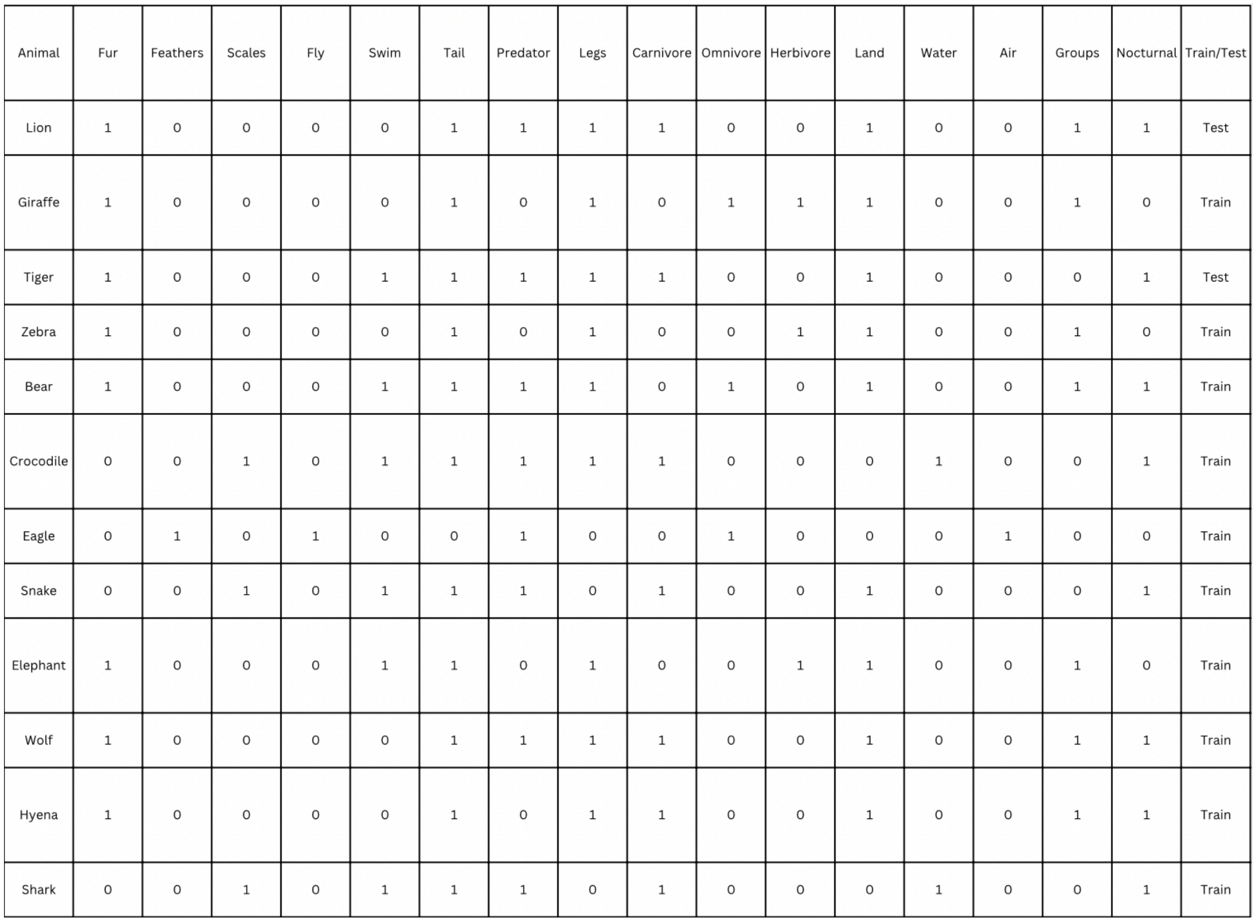
Q1 [10 pts.] Hand in a table displaying the set of patterns you have constructed, and explain how you designed them. Be sure to identify which are withheld for testing.



Animal	Fur	Feathers	Scales	Fly	Swim	Tail	Predator	Legs	Carnivore	Omnivore	Herbivore	Land	Water	Air	Groups	Nocturnal	Train/Test
Lion	1	0	0	0	0	1	1	1	1	0	0	1	0	0	1	1	Test
Giraffe	1	0	0	0	0	1	0	1	0	1	1	1	0	0	1	0	Train
Tiger	1	0	0	0	1	1	1	1	1	0	0	1	0	0	0	1	Test
Zebra	1	0	0	0	0	1	0	1	0	0	1	1	0	0	1	0	Train
Bear	1	0	0	0	1	1	1	1	0	1	0	1	0	0	1	1	Train
Crocodile	0	0	1	0	1	1	1	1	1	0	0	0	1	0	0	1	Train
Eagle	0	1	0	1	0	0	1	0	0	1	0	0	0	1	0	0	Train
Snake	0	0	1	0	1	1	1	0	1	0	0	1	0	0	0	1	Train
Elephant	1	0	0	0	1	1	0	1	0	0	1	1	0	0	1	0	Train
Wolf	1	0	0	0	0	1	1	1	1	0	0	1	0	0	1	1	Train
Hyena	1	0	0	0	0	1	0	1	1	0	0	1	0	0	1	1	Train
Shark	0	0	1	0	1	1	1	0	1	0	0	0	1	0	0	1	Train

(Table made on Canva)

The input features were based on physical traits and actions, they were whether the animals had these traits: fur, feathers, scales, fly, swim, tail, predator, legs, this was to ensure the animals had some overlap in shared features with similar attributes so the input patterns could be similar but also a little different. The target features were based on habitat, diet, and activity patterns, and they were carnivore, omnivore, herbivore, land, water, air, groups, nocturnal. I designed the set of patterns to have an equal amount of generalizability and learnability, based on distributed representations where each pattern is represented across most units and features apply to multiple animals rather than localist representations, so the network can get shared features and unique ones. Also the design avoids identical input patterns which prevents catastrophic interference, so basically no 2 animals share the exact same features or else two inputs mapped would lead to different outputs. I chose the lion and tiger as the test cases because those 2 animals are similar in the wild but still differ in specific individual features and I thought those two cases were likely to show at least some degree of generalization based on the nature of the remaining training examples. Then there will be evaluation of the model's ability to generalize knowledge to new but similar examples needed for interleaved learning and stability. Later the network will learn distinguished exceptions while obtaining the systematic animal classifying parameters by interleaving different species, meeting the need to integrate systematic and idiosyncratic information.

Q2 [20 pts.] Examine how well the network did at the end of training in *learning the training examples* (i.e., how well the generated activations match the targets for each trained pattern), and explain the successes and failures (based on the relationships among the trained patterns).

After clicking the train network 4 times, the error graph seems to have stabilized and isn't changing much since the weight adjustments become minor and they gradually reduce error until convergence. At the end of training, the network successfully learned most training examples, from the similar match between generated activations and target outputs in the Unit Viewer. For example, carnivorous animals like the crocodile and wolf had high activations, 0.98 and 0.62 respectively for the carnivore unit but low for the herbivore unit 0.02 and 0.01 respectively, while herbivorous animals like the giraffe and zebra had strong activation for herbivore, 0.67 for both but weak activations for carnivore, 0.23 for both. The successes happened due to shared patterns from multiple examples, and consistent input-output pairings which reinforces accurate classification. Distributed representations occur since overlapping features help correct learning. There were also some failures in generalization such as the giraffe only showing a weaker activation 0.08 in the omnivore unit despite the fact that it eats insects occasionally, signifying some uncertainty in classification from terminological ambiguity so basically the network mainly relies on the dominant and similar features for the giraffe, omnivore features was more prevalent for other animals like bears and eagles.

Q3 [15 pts.] Examine the *time course of learning*. To do this, reset the network and then retrain it only a few epochs at a time (using the "Weight Updates" setting on the main console). Try to identify what aspects of your patterns the network learns first and what aspects it learns only later. Describe what you observe and try to explain why it happens.

At the start of training, weight updates = 5 → 10, the aspects of the patterns that the network learns first are strong activations for whether an animal lives on land, lives in a group, and is nocturnal those seem the most prevalent amongst the training examples as the extent to which the generated activation for each output unit (the center of the square) matches its target (the outside ring) is highest for majority of the examples here, output units 3, 6, 7. This is because these features are shared by multiple animals, making them easier for the network to generalize quickly based on input. Then afterwards from weight updates 15 → 20, the aspects it learns only later are omnivore and those specific or ambiguous habitat structures from the unit viewer, over several training epochs, the hidden unit activations are gradually improved. This is because of distributed representation principles refining the habitat, diet, and activity pattern categories by a gradual learning trajectory. The features learned later include giraffes' partial omnivore feature, probably needing more precise weight modifications because of overlapping or unclear feature patterns. The unit viewer shows how the network works on the more distinguishable characteristics through weight modifications and backpropagation after first learning the dominating and common features.

Q4 [25 pts.] Now consider how well the trained network *generalizes* to the two patterns that you set aside, again comparing the generated and target activations. (To do this, open the Unit Viewer and select "Testing Set" from the "Example Set" menu, and then click on the examples.) Report what happens when you test with these, and explain the results in terms of the relationship between the test patterns and the trained patterns.

When testing the lion and tiger examples, the Unit Viewer shows that their generated activations closely align with trained carnivores like wolves and bears, meaning the network successfully generalized key features such as being a land predator that is carnivore. The lion's activations for carnivore related units are fairly strong, but the group unit is a little misaligned, the colors aren't the same, probably because of overlap with social predators like hyenas, even though lions live in prides. Not all the land predators have completely similar features so the lions can overlap with some that aren't entirely equal in features to it. Also the group feature of tigers was fairly active even though tigers live/hunt alone so it was misclassified probably due to limited overlap with trained examples as the network learns shared patterns but struggles with exceptions so hidden unit activations probably grouped tigers with wolves since they live in packs which reinforced an incorrect mapping. So the hidden unit activation patterns had overlap amongst different related animals especially with the land predator carnivores as expected from lions and tigers. Overall, the classification failures are mainly from specificity and generalization based on shared features within ambiguous or underrepresented data and as neural networks value distributed representations.

Q5 [30 pts.] How well does the new version of the network learn the training patterns, and how well does it generalize to the test patterns? Are there any differences in how the network performs compared to the original version, and why?

My original network succeeded at learning the training examples fully so I introduced weight decay by editing the 20th line in 8-n-8_in.txt. The new version of the network with weight decay = 0.005 also successfully learns the training patterns again when looking at the Unit Viewer diagram as the activations align with the target activations across the training set. On the testing set though, the lion and tiger examples, the network had some misalignment in the group and nocturnal units meaning a weaker generalization compared to the original version. By avoiding overfitting to the training set, weight decay basically regulates the network, producing more different representations while decreasing accuracy on new inputs. The new version promotes the overall broad category properties for exact matches, leading to lower performance but increase in generalization of results on the data that was set aside compared to the original network, so overall better memorized training samples.