Table 1: Truth Table for Problem 1(a)

	h_1	h_2	h_3	h_4
A	0	0	0	0
B	0	0	1	0
C	0	1	1	0
D	1	1	1	0
E	1	1	1	1
F	1	1	0	1
G	1	0	0	1
H	0	0	0	1
I	0	0	1	1
J	0	1	1	1
K	0	1	0	1

Table 2: Truth Table for Problem 1(c) part 1

	h_1	h_2	h_3	h_4	$\sigma(\cdot)$
A	0	0	0	0	0
B	0	0	1	0	0
C	0	1	1	0	0
D	1	1	1	0	1
E	1	1	1	1	0
F	1	1	0	1	0
G	1	0	0	1	0
H	0	0	0	1	0
I	0	0	1	1	0
J	0	1	1	1	0
K	0	1	0	1	0

1(a)

(b) An example of a bit pattern that is missing is 1,0,1,0. We know that there are more bit patterns missing as the cardinality of the power set of our 4 bits is $(2^4) = 16$ and we only have 11 possibilities.

(c)

1. $y \approx 1$ if $x \in D$, $y \approx 0$ if $x \notin D$

   Logical Function: $h_1 \wedge h_2 \wedge h_3 \wedge \neg h_4$

   $w_1 = 20$

   $w_2 = 20$

   $w_3 = 20$

   $w_4 = -20$

   $b = -50$

Table 3: Truth Table for Problem 1(c) part 2

	h_1	h_2	h_3	h_4	$\sigma(\cdot)$
A	0	0	0	0	1
B	0	0	1	0	1
C	0	1	1	0	1
D	1	1	1	0	1
E	1	1	1	1	1
F	1	1	0	1	1
G	1	0	0	1	1
H	0	0	0	1	1
I	0	0	1	1	1
J	0	1	1	1	0
K	0	1	0	1	1

2. $y \approx 0$ if $y \in J$, $y \approx 1$ if $y \notin J$

   Logic Function: $\neg(\neg h_1 \wedge h_2 \wedge h_3 \wedge h_4)$

   $w_1 = 20$

   $w_2 = -20$

   $w_3 = -20$

   $w_4 = -20$

   $b = 50$

2. Setting $f_1(z) = z$ is a terrible choice for a neural network because it defeats the purpose of performing non-linearity, which is solving the non-linearity involved in some problems such as classification of pixel images that have numerous properties like color, rotation, illumination and so forth which can be much harder to do with a linear model. Also, there are possible issues with back propagation as gradients get smaller/vanish as slope goes to 0.

3. We need the predictive power and expressivity of the model to match the complexity of the problem we are solving. Neural networks rely on the ability to properly identify the number of hidden units, but the catch is there is no guarantee we actually train the model sufficiently from a finite training set, unless the set is enormous and if the set is too small we expose ourselves to bias from low expressivity. Even though Neural Nets are universal approximators we need to match the complexity of the net to the problem we are solving to get a good approximation/ model.

4.

- Net 1:

  $f_1 = \text{ReLU}$
  $f_2 = \text{Softmax}$
  $L = 64$
  $C = 10$

- Net 2:

  $f_1 = \text{Sigmoid}$
  $f_2 = \text{Sigmoid}$
  $L = 128$
  $C = 1$

- Net 3:

  $f_1 = \tanh$
  $f_2 = \text{identity}$
  $L = 256$
  $C = 1$

- Net 4:

  $f_1 = \text{ReLU}$
  $f_2 = \text{Softmax}$
  $L = 32$
  $C = 100$