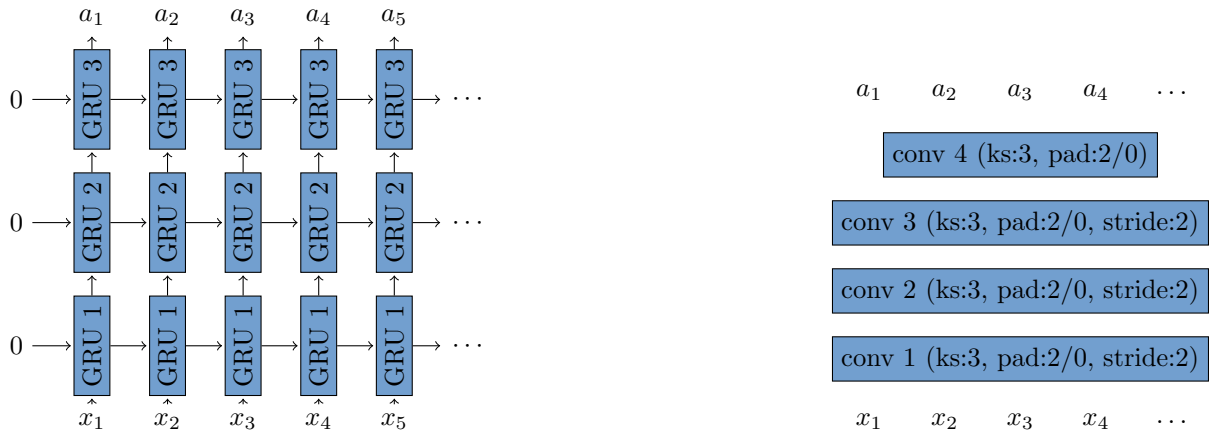
Exercise 9: Temporal models - Solution

Name: ..... UTID: .....

Let's go play some video games. In this class, we'll train deep networks to play video games. Why? Because even deep networks deserve a break to blow off some steam.

In a video game your inputs are individual images x_t for each time step t. The output of your network is an action a_t (e.g. go left, jump, duck, ...) for each time step t.

You're trying to decide between two different network architectures: A recurrent network built out of GRUs and a temporal convolutional network. Both networks are shown below



The padding in the temporal CNN is one sided (two elements in the past, none in the future). Furthermore, we evaluate the strided convolutions at every time-step t (instead of skipping half of them in striding). This leads to increased computation, but ensures we predict a unique action at every timestep. This is also known as holes or dilation. You do not need to fully understand this to complete the exercise.

To better decide which model to use, let's look at some of their properties:

a) What is the longest temporal dependency (distance $t' - t$ such that an input at time t influences an output at t') that each model captures? (An approximate number is sufficient.)

RNN: ∞
CNN: $1 + 2 * 8 + 2 * 4 + 2 * 2 + 2 * 1 = 31$

b) How long is the longest path between input and output in each network (how many layers does it cross)? How does is scale with the length of the sequence. (An approximate number is sufficient.)

RNN: ∞, scales linearly with length of sequence (or dependency)
CNN: 4, scales logarithmically with length of sequence (or dependency)

c) Which model would you chose, justify why

CNN as the longest dependency grows exponentially in the longest path between input and output. Hence, we get fewer vanishing gradients.

d) Now that you settled on a network you decide to train it. You decide to train the network using imitation learning. In imitation learning the network observes a human player and learns to predict exactly the same outputs as the human (using a cross entropy loss). Is imitation learning a good strategy to learn an agent? What are some of the advantages and disadvantages of imitation learning:

Keep in mind our ultimate goal is to train an agent to play a game well.

- ☐ Suffers from vanishing gradients
- ☒ Uses all the theory and machinery from supervised learning
- ☐ Outperforms a human player given enough training data
- ☒ Can get confused if trained using multiple different strategies or behaviors
- ☒ The training and evaluation loss do not perfectly align
- ☒ Easy to implement

This and next week, we will learn a few other strategies to train an agent without using any human observations.