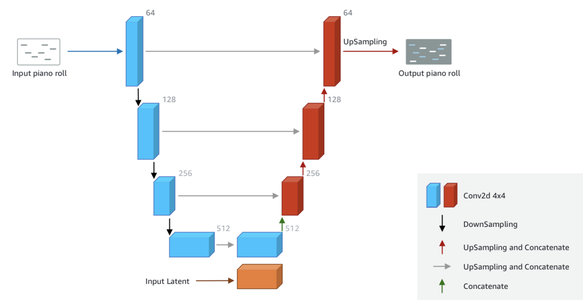
How the Model Works

The model consists of two networks, a generator and a critic. These two networks work in a tight loop:

- The generator takes in a batch of single-track piano rolls (melody) as the input and generates a batch of multi-track piano rolls as the output by adding accompaniments to each of the input music tracks.
- The discriminator evaluates the generated music tracks and predicts how far they deviate from the real data in the training dataset.
- The feedback from the discriminator is used by the generator to help it produce more realistic music the next time.
- As the generator gets better at creating better music and fooling the discriminator, the discriminator needs to be retrained by using music tracks just generated by the generator as fake inputs and an equivalent number of songs from the original dataset as the real input.
- We alternate between training these two networks until the model converges and produces realistic music.

The discriminator is a **binary classifier** which means that it classifies inputs into two groups, e.g. "real" or "fake" data.



Defining and Building Our Model

9. Run the cell that defines the generator
10. Run the cell that builds the generator
11. Run the cell that defines the discriminator
12. Run the cell that builds the discriminator

Model Training and Loss Functions

As the model tries to identify data as "real" or "fake", it's going to make errors. Any prediction different than the ground truth is referred to as an error.

The measure of the error in the prediction, given a set of weights, is called a **loss function**. Weights represent how important an associated feature is to determining the accuracy of a prediction.

Loss functions are an important element of training a machine learning model because they are used to update the weights after every iteration of your model. Updating weights after iterations optimizes the model making the errors smaller and smaller.

Setting Up and Running the Model Training

13. Run the cell that defines the loss functions
14. Run the cell to set up the optimizer
15. Run the cell to define the generator step function
16. Run the cell to define the discriminator step function
17. Run the cell to load the melody samples
18. Run the cell to set the parameters for the training
19. Run the cell to train the model!!!!

Training and tuning models can take a very long time – weeks or even months sometimes. Our model will take around an hour to train.

Model Evaluation

Now that the model has finished training it's time to evaluate its results.

There are several evaluation metrics you can calculate for classification problems and typically these are decided in the beginning phases as you organize your workflow.

In our example we:

- Checked to see if the losses for the networks are converging
- Looked at commonly used musical metrics of the generated sample and compared them to the training dataset.

Evaluating Our Training Results

20. Run the cell to restore the saved checkpoint. If you don't want to wait to complete the training you can use data from a pre-trained model by setting `TRAIN = False` in the cell.
21. Run the cell to plot the losses.
22. Run the cell to plot the metrics.

Results and Inference

Finally, we are ready to hear what the model produced and visualize the piano roll output!

Once the model is trained and producing acceptable quality, it's time to see how it does on data it hasn't seen. We can test the model on these unknown inputs, using the results as a proxy for performance on future data.

Evaluate the Generated Music

23. In the first cell, enter `0` as the iteration number:

```
iteration = 0
```

run the cell and play the music snippet.
Or listen to this example snippet from iteration 0:

0:01 / 0:16

24. In the second cell, enter `0` as the iteration number:

```
iteration = 0
```

run the cell and display the piano roll.

iteration: 0