Q1. Generalization

(i) Suppose you train a classifier and test it on a held-out validation set. It gets 80% classification accuracy on the training set and 20% classification accuracy on the validation set.

From what problem is your model most likely suffering?

○ Underfitting ● Overfitting

Fill in the bubble next to any measure of the following which could reasonably be expected to improve your classifier's performance on the validation set.

● Add extra features ● Remove some features

Briefly justify: Either answer was accepted with justification. Add extra features – adding some really good features could better capture the structure in the data. Remove some features – the model may be using the noise in the abundant feature set to overfit to the training data rather than learning any meaningful underlying structure.

● Collect more training data ○ Throw out some training data

More data should yield a more representative sample of the true distribution of the data. Less data is more susceptible to overfitting.

Assuming features are outcome counts (k is the Laplace smoothing parameter controlling the number of extra times you "pretend" to have seen an outcome in the training data):

● Increase k ○ Decrease k (assuming $k > 0$ currently)

Increasing k reduces the impact of any one training instance to make the classifier less sensitive to overfitting of rare (= low count) patterns.

Assuming your classifier is a Bayes' net:

○ Add edges ● Remove edges

Removing edges reduces the class of distributions the Bayes' net can represent. Adding edges introduces more parameters so that the model could further overfit.

(ii) Suppose you train a classifier and test it on a held-out validation set. It gets 30% classification accuracy on the training set and 30% classification accuracy on the validation set.

From what problem is your model most likely suffering?

● Underfitting ○ Overfitting

Fill in the bubble next to any measure of the following which could reasonably be expected to improve your classifier's performance on the validation set.

● Add extra features ○ Remove some features

Briefly justify: Under the current feature representation, we are unable to accurately model the training data for the purpose of the classification task we're interested in. The classifier may be able to deduce more information about the connections between data points and their classes from additional features, allowing it to better model the data for the classification task. For example, a linear perceptron could not accurately model two classes separated by a circle in a 2-dimensional feature space, but by using quadratic features in a kernel perceptron, we can find a perfect separating hyperplane.

● Collect more training data ○ Throw out some training data

More training data can only be a good thing. Marking neither of the bubbles was accepted, too, as given that train and hold-out validation already achieve the same performance, likely the underlying problem is not a lack of training data.

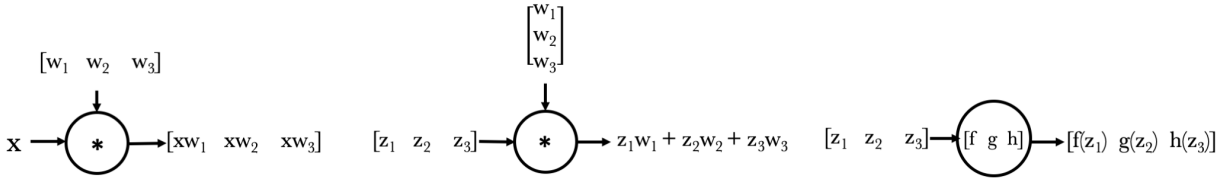
(iii) Your boss provides you with an image dataset in which some of the images contain your company's logo, and others contain competitors' logos. You are tasked to code up a classifier to distinguish your company's logos from competitors' logos. You complete the assignment quickly and even send your boss your code for training the classifier, but your boss is furious. Your boss says that when running your code with images and a random label for each of the images as input, the classifier achieved perfect accuracy on the training set. And this happens for all of the many random labelings that were generated.

Do you agree that this is a problem? Justify your answer.

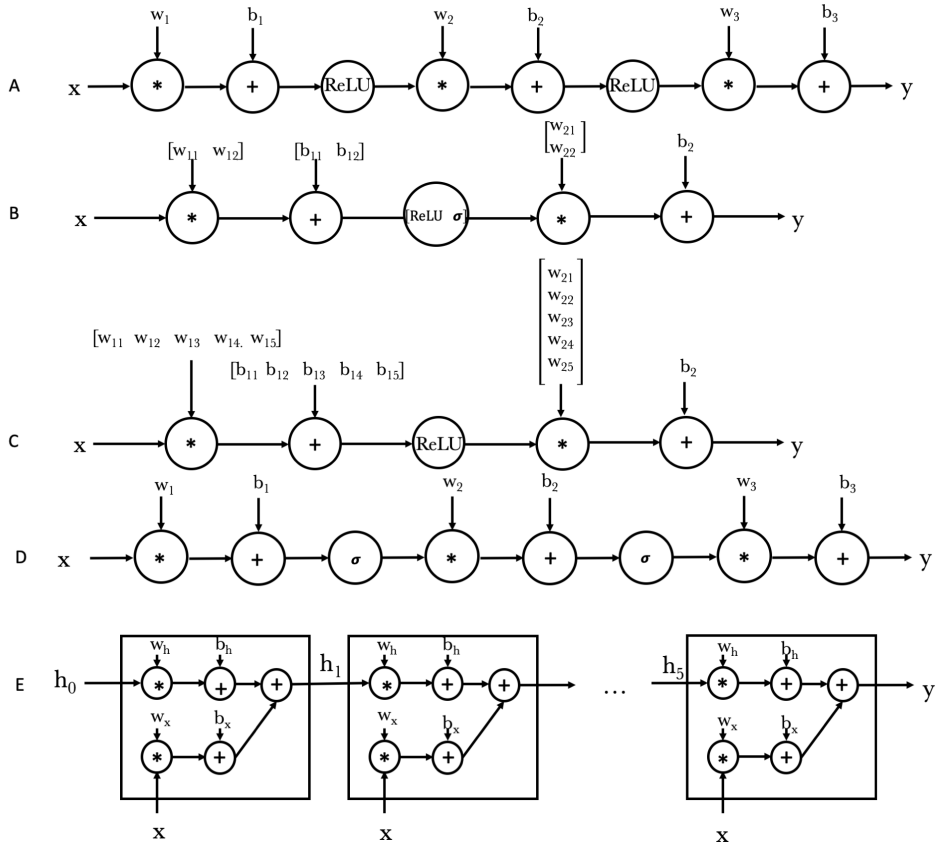
Yes, this is a problem. The classifier is overfitting the training set. The fact that it had perfect accuracy with random labels suggests that it does not learn any real underlying structure in the data; it most likely essentially memorized each of the training cases.

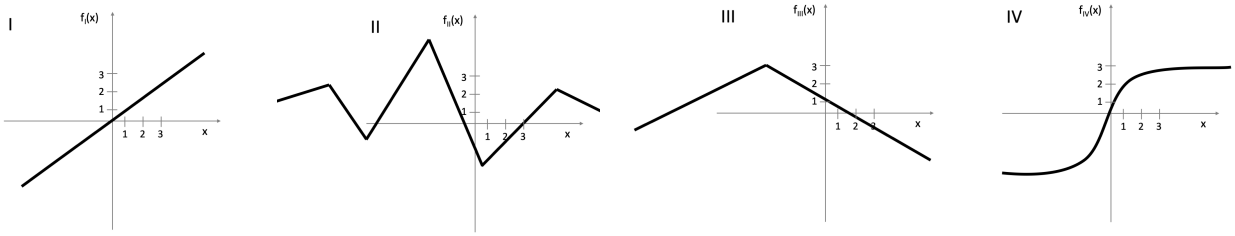
Q2. Short Questions

(a) We are given the following 5 neural networks (NN) architectures. The operation $*$ represents the matrix multiplication operation, $[w_{i1}...w_{ik}]$ and $[b_{i1}...b_{ik}]$ represents the weights and the biases of the NN, the orientation (vertical and horizontal) its just for consistency in the operations. The term [ReLU σ] in **B** means applying a ReLU activation to the first element of the vector and a sigmoid (σ) activation to the second element. These operations are depicted in the following figures:



Which of the following neural networks can represent each function?





(i) $f_I(x)$: ☐ A ☐ B ■ C ☐ D ■ E

A and B cannot represent this plot since the ReLU activation results in a flat semi-line. C can represent it by having $w_{11} = 1, w_{12} = -1, w_{21} = 1, w_{22} = 1$ and the rest of the parameters being 0. D cannot because the sigmoid activations are non-linear. E is a linear function and therefore can represent the identity.

(ii) $f_{II}(x)$: ☐ A ☐ B ■ C ☐ D ☐ E This is a piecewise linear function with 6 pieces. As a result you can represent it by lineraly combining 5 ReLU functions. The only possible graph that can represent this function is C.

(iii) $f_{III}(x)$: ☐ A ☐ B ■ C ☐ D ☐ E This is a piecewise linear function with 2 pieces. This can be obtained by linearly combining to ReLU functions. The only possible solution is then C. Note that A cannot represent this function since the last ReLU of the network would result in a flat semi-line.

(iv) $f_{IV}(x)$: ☐ A ■ B ☐ C ☐ D ☐ E This function corresponds to a scaled sigmoid function. A, C, E cannot represent any non-linear or non-piecewise linear functions. B can represent it by setting $w_{11} = b_{11} = 0$. D does not work because the composition of sigmoid with sigmoid is not a sigmoid.