1. Write the sampling program. Explain in words the effect of temperature parameter and the concept of logit and sigmoid functions.

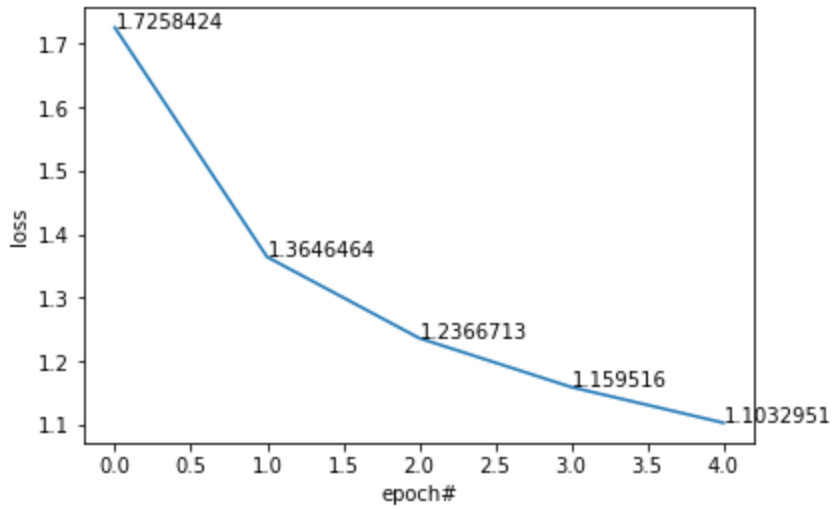
The temperature parameter is used to control the randomness (diversity) in the music generation. When we divide the activation values by the temperature parameter T, this can make the selection of next character conservation or random. If the value of T is small, the selection is conservative while a higher T leads to the generation of diverse (often random) tunes.
Logits is the unnormalized final scores of an ML model. In this case, we apply softmax to it to get a probability distribution over different classes (predicting the next character in the sequence). The softmax function then generates a vector of (normalized) probabilities with one value for each possible class.
Sigmoid is the activation function that that is often used for binary classification. The sigmoid function is represented as $1/(1 + e^{-z})$, where z is the product of weight (w_i) and input (x_i). The idea behind using sigmoid activation function is to introduce non-linearity in the ML predictions. The sigmoid function maps the product in the range of (0,1).

2. Run the program (learning and generation) with different parameters (temperature, sentence length, etc...). Can you report the effects? At what temperature levels the music tends to become more or less repetitive? How does the sentence length affect the result?

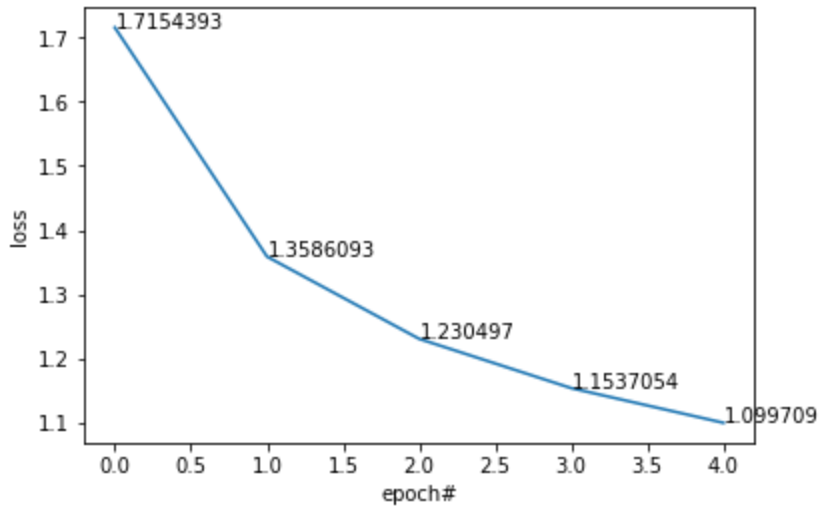
The music becomes more repetitive for smaller values of temperature (close to 0). This is because the probability for the next music char is reinforced after dividing by a smaller temperature value. I could see the music turning repetitive for T = 0.2. The inverse is true if we increase the value of T (close to 1). This brings randomness in the music generated.
For smaller sentence length, the RNN formed is of smaller length. This means the parameters are reused more often and the music generated is repetitive. The inverse is true for longer sequences.

3. Report some tracking of the error progress using either matplotlib or tensorboard (Plotting and visualization libraries in python)

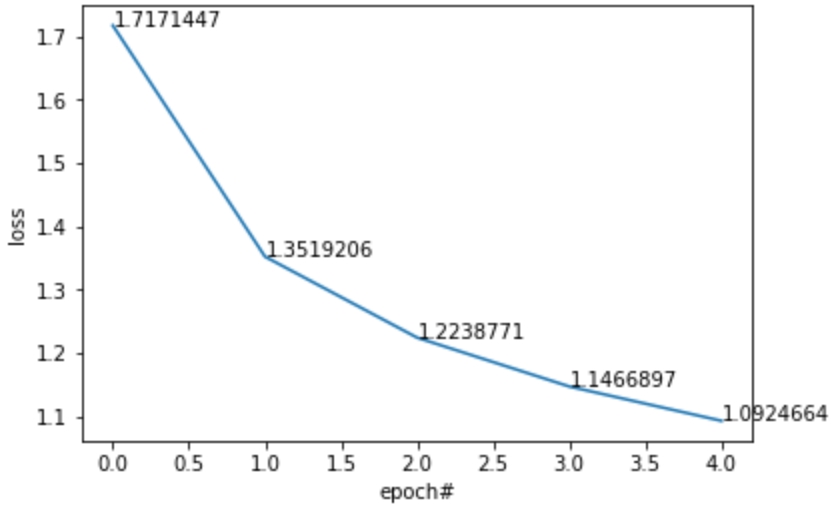
Find below the plots for different values of T and sentence lengths:



T = 0.1, L = 100



T = 1.0, L = 100



T = 0.7, L = 1000 (Default configuration)

Observations: For all the three plots, as shown above, the loss value continuously decreases. Although, the plots are generated for only 5 epochs (could not setup GPU), we should expect a similar trend for more epochs. The loss values taper towards the end, signifying convergence.

4. Here is a musical question about the results: In your observation of the musical quality of the output pay attention to musical aspects of the generated music. For example, the header includes key and meter values. Does the generated music correspond/comply with these values? Is there a correct or consistent metric structure (number of beats in a measure)? Are the different voices or chord harmonious (sounding good together)? Any other musical comments and observations (harmony, musical form, etc.)?

Yes, most of the generated music sequences (in the ABC format) follow that pattern. Occasionally, I've encountered a few sequences not complying (especially the initial music sequences). With more epochs, the values and the structure becomes more consistent. This is accompanied by lower loss values. The initial music is not very pleasant to listen. With more training, the music starts to turn harmonious. I could hear multiple chords being played at times and in sync on most occasions (with an occasional offbeat). The longer sequences generated better harmonies (although they took longer to train). Sometimes, the ABC converter failed to convert the ABC file into a midi file. This was usually for malformed char sequences.