Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science

6.036 Introduction to Machine Learning

Homework 4 Solutions

1 Neural networks

1. To obtain the boundaries, substitute the values of W for $z_1, z_2 = 0$.

$$\begin{aligned} z_1 &= x_1 W_{11} + x_2 W_{21} + W_{01} \\ 0 &= x_1 + x_2 - 1 \\ x_2 &= -x_1 + 1 \end{aligned}$$

$$\begin{aligned} z_2 &= x_1 W_{12} + x_2 W_{22} + W_{02} \\ 0 &= x_1 - x_2 - 1 \\ x_2 &= x_1 - 1 \end{aligned}$$

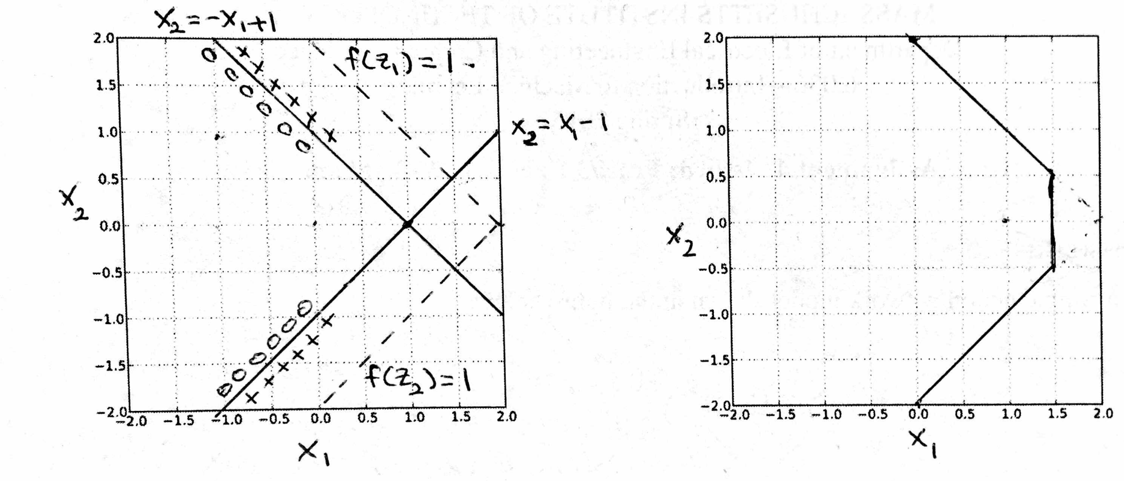
See boundaries in figure. To calculate which side is 0 and which side is +, substitute a point such as $[x_1, x_2]^T = [0, 0]^T$ into each formula and observe the values of z_1, z_2 to assign 0 or + to the region where that point lies.

2. Calculate the new boundaries:

$$\begin{aligned} z_1 &= x_1 W_{11} + x_2 W_{21} + W_{01} \\ 1 &= x_1 + x_2 - 1 \\ x_2 &= -x_1 + 2 \end{aligned}$$

$$\begin{aligned} z_2 &= x_1 W_{12} + x_2 W_{22} + W_{02} \\ 1 &= x_1 - x_2 - 1 \\ x_2 &= x_1 - 2 \end{aligned}$$

See dashed boundaries in figure.



3. Point $[2, 0]^T$ belongs to $f(z_1) = 1$ and $f(z_2) = 1$ (you can see this clearly in figure), therefore $[f(z_1), f(z_2))^T = [1, 1]^T$.

4. Calculate the boundary for each region:

$$\begin{aligned}
\text{Region 1} &: \quad f(z_1) = 0, f(z_2) = 0 \Rightarrow \text{ no boundary} \\
\text{Region 2} &: \quad f(z_1) > 0, f(z_2) = 0 \Rightarrow 0 = f(z_1) - 1 \Rightarrow x_2 = -x_1 + 2 \\
\text{Region 3} &: \quad f(z_1) = 0, f(z_2) > 0 \Rightarrow 0 = f(z_2) - 1 \Rightarrow x_2 = x_1 - 2 \\
\text{Region 4} &: \quad f(z_1) > 0, f(z_2) > 0 \Rightarrow 0 = f(z_1) + f(z_2) - 1 \Rightarrow x_1 = 1.5
\end{aligned}$$

2 Recursive Neural Networks

1.

$$z_3^t = f(x_1 * W_{12} + x_2 * W_{22} + f(z_3^{t-1}) * U_2) \qquad (1)$$

$$z_4^t = f(x_1 * W_{11} + x_2 * W_{21} + f(z_4^{t-1}) * U_1) \qquad (2)$$

2. The output sequence is 4, 2, 2. This can be seen by plugging the values of x_1 and x_2 into the equations from part a.

3. There are many possible answers that could work here. Any interesting solution could be $U_1 = U_2 =$ -100. This effectively forces the output from the second layer to 0 any time z_3 or z_4 are equal to 1. For the example above, the new output would be 0, 0, 0.

3 Back-propagation

In each step of the back-propogation derivation, we first set up the chain rule to compute each derivative, then compute each term.

1.

$$\begin{align}
\delta_{3j} &= \left[\frac{\partial}{\partial z^{(t)}}\mathrm{Loss}\big(y^{(t)} z^{(t)}\big)\right]\left[\frac{\partial z^{(t)}}{\partial f(z_{3j}^{(t)})}\right]\left[\frac{\partial f(z_{3j}^{(t)})}{\partial z_{3j}^{(t)}}\right] \tag{3}\\
&= \left[\begin{array}{l} -y^{(t)} \text{ if } \mathrm{Loss}\big(y^{(t)} z^{(t)}\big) > 0 \\ \text{and zero otherwise} \end{array}\right][V_j]\,\llbracket\, z_{3j}^{(t)} > 0\,\rrbracket \tag{4}
\end{align}$$

2.

$$\begin{align}
\frac{\partial}{\partial W_{3;ij}}\mathrm{Loss}(yF(x;\theta)) &= \left[\frac{\partial}{\partial z_{3j}^{(t)}}\mathrm{Loss}(yF(x;\theta))\right]\left[\frac{\partial z_{3j}^{(t)}}{\partial W_{3;ij}}\right] \tag{5}\\
&= \delta_{3j}\, f(z_{2i}) \tag{6}
\end{align}$$

3.

$$\begin{align}
\delta_{2i} &= \sum_{j=1}^{m}\left[\frac{\partial}{\partial z_{3j}^{(t)}}\mathrm{Loss}\big(y^{(t)} z^{(t)}\big)\right]\left[\frac{\partial z_{3j}^{(t)}}{\partial f(z_{2i}^{(t)})}\right]\left[\frac{\partial f(z_{2i}^{(t)})}{\partial z_{2i}^{(t)}}\right] \tag{7}\\
&= \sum_{j=1}^{m}[\delta_{3j}]\,[W_{3;ij}]\,\llbracket\, z_{2j}^{(t)} > 0\,\rrbracket \tag{8}\\
& \tag{9}
\end{align}$$

4.

$$\begin{align}
\delta_{1j} &= \sum_{j=1}^{m}\left[\frac{\partial}{\partial z_{2j}^{(t)}}\mathrm{Loss}\big(y^{(t)} z^{(t)}\big)\right]\left[\frac{\partial z_{2j}^{(t)}}{\partial f(z_{1i}^{(t)})}\right]\left[\frac{\partial f(z_{1i}^{(t)})}{\partial z_{1i}^{(t)}}\right] \tag{10}\\
&= \sum_{j=1}^{m}[\delta_{2j}]\,[W_{2;ij}]\,\llbracket\, z_{1i}^{(t)} > 0\,\rrbracket \tag{11}\\
& \tag{12}
\end{align}$$

5.

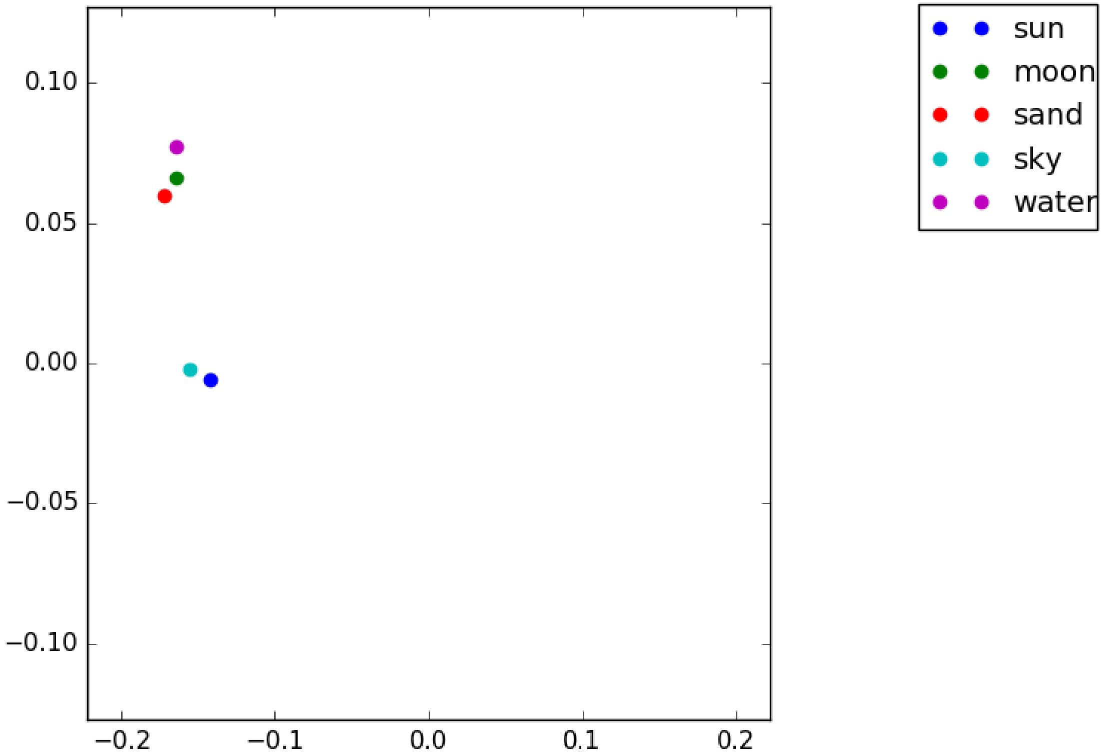
$$\frac{\partial}{\partial W_{1;ij}} \mathrm{Loss}(yF(x;\theta)) = \sum_{j=1}^{m} \left[\frac{\partial}{\partial z_{1j}^{(t)}} \mathrm{Loss}(yF(x;\theta)) \right] \left[\frac{\partial z_{1j}^{(t)}}{\partial W_{1;ij}} \right] \tag{13}$$

$$= \sum_{j=1}^{m} \delta_{1j}\, x_i^{(t)} \tag{14}$$

6. Combining our back-propogation from the previous parts we compute our SGD update rule. Whenever $x^{(t)}$ isn't classified sufficiently well, i.e., $\mathrm{Loss}\big(y^{(t)} z^{(t)}\big) > 0$,

$$\begin{aligned}
W_{3;ij} &\leftarrow W_{3;ij} - \eta_k\, f(z_{2i}^{(t)})\delta_{3j}, \;\; i = 1, \ldots, d,\; j = 1, \ldots, m \\
W_{2;ij} &\leftarrow W_{3;ij} - \eta_k\, f(z_{1i}^{(t)})\delta_{2j}, \;\; i = 1, \ldots, d,\; j = 1, \ldots, m \\
W_{2;ij} &\leftarrow W_{3;ij} - \eta_k\, x_i^{(t)}\delta_{1j}, \;\; i = 1, \ldots, d,\; j = 1, \ldots, m \\
V_j &\leftarrow V_j + \eta_k y^{(t)} f(z_{3j}^{(t)}), \;\; j = 1, \ldots, m
\end{aligned}$$

4 (Word) Embeddings

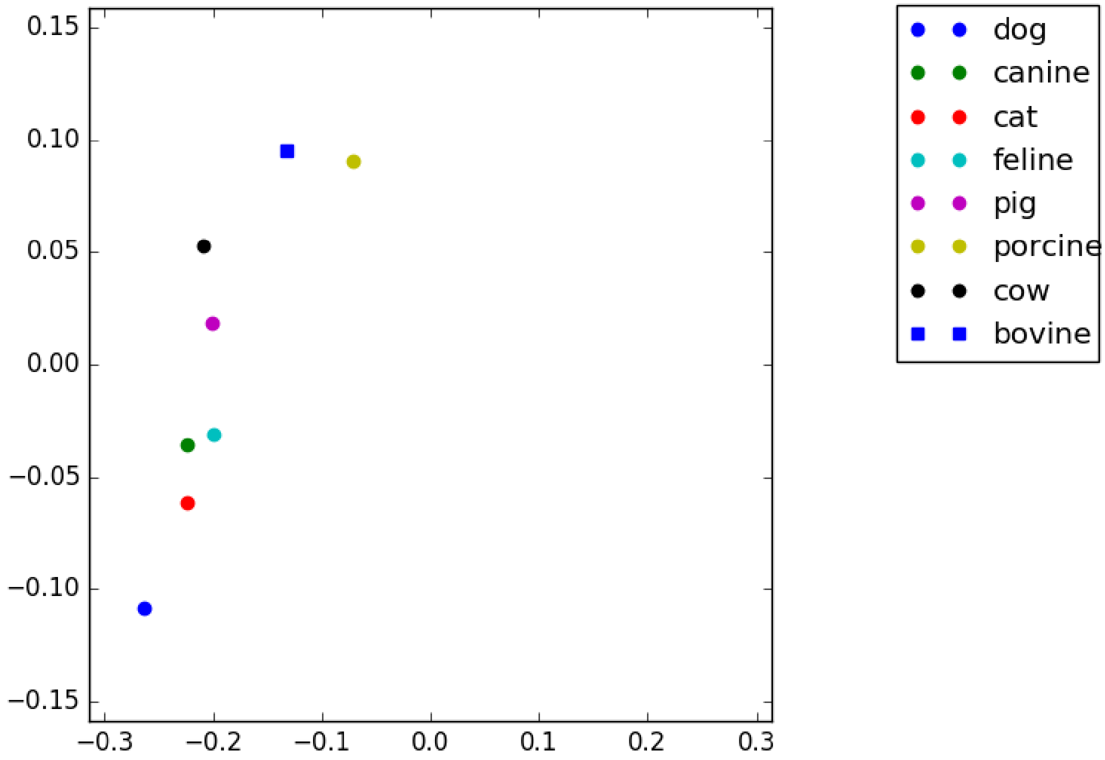


1.

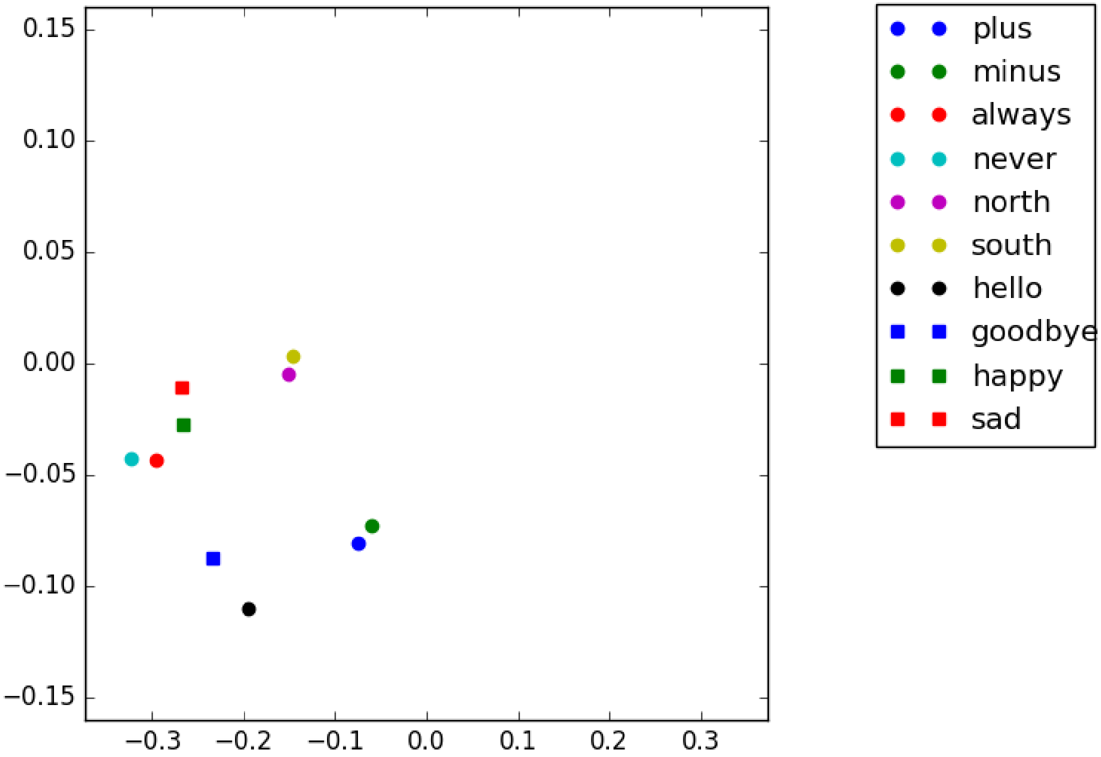
2. The first network will produce a random output. As a result of the words not being seen during training, the corresponding input units always had an output of zero, which means that no gradient was backpropagated into the weights and the randomly-initialized weights remain unchanged.

   The second network, however, has a fighting chance at classifying the sequence. Since the embedding process causes words that appear in similar contexts to have similar locations in vector space, the unseen words will likely fall into a region that contains words that the network *has* seen and knows how to correctly classify.

3. The difference vectors all have about the same magnitude and direction!



4. The words in the pairs, though opposites, are located quite close to each other since they appear in similar contexts. A network that depends on each word's valence might have a hard time differentiating between the words in each pair since the resulting decision boundary would have a very small margin and need to be highly nonlinear. Some solutions to this problem include adding more layers and training your own embeddings from scratch.


0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
−0.3
−0.2
−0.1
0.0
0.1
0.2
0.3
plus
minus
always
never
north
south
hello
goodbye
happy
sad