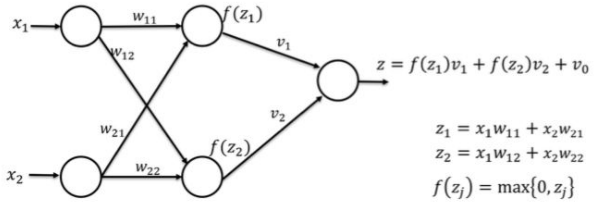
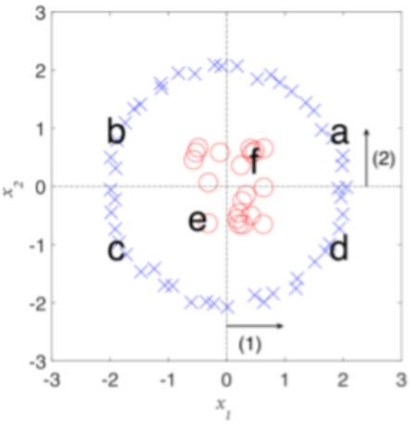
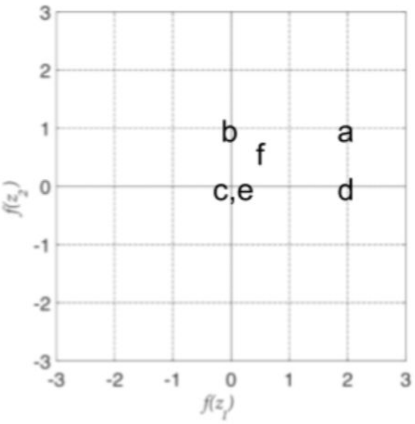
PROBLEM 13

Problem 4 Consider a 2-layer feed-forward neural network that takes in $x \in \mathbb{R}^2$ and has two ReLU hidden units. *Note that the hidden units have no offset parameters.*


x_1
x_2
w_{11}
w_{12}
w_{21}
w_{22}
$f(z_1)$
$f(z_2)$
v_1
v_2
$z = f(z_1)v_1 + f(z_2)v_2 + v_0$
$z_1 = x_1 w_{11} + x_2 w_{21}$
$z_2 = x_1 w_{12} + x_2 w_{22}$
$f(z_j) = \max\{0, z_j\}$

(4.1) **(6 points)** The values of the weights in the hidden layer are set such that they result in the z_1 and z_2 "classifiers" as shown in the figure by the decision boundaries and the corresponding normal vectors marked as (1) and (2). Approximately sketch on the right how the input data is mapped to the 2-dimensional space of hidden unit activations $f(z_1)$ and $f(z_2)$. Only map points marked 'a' through 'f'. Keep the letter indicators.





(4.2) **(2 points)** If we keep these hidden layer parameters fixed but add and train additional hidden layers (applied after this layer) to further transform the data, could the resulting neural network solve this classification problem? (Y/N) **(** N **)**

Reasoning: Since points c and e are mapped to the origin but have opposite label, it is impossible to classify them both correctly.

(4.3) **(3 points)** Suppose we stick to the 2-layer architecture but add lots more ReLU hidden units, all of them without offset parameters. Would it be possible to train such a model to perfectly separate these points? (Y/N) **(** Y **)**

(4.4) **(3 points)** Initialization of the parameters is often important when training large feed-forward neural networks. Which of the following statements is correct? Check T or F for each statement.

(T **)** If we use tanh or linear units and initialize all the weights to very small values, then the network initially behaves as if it were just a linear classifier

(T **)** If we randomly set all the weights to very large values, or don't scale them properly with the number of units in the layer below, then the tanh units would behave like sign units.

(T **)** A network with sign units cannot be effectively trained with stochastic gradient descent

(4.5) **(3 points)** There are many good reasons to use convolutional layers in CNNs as opposed to replacing them with fully connected layers. Please check T or F for each statement.

(T **)** Since we apply the same convolutional filter throughout the image, we can learn to recognize the same feature wherever it appears.

(T **)** A fully connected layer for a reasonably sized image would simply have too many parameters

(F **)** A fully connected layer can learn to recognize features anywhere in the image even if the features appeared preferentially in one location during training

Additional explanation:

13)

a) The new mapping comes from applying the given calculations. Note that points c and e, with opposite label, are both mapped to the origin.

b) Since points e and c both map to the origin, additional layers on top of the current mapping would not be able to provide linear separability.

c) Keeping a 2-layer architecture but with more hidden units implies retraining the current model, so it would be possible to perfectly separate the points with sufficient complexity.

d) All true. Using linear or tanh units with small initializations behaves like a linear classifier because both functions are linear or approximately linear near 0. We can get tanh units to behave like sign units with large differences in scaling. A network with sign units can't be effectively trained with SGD because it doesn't have a nice derivative.

e) True, True, and False. Using the same convolutional filter applied in different parts of the image allows us to recognize similar features. CNNs give the advantage of parameter sharing, whereas a fully connected layer would have an impractical number of parameters to train. Importantly, a fully connected layer would learn different parameters for different parts of the image, so bias toward features in particular locations could be introduced in training.