1. Just like last time, provide plots for training error, test error, and test accuracy. Also provide a plot of your train and test perplexity per epoch.
    - In class we defined perplexity as 2^(p*log_2(q)), However the PyTorch cross entropy function uses the natural log. To compute perplexity directly from the cross entropy, you should use e^p*ln(q).
    - We encourage you to try multiple network modifications and hyperparameters, but you only need to provide plots for your best model. Please list the modifications and hyperparameters.
2. What was your final test accuracy? What was your final test perplexity?
    - Final accuracy was ~60% for all networks. It was actually a little bit higher for the LSTM network I trained on reversed data. Maybe when looked at in reverse the English language has a lower variance in how letters are distributed and this made it easier to generalize?
3. What was your favorite sentence generated via each of the sampling methods?
    - My favorite sentences generated were:

        i. Max:

        1. Harry Potter and the second of the centaurs of the centaurs of the centaurs of the centaurs of the centaurs of the centaurs of the centaurs of the centaurs of the centaurs of the centaurs of the centaurs of the centaurs

        ii. Sample:

        1. 2:0 The spirits and goodness is than the world. 10:7 And they shall become me, and all terry: and he saw that it was good.

        iii. Beam:

        1. God is good., and they shall bring them, and they shall bring their house, and they shall bring their house, and they shall bring their house, and they shall bring their house, and they shall bring them, and they
4. What was the prompt you gave to generate that sentence?
    - Max: “Harry Potter and the”
    - Sample: “...and he saw that it was good.” (This was my take on training on reversed data)
    - Beam: “God is good.”
5. Which sampling method seemed to generate the best results? Why do you think that is?
    - Sample Sampling seemed to generate the best across the board. I swept temperature for Sample and Beam. Beam was less affected, but there was a sweet spot for Sample at about T = 0.4-0.6 that generated some unique

sentences without creating garbage (e.g. Harry Potter and the HHeW'p's S\vcent?" Roon room reqise;." Bull. Fr)

6. For sampling and beam search, try multiple temperatures between 0 and 2.
    - Which produces the best outputs? Best as in made the most sense, your favorite, or funniest, doesn't really matter how you decide.
        - i. As addressed above, for Sample a T=0.4 was probably the best. It made the most coherent sense. Beam a higher temp of 2 seemed to work better because it repeated itself less and broke out of loops more easily.
    - What does a temperature of 0 do? What does a temperature of 0<temp<1 do? What does a temperature of 1 do? What does a temperature of above 1 do? What would a negative temperature do (assuming the code allowed for negative temperature)?
        - i. Temp 0 would result in NaNs because we are dividing an exponent by T in the softmax calculation. exp(inf) / sum exp(inf) = NaN
        - ii. 0<T<1 focused the probabilities around the peak(s) of the PMF output. It helps focus the output and ignore lower probabilities
        - iii. T>1 does the opposite and begins to spread the PMF, giving lower probability options a shot. This is why we see on Beam Search a T=2 helps us break the repeating loops.