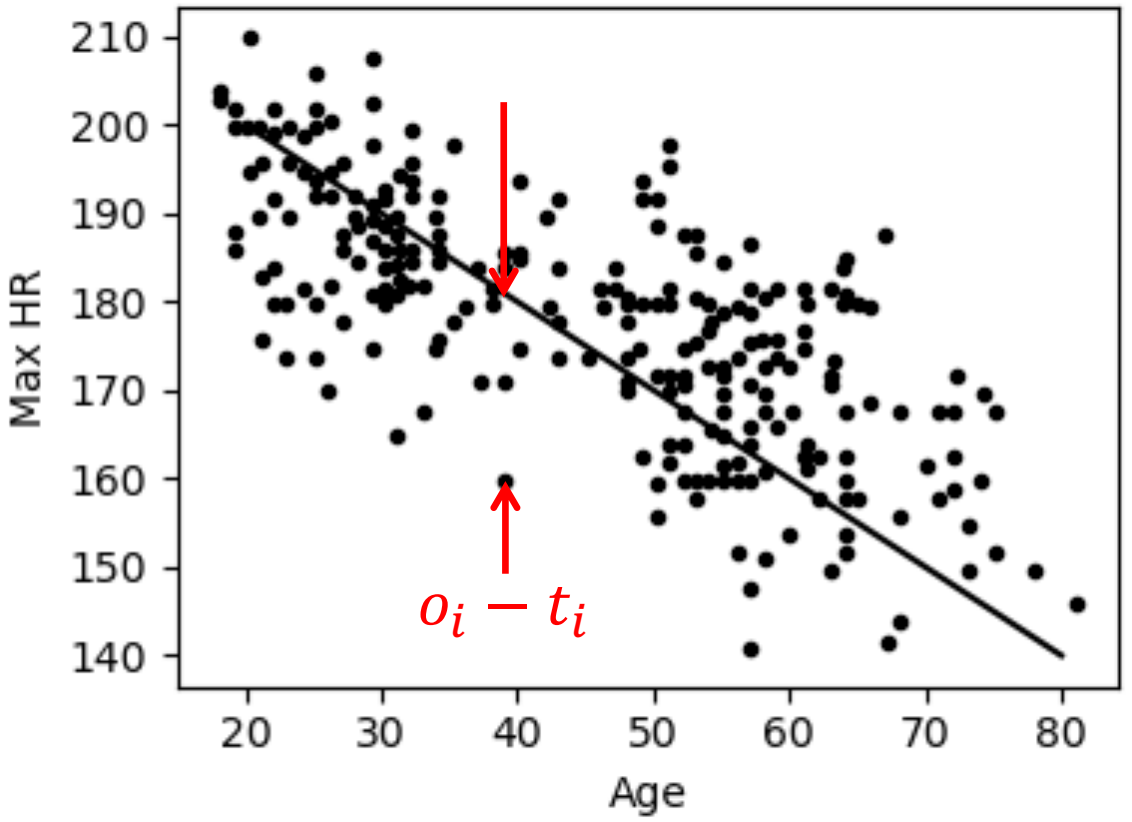
- Usually, we don't know what a networks weights and biases should be, and it isn't possible to derive them analytically
- We must adjust them empirically according to their effect on the network's performance
- To do this, the network's mistakes must be summarized as a single number, which is called the "loss" or "cost"
- The loss is a function of the difference between the network's outputs and the correct outputs (the "targets")

Mean-squared error loss

- For regression tasks (prediction of continuous values), a typical loss is the mean-squared error,

$$\frac{1}{n}\sum_{i=1}^{n}(o_i - t_i)^2$$

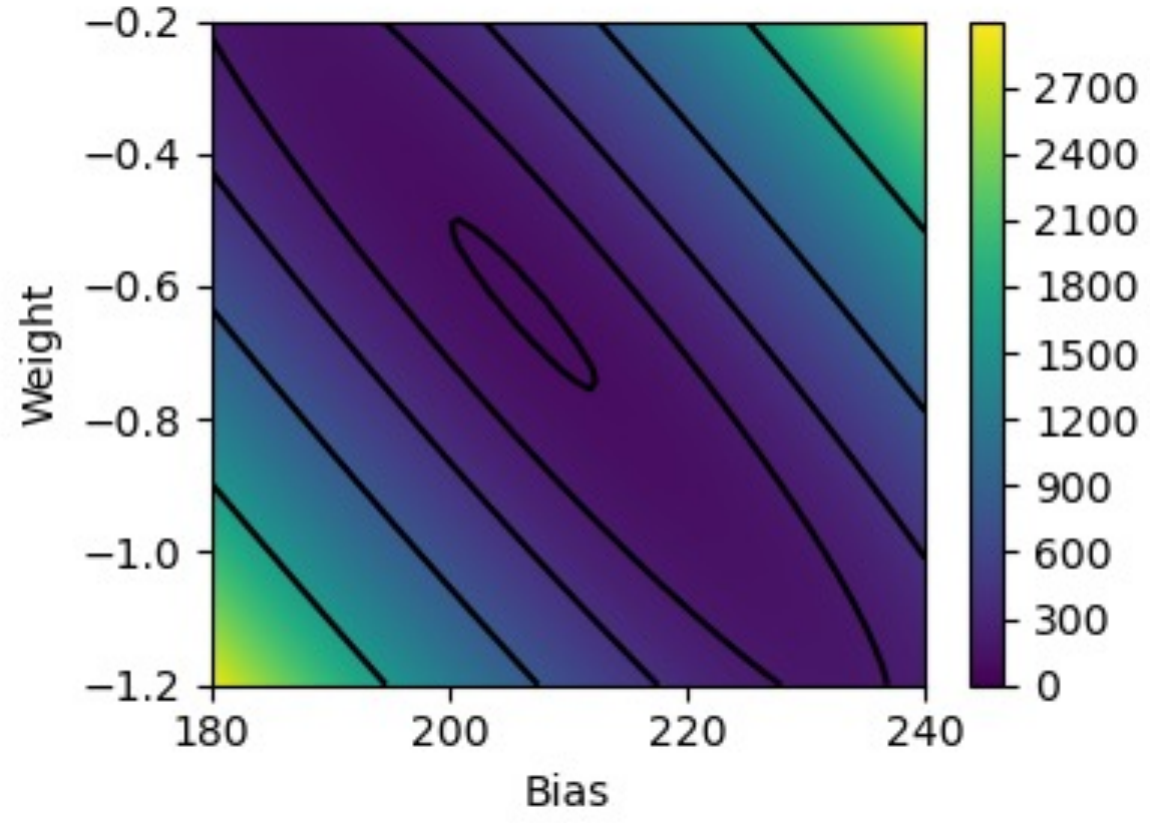
- i is the index of a "labelled" example, of which there are n
- Each example consists of an input (e.g., an age) and a correct output (e.g., a max heart rate)
- o_i is the network's output and t_i is the correct output, or target value



A NETWORK'S WEIGHTS AND BIASES ARE IMPROVED BY GRADIENT DESCENT

Loss vs. parameters

- Once we choose a dataset, network structure, and cost function, the loss depends only on the network's parameters
- Our max heart rate network has only two parameters, so we can plot the loss as a function of these using a contour map
- The lowest cost is not produced by weight -1 and bias 220, but by values closer to 207 and -0.65 (the darkest purple area)

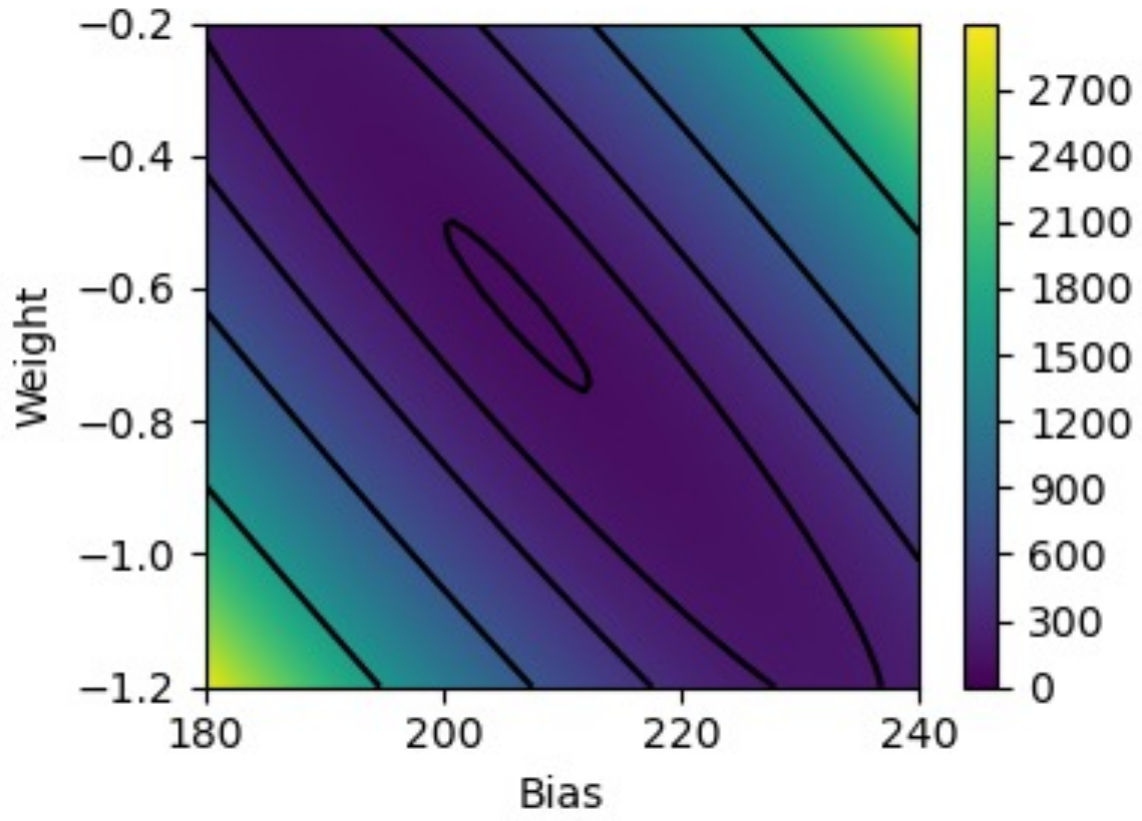


Loss vs. parameters

- Deep networks typically have millions of parameters rather than just two, so we can't produce a contour map
  - It would take forever to calculate cost throughout the space of parameters
- Fortunately, we just need one set of weight and bias values that gives us an acceptably low cost

Loss vs. parameters

- To find such values we move downhill
- Starting at -1 and 220, the downhill direction (as seen on the contour map) is roughly upwards toward the top of the figure
- After moving up a little, the downhill direction would curve to the left, toward the centre of the smallest contour

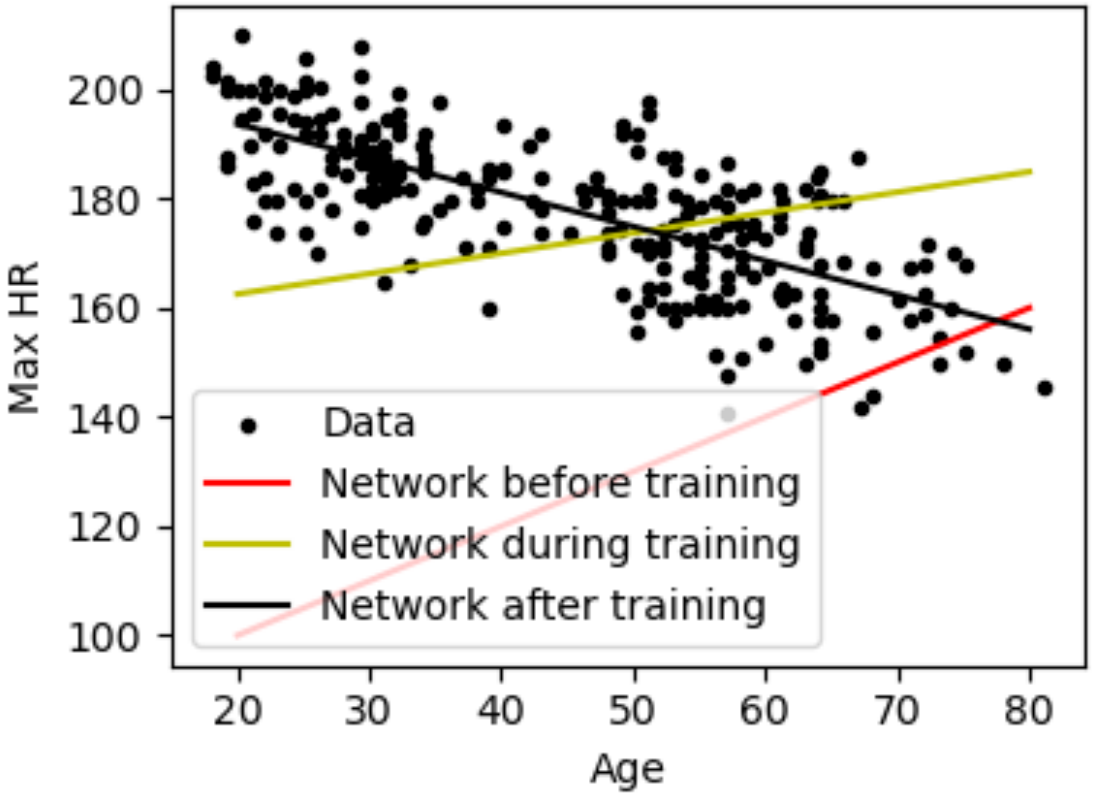


Loss vs. parameters

- Rather than calculating the whole contour map, deep learning software figures out which direction is downhill given the current parameters
- Then it takes small, cautious steps downhill, because it doesn't know in advance how far to go, and because the downhill direction may change as the weights and biases change
- This approach is called "gradient descent"
  - The gradient of the loss with respect to the weights and biases (the uphill direction)
  - We move the network parameters opposite to that direction (downhill/descending)

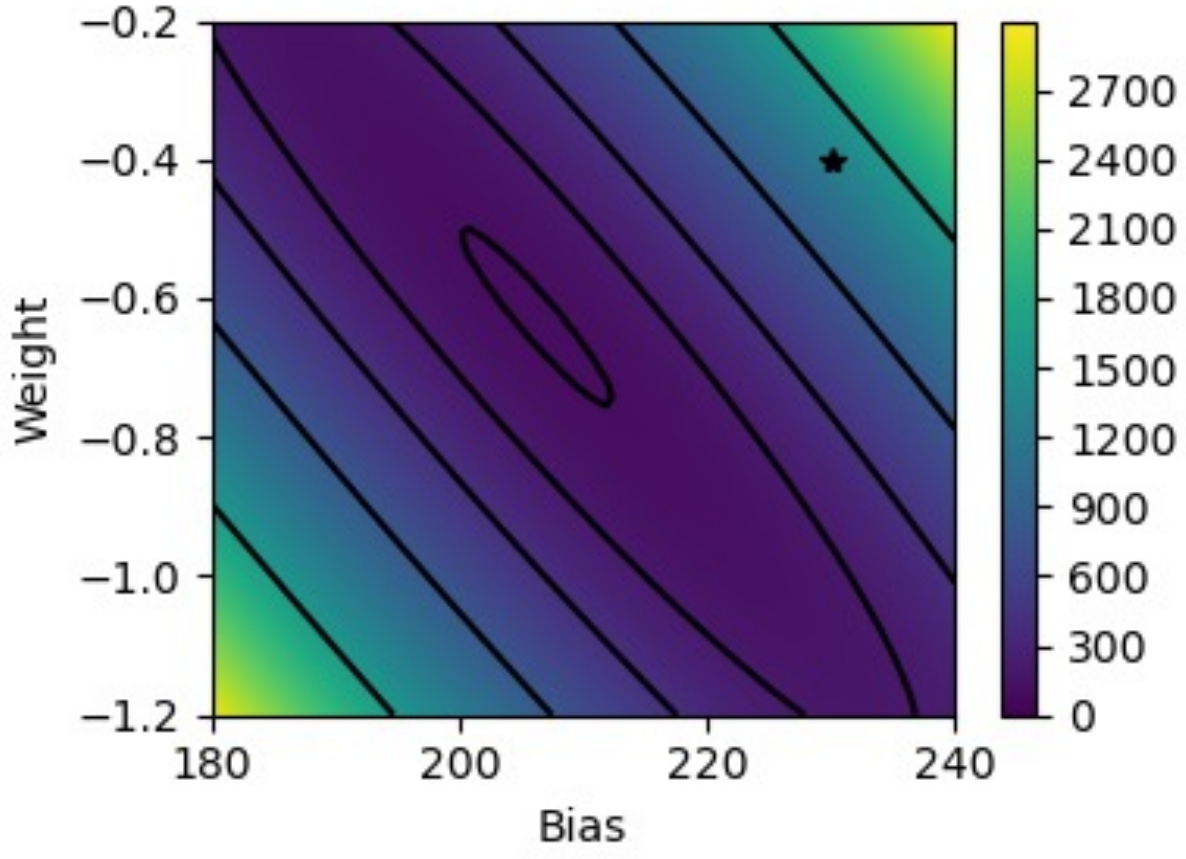
Improvement due to gradient descent

- The plot below shows a network that starts out with a weight and bias that are very wrong (red line)
- The network is trained to reduce the cost, leading to improved parameters (yellow) part way through the process, and eventually optimal parameters (black line)



Question

- Suppose a network has the cost function shown in the plot below, and the weight and bias values are as indicated by the star. In which direction would a gradient descent step take these parameters?



DEEP NETWORKS MUST GENERALIZE FROM EXAMPLES

Generalization is essential

- In the previous example, max heart rate was measured using a treadmill test in which the treadmill grade was increased until volitional exhaustion
- The network's appeal is that it gives an estimate without requiring a treadmill test
- For a given person, the network's prediction might be off by ten beats per minute or so, but this may be useful enough for some purposes (e.g., target heart rates for a prescribed exercise program)
- The network is not useful for predicting the data points that were used to train it, because these data points are already known; a network's only use is in generalizing beyond these data points

Generalization is not guaranteed

- The network is only useful for people who are *outside* the dataset, but we can only adjust weights and biases so that the network makes better predictions for people *within* the dataset
- Differences between these groups (which always exist to some extent) can make the network less useful and potentially misleading. Differences can arise in two ways:
  - If there are systematic differences between those inside and outside the training dataset; e.g., if a deep network were trained to classify skin lesions using labelled data from Caucasians, it might not work as well with other patients.
  - If the system excessively approximates random variations in the dataset
- Both these situations are common

Question

- True or false: It is best to apply a deep network only to the labelled examples on which it was trained

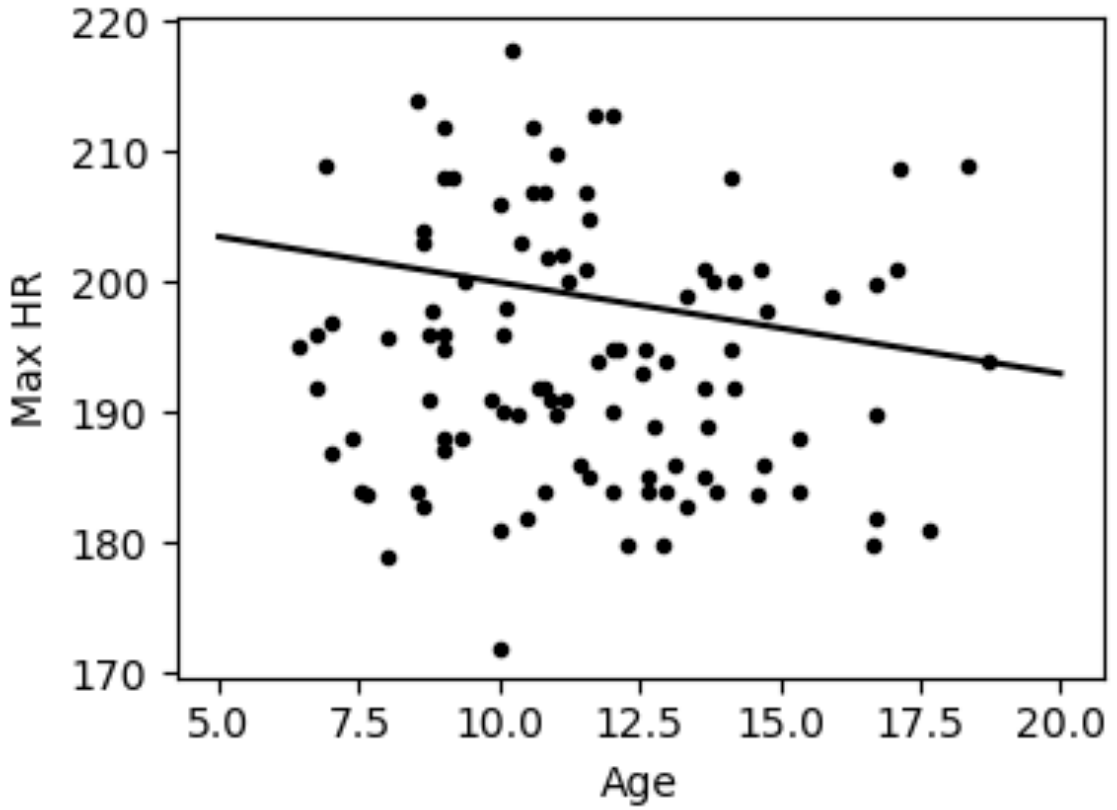
A DEEP NETWORK SHOULD ONLY BE APPLIED TO THE DISTRIBUTION ON WHICH IT WAS TRAINED

Defining the training distribution

- The study of max heart rate that we used included people who:
  - Did not smoke
  - Did not use medications (other than hormone replacement for postmenopausal women)
  - Had a body mass index under 35 kg/m^2
  - Were between the ages of 18 and 81
- A network trained with this dataset should only be used for people in this group

Venturing outside the training distribution

- For example, max heart rate has little association with age in younger people (Verschuren et al., 2011, Developmental Medicine and Child Neurology), so the network's predictions are not as realistic for younger people
- Here are data from Verschuren et al. along with our network's prediction
- The network predicts a slope that isn't evident in the data



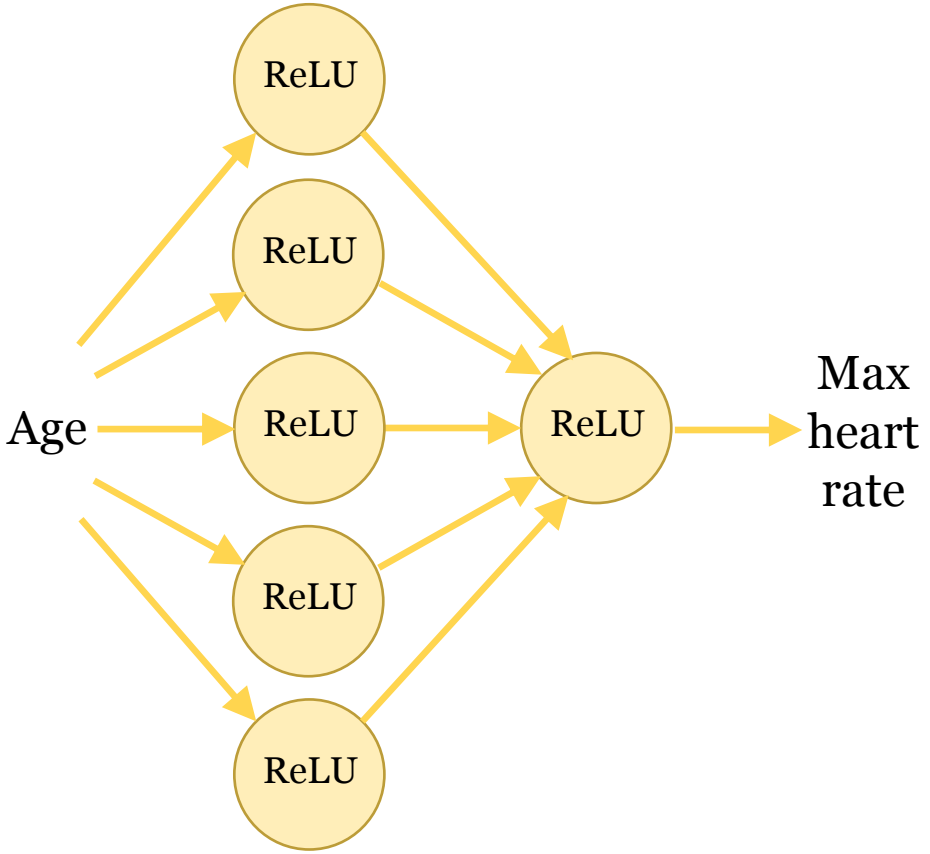
A DEEP NETWORK MUST BE TRAINED AND VALIDATED WITH DIFFERENT DATASETS

Overfitting

- Problems may remain even if a deep network is trained on examples that are drawn at random from the target distribution
- Any sample from a distribution will have its own peculiarities that don't apply to the distribution as a whole
- A deep network may learn the random quirks of its training dataset, in addition to learning the population trends
- This phenomenon is called "overfitting"; it always happens to some degree
- For a given task, this problem tends to become worse with smaller datasets, and with more complex networks

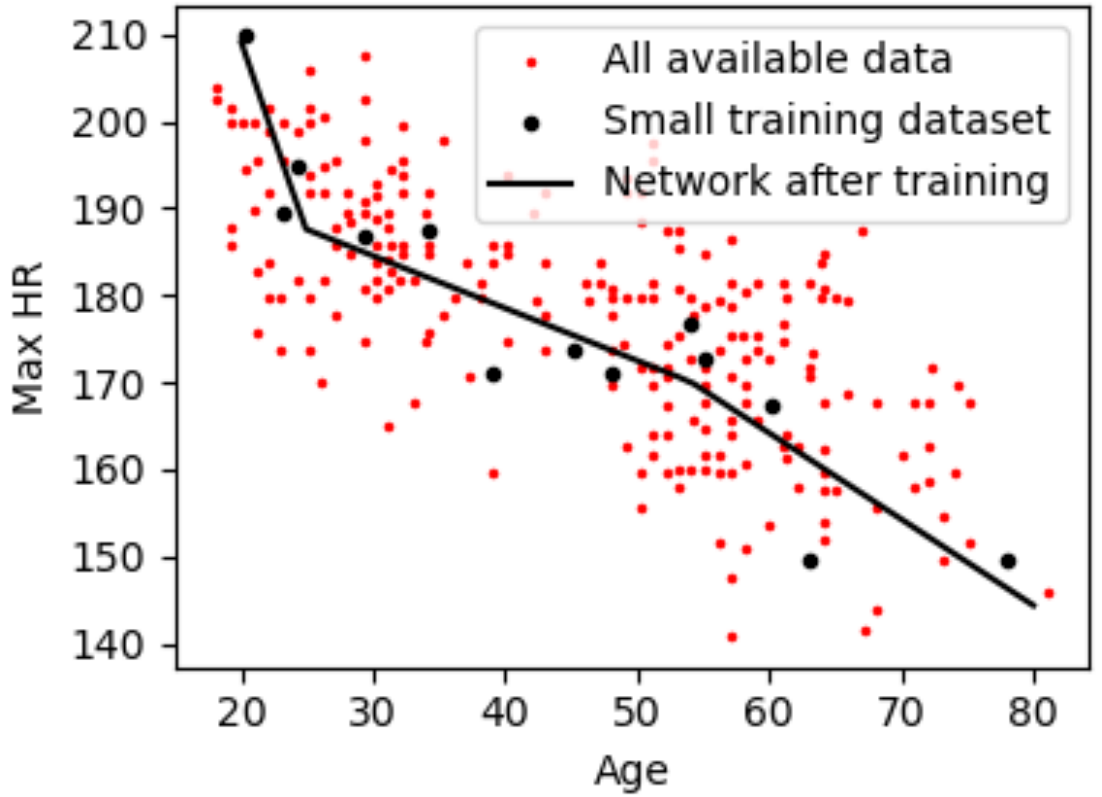
Overfitting

- Example: To illustrate, let's train a more complex ANN with a small subset of the adult max heart rate data
- The network shown here can learn a relationship between age and max heart rate that is not a straight line
- In principle, this could improve accuracy, because such a system could learn that max heart only varies with age in adults
- However, this greater flexibility can be counter-productive



Overfitting

- As shown here, the more complex network may learn to approximate the random ups and downs of the small sample
- This makes its cost lower, because it approximates the training examples more closely, but its predictions tend to be less accurate for new patients
- In this example, this is particularly true for patients about 20 years old, where the system has been excessively influenced by an example with a very high max heart rate



Reducing overfitting

- For a given task, there is less overfitting with
  - Larger datasets (but these are expensive)
  - Less complex networks (but these may not be powerful enough to make good predictions)
  - Regularization

Measuring overfitting

- The available data is divided into at least two parts
- The first part is a “training dataset” that is used to calculate the loss for gradient descent; the network learns to approximate this dataset
- A second part is usually called a “validation dataset ”; this is not used for training, but only to test the network’s accuracy on separate, “held out” data
- Accuracy on the validation dataset gives a better estimate of how well the network will perform in the real world, with new examples that it has not encountered during training

TWO DATASETS MAY NOT BE ENOUGH

Overfitting the validation data

- We usually experiment with various hyperparameter combinations to see which one performs best on the validation set
- With such experimentation, one may find a network that generalizes better to the validation dataset than to unseen data in general
- To see how much this has happened, labelled examples are usually divided into *three* parts: training, validation, and an additional "test" dataset
- The test set is only used after the design & hyperparameters have been finalized
- If the final network's performance on the test dataset is much worse than performance on the validation dataset, the development process as a whole has overfit the validation data

Question

- Which better predicts a system's accuracy in practice, with new inputs that have never been encountered before?
  - Accuracy on the validation dataset, or
  - Accuracy on the test dataset?

SUMMARY

1. The best performing machine learning systems are usually based on deep networks or decision trees
2. Deep networks are artificial neural networks that are large and often complex
3. Important elements of a deep learning system include inputs, outputs, architecture, loss, parameters, optimization algorithm, and hyperparameters
4. Supervised learning changes a deep network's parameters so that it emulates examples
5. An artificial neuron's output has a simple mathematical relationship with its input
6. Training a network involves setting the weights and biases so that the network does something useful

7. A network's errors must be summarized as a single number
8. A network's weights and biases are improved by gradient descent
9. Deep networks must generalize from examples
10. A deep network should only be applied to the population on which it was trained
11. A deep network must be trained and validated with different datasets
12. Two datasets may not be enough

References

- Olson, R. S., Cava, W. L., Mustahsan, Z., Varik, A., & Moore, J. H. (2018). Data-driven advice for applying machine learning to bioinformatics problems. In *Pacific Symposium on Biocomputing 2018: Proceedings of the Pacific Symposium* (pp. 192-203).
- Sebald, A. V. (1989). Use of neural networks for detection of artifacts in arterial pressure waveforms. In Images of the Twenty-First Century. *Proceedings of the Annual International Engineering in Medicine and Biology Society*, (pp. 2034-2035). IEEE.
- Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A. (2017, February). Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-First AAAI Conference on Artificial Intelligence*.
- Tanaka, H., Monahan, K. D., & Seals, D. R. (2001). Age-predicted maximal heart rate revisited. *Journal of the American College of Cardiology*, 37(1), 153-156.
- Verschuren, O., Maltais, D. B., & Takken, T. I. M. (2011). The 220-age equation does not predict maximum heart rate in children and adolescents. *Developmental Medicine & Child Neurology*, 53(9), 861-864.