Ling/CSE 472: Introduction to Computational Linguistics

4/30: Neural language models

Overview

- Neural nets and language processing: Some history
- XOR and “representations”
- Reading questions

Some history

- McCulloch-Pitts neuron: 1940s
- Perceptron: Rosenblatt 1958
- Single perceptron can't do XOR: Minsky & Paper 1969
- Error backpropagation: Rumelhart et al 1986
- 1980s: Neural nets as models of human cognition
- 1990s: early NLP applications (handwriting recognition: LeCun et al 1989, 1990; ASR: Morgan and Bourlard 1989, 1990)
- 2000s: "deep learning" (Hinton et al 2006, Bengio et al 2007)

A basic neural unit

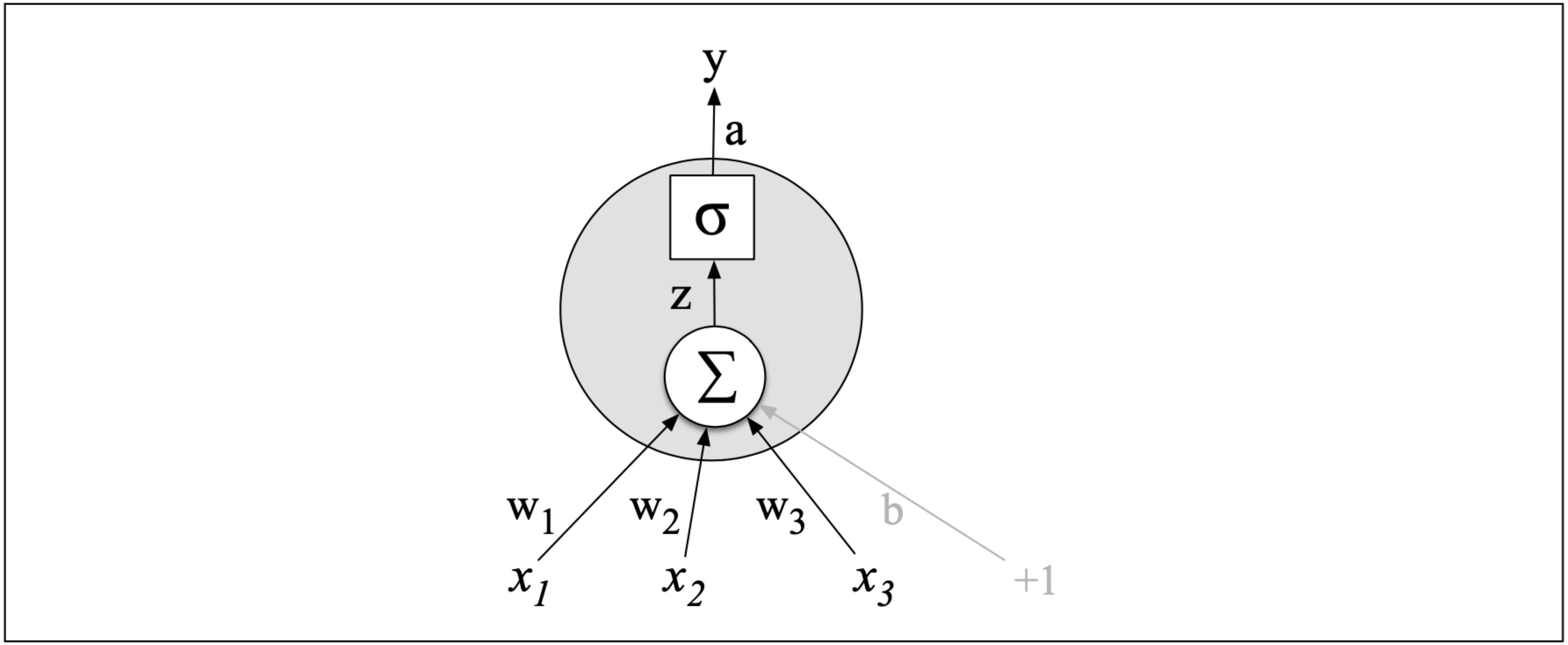
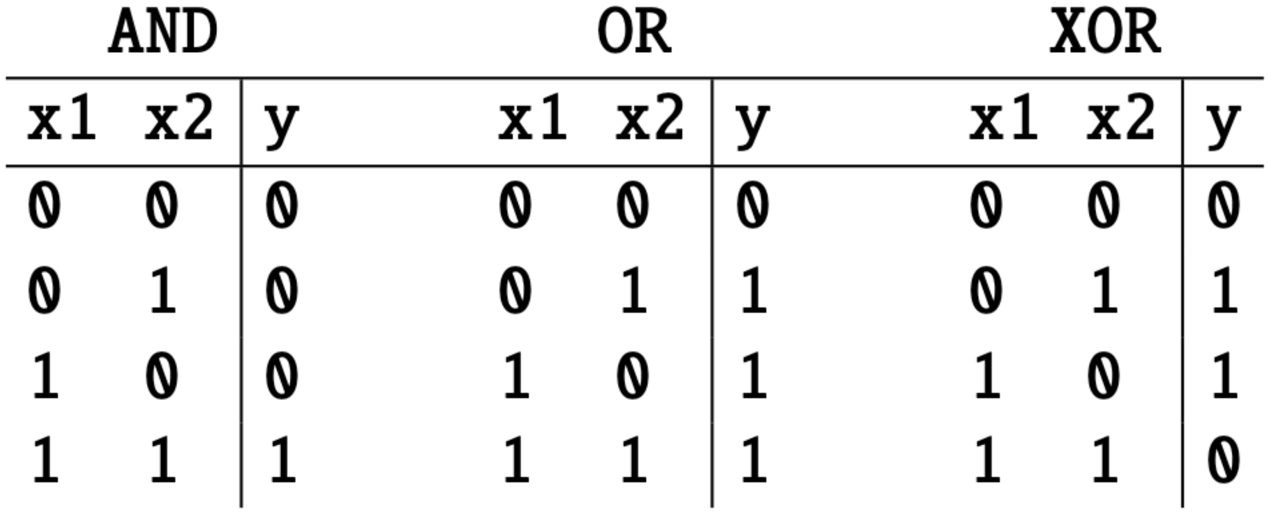


Figure 7.2 A neural unit, taking 3 inputs x_1, x_2, and x_3 (and a bias b that we represent as a weight for an input clamped at +1) and producing an output y. We include some convenient intermediate variables: the output of the summation, z, and the output of the sigmoid, a. In this case the output of the unit y is the same as a, but in deeper networks we'll reserve y to mean the final output of the entire network, leaving a as the activation of an individual node.

The XOR problem, with a single perceptron



AND			OR			XOR		
x1	x2	y	x1	x2	y	x1	x2	y
0	0	0	0	0	0	0	0	0
0	1	0	0	1	1	0	1	1
1	0	0	1	0	1	1	0	1
1	1	1	1	1	1	1	1	0

$$y = \begin{cases} 0, & \text{if } w \cdot x + b \leq 0 \\ 1, & \text{if } w \cdot x + b > 0 \end{cases}$$

The XOR problem, with a single perceptron

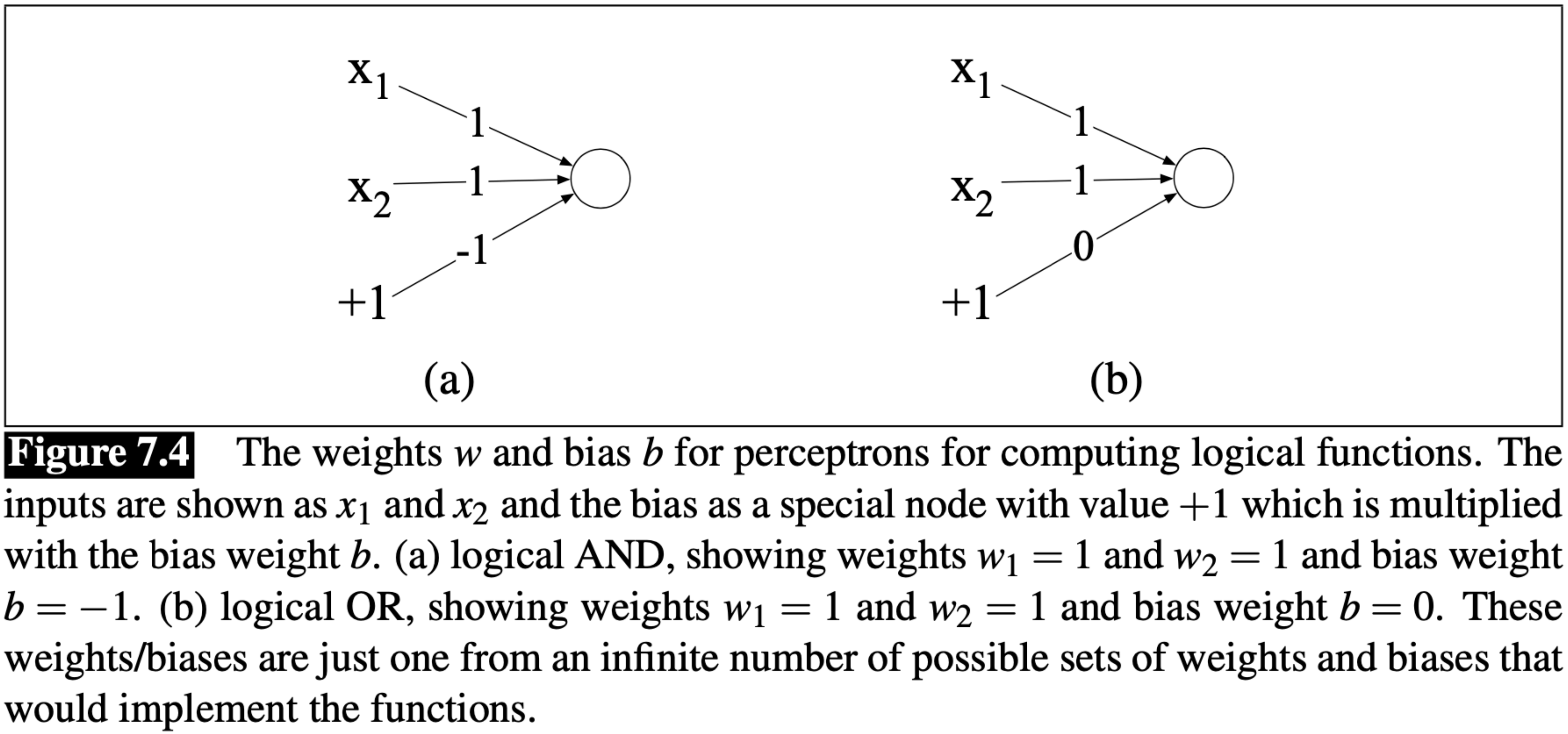
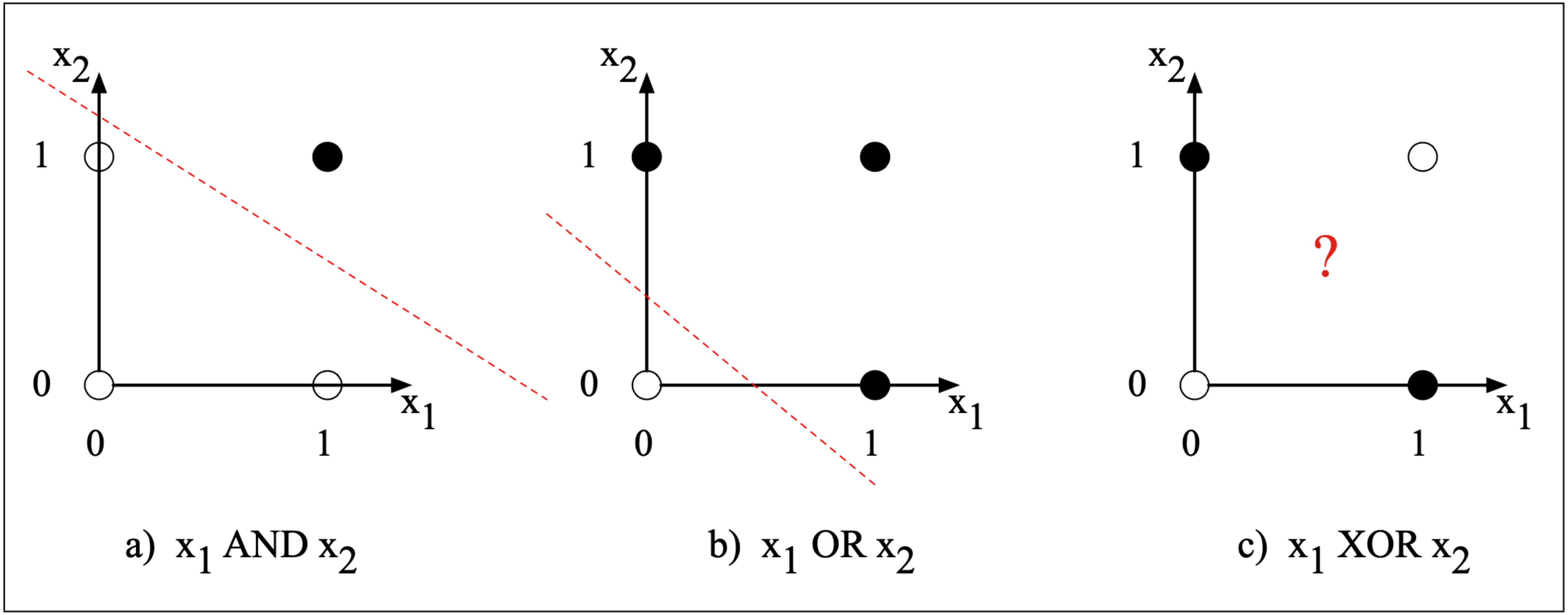


Figure 7.4 The weights w and bias b for perceptrons for computing logical functions. The inputs are shown as x_1 and x_2 and the bias as a special node with value $+1$ which is multiplied with the bias weight b. (a) logical AND, showing weights $w_1 = 1$ and $w_2 = 1$ and bias weight $b = -1$. (b) logical OR, showing weights $w_1 = 1$ and $w_2 = 1$ and bias weight $b = 0$. These weights/biases are just one from an infinite number of possible sets of weights and biases that would implement the functions.

The XOR problem, with a single perceptron



a) x_1 AND x_2

b) x_1 OR x_2

c) x_1 XOR x_2

Figure 7.5 The functions AND, OR, and XOR, represented with input x_1 on the x-axis and input x_2 on the y axis. Filled circles represent perceptron outputs of 1, and white circles perceptron outputs of 0. There is no way to draw a line that correctly separates the two categories for XOR. Figure styled after Russell and Norvig (2002).

XOR solution, with a hidden layer and ReLU units

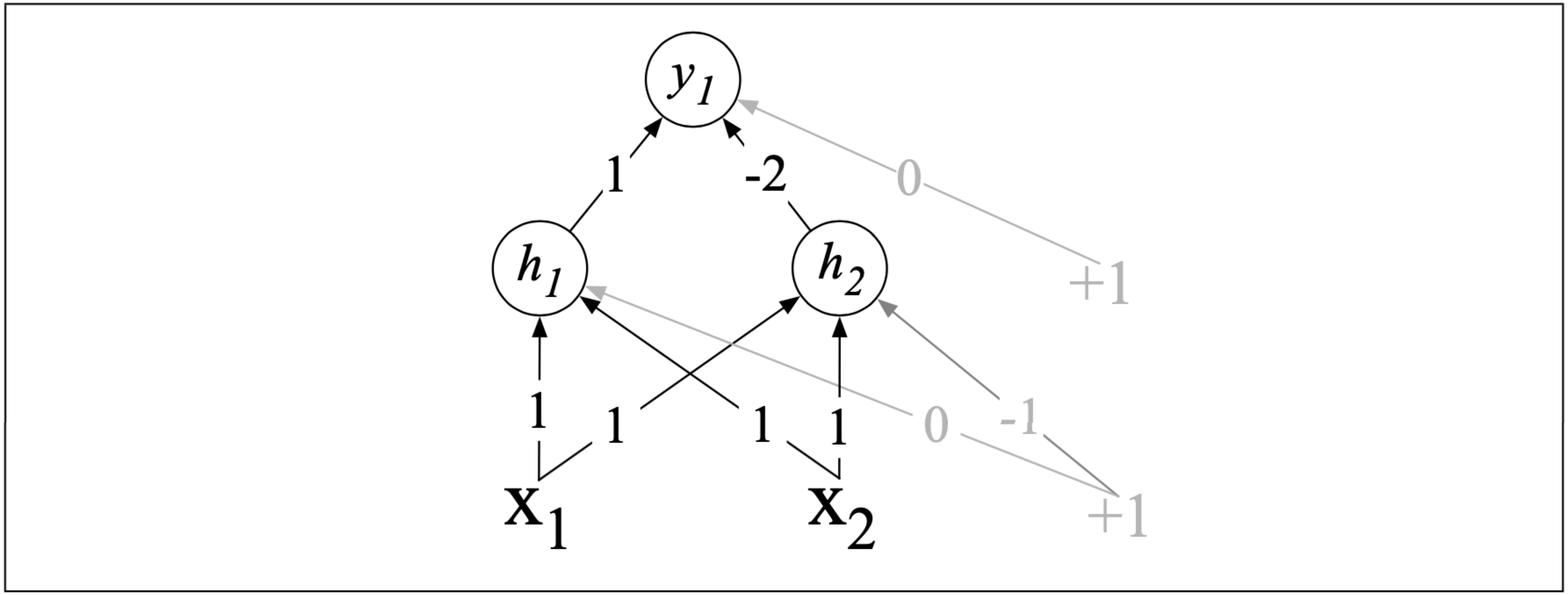
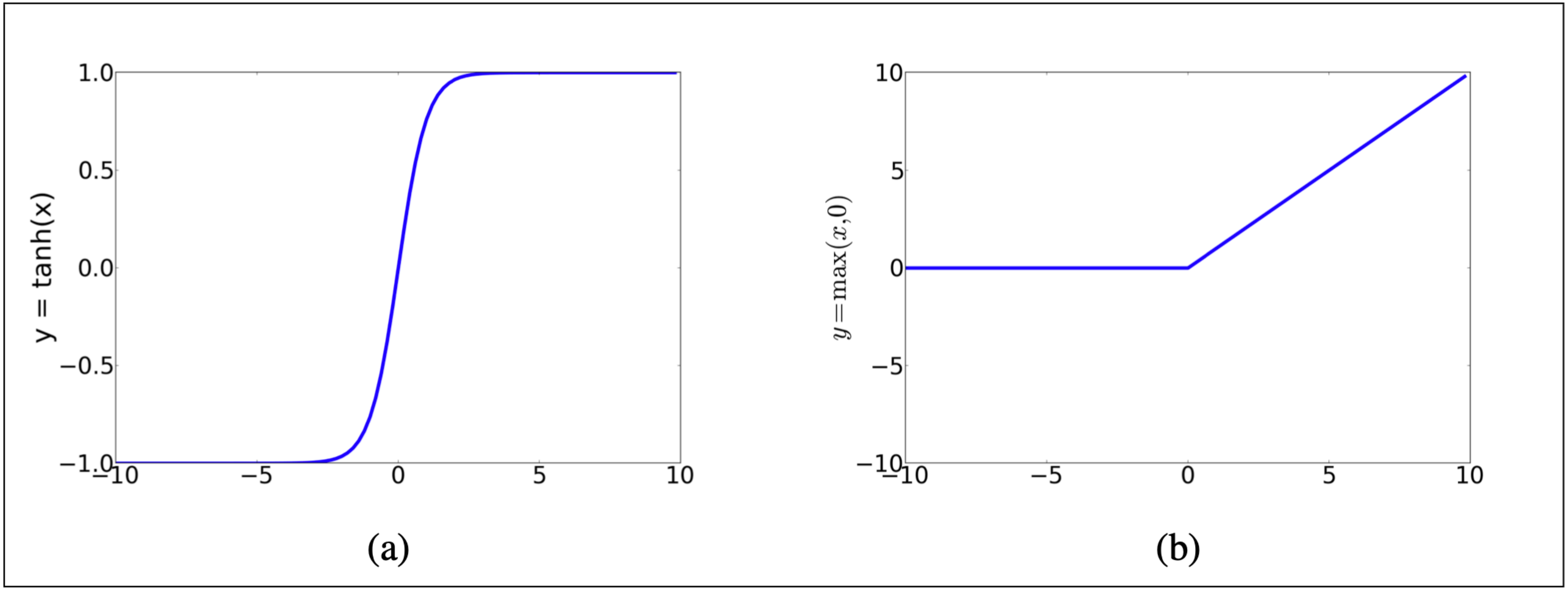


Figure 7.6 XOR solution after Goodfellow et al. (2016). There are three ReLU units, in two layers; we've called them h_1, h_2 (h for "hidden layer") and y_1. As before, the numbers on the arrows represent the weights w for each unit, and we represent the bias b as a weight on a unit clamped to +1, with the bias weights/units in gray.

tanh and ReLU activation functions

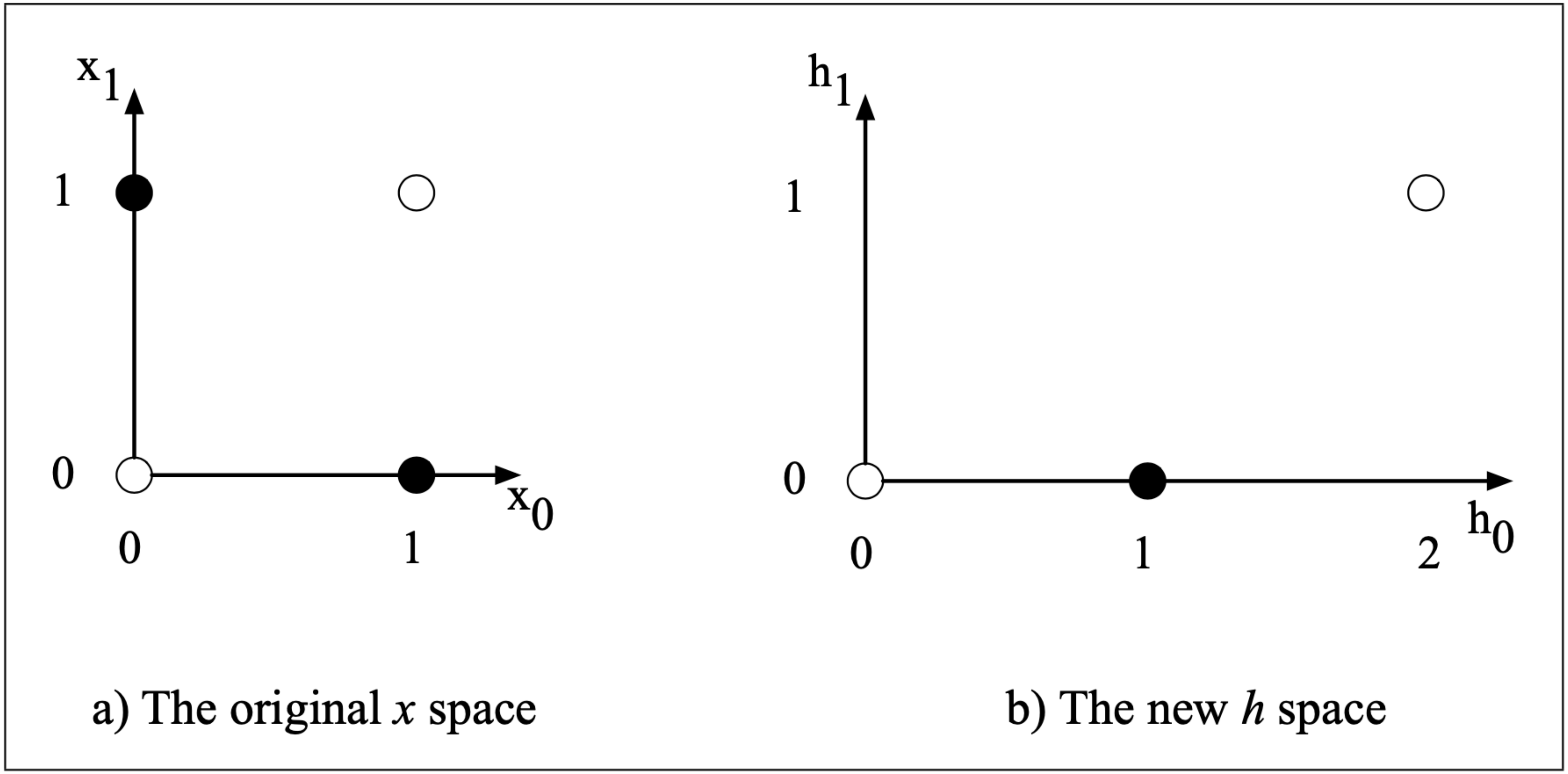


(a)

(b)

Figure 7.3 The tanh and ReLU activation functions.

XOR solution, with a hidden layer and ReLU units



a) The original *x* space

b) The new *h* space

Figure 7.7 The hidden layer forming a new representation of the input. Here is the representation of the hidden layer, *h*, compared to the original input representation *x*. Notice that the input point [0 1] has been collapsed with the input point [1 0], making it possible to linearly separate the positive and negative cases of XOR. After Goodfellow et al. (2016).

“Representations”

- “Notice that hidden representations of the two input points *x* = [0 1] and *x* = [1 0] (the two cases with XOR output = 1) are merged to the single point *h* = [1 0]. The merger makes it easy to linearly separate the positive and negative cases of XOR. In other words, we can view the hidden layer of the network as forming a representation for the input.”

- “In this example we just stipulated the weights in Fig. 7.6. But for real examples the weights for neural networks are learned automatically using the error backprop- agation algorithm to be introduced in Section 7.4. That means the hidden layers will learn to form useful representations.”

(J&M Ch7 p6, emphasis added)

Reading questions: All the math

- This reading used a lot of mathematical terminology and assumed a lot of background knowledge. What is logistic regression? What is a sigmoid function? What is gradient decent? Normalization, bias, vectors, and softmax function are all...a lot.

- I noticed that a few of the readings we have done recently have been very math-heavy. How exactly is the math incorporated into the building of the systems? And is it something that stays pertinent through the functioning of the system? (i.e. does the developer have to consistently be making calculations?)

Reading questions: All the math

- What does it mean when they talk about linearity vs. non-linearity in the neural networks?

- Do we calculate the probability of each word occurring given the N words before it in the dataset before training the network?

- What are the consequences of not having a probability distribution for the next word (such as in stupid backoff)?

- What is gradient descent, and how does it relate to optimization?

Reading questions: Language modeling

- Is language modeling always a matter of predicting upcoming text?

Reading questions: Distributional semantics

- How would neural language models know that "cat" and "dog" have similar embeddings? Would an animal such as "horse" also have a similar embedding because it is an animal or would it be different because it isn't a common pet that people have?

- I'm not quite sure I understand what the reading is talking about when it talks about a word having an "embedding". Do words with similar embeddings have similar word sense / contexts that they would be used in?

Reading questions: Pre-training

- What are examples of cases in which pretrained word embeddings are sufficient?

- I am curious about what is the difference between pretrained embedding and training embedding from scratch. Also, since we are talking about word2vec, which should be a one-direction, what will the graph be when we are talking about bi-directional like BERT.

Reading questions: Network architecture

- In what situations (and how) do you know that adding more layers to a model, beyond those implied by the obvious constraints of things like the XOR problem, will improve performance? It seems like a lot of really current models aren’t well-understood in the first place, so I'm wondering if something as fundamental as the number of layers is part of that. Another way of phrasing this might be: can we discern what's going on at different layers of a network (despite being "hidden")?

Reading questions: Network architecture

- I'm interested in the developers roll in developing and fine tuning a neural network. Using a dev set of data, how does a a developer decide hyperparameters like model architecture? Wouldn't that require re-training the model to see what works best?

- The concept of backward differentiation/propagation is really interesting as it seems like magic - this is quite some use of the chain rule. The text mentions dropout, which is randomly dropping out units to minimize overfitting. I wonder if it is always random or if there exist some more deterministic way of dropping units to reduce overfitting.

Reading questions: Applications

- It's interesting neural language systems are better than the n-gram models in most ways, but because they take longer to train, n-grams are mostly still preferred. How long does it take to train a neural language system and what makes it worth it rather than just using an n-gram?

- What specifically causes the neural net language models to be much slower to train than traditional language models? And how much slower would it take compared to n-gram language models run on the same training sets?

- Are there language models that combine the best of two worlds, using n-gram modeling in conjunction with neural network learning? If so, how does the performance of such a combination compare against the performance of independent n-gram language models and neural networks?

Reading questions: : Applications

- What are the most prominent areas of language processing tasks that are difficult to perform in the context of neural network and what are the areas that the neural network is used the most?

Reading questions: Relationship to linguistics

- Is the relative power of neural language models an argument for connectionist models of language acquisition or cognition?