Q1. According to Wikipedia's list of NVidia graphics cards to an external site., the GeForce RTX 2080 Ti graphics card is capable of 107 TFlops half precision (FP16) using tensor compute and a memory bandwidth of 616 GBytes/sec (this is the bandwidth from the GPU main memory to the ALU registers). Assume you are training an LSTM model using half precision with tensor compute. With these specs, what is the minimum batch size you would want to use to fully utilize the GPU's compute capabilities. Taking into account NVidia's recommendation that batch sizes should be multiples of 128 for optimium memory access patterns, what is the minimum batch size you would recommend using? Show your work, and explain your reasoning and assumptions. Note: FP16 takes 2 bytes of memory.

BWarith = TFLOPS* 2(FP16) = 214

BWmemory = 616 GBytes/sec

BWarith / BWmemory = 214/0.616 = 347.40

Need to be a multiple of 128

128*3 = 384

minimum batch size = 384

Q2. Here are some grammatical and ungrammatical sentences (* indicates an ungrammatical sentence):

The mouse jumped over the cat.

Over the cat, the mouse jumped.

** Over the, the mouse jumped cat.*

According to linguistic tests for constituency, what does this indicate about the words “over the cat” and "over the" in this sentence? Which test(s) did you use to make your determination?

Ans: The original meaning is *The mouse jumped over the cat.*
But "over the cat" also mean m*ouse jumped over the cat ,but it highlight over the cat that behavior* and "*Over the*" mean come the meaning is total different compare to the original meaning.
we can used syntax to determinate the sentence id grammatical or ungrammatical
also we can used the context to confirm grammatical correctness

Q3. Consider a 3-way classification problem for emails, where the classes are SPAM (S), URGENT (U), and OTHER (O). We are training using the perceptron algorithm, and our features are words contained in the email (like in the example in class).

As an example, the feature representation for the email "get here now" when classified as OTHER, i.e. f(x = "get here now", y = OTHER) is

Feature	Value
"label=O & word=get"	+1
"label=O & word=here"	+1
"label=O & word=now"	+1

1) Write the feature vector representation for the email "call me now" when classified as URGENT. In other words, what is f(x = "call me now", y = URGENT)?

Feature	Value
"label=U & word=call"	+1
"label=U & word=me"	+1
"label=U & word=now"	+1

2) Write the feature vector representation for the email "call me now" when classified as SPAM. In other words, what is f(x = "call me now", y = SPAM)?

Feature	Value
"label=S & word=call"	+1
"label=S & word=me"	+1
"label=S & word=now"	+1

3) Suppose our current weight vector is:

Feature	Value
"label=S & word=hello"	-1
"label=U & word=me"	+1
"label=S & word=now"	+2

Values for features not listed are zero.

What class does our classifier classify the email "call me now" as?

score(S) = 0 * "call" + 0 * "me" + 2 * "now" = 2
score(U) = 0 * "call" + 1 * "me" + 0 * "now" = 1
score(O) = 0
Max = 2 , So “call me now ” is Spam

3) Using the same weight vector as in part 3, suppose we have the training example email "call me now" which is classified as URGENT in the gold standard, but our classifier classifies it as you calculated in part 3. What is the new weight vector after applying the perceptron update with stepsize = 1?

Step 1.

Feature	Value
"label=S & word=call"	0 -> -1
"label=S & word=me"	0 -> -1
"label=S & word=now"	+2 -> +1

Step2.

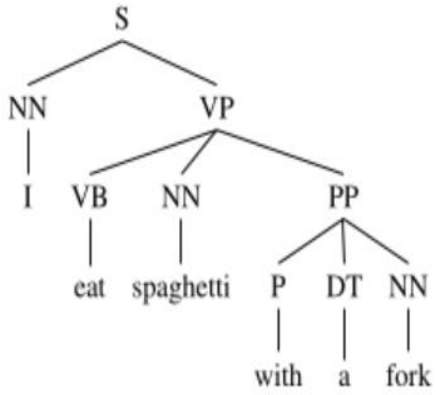
Feature	Value
"label=U & word=call"	0 -> 1
"label=U & word=me"	+1 -> +2
"label=U & word=now"	+1 -> +1

Step3.

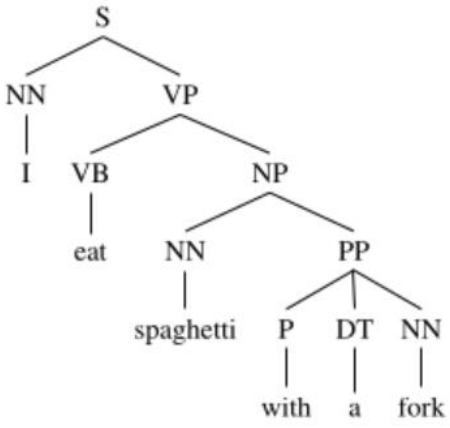
Feature	Value
"label=S & word=hello"	-1
"label=U & word=me"	+1
"label=S & word=now"	+2
"label=U & word=call"	+1
"label=U & word=now"	+1
"label=S & word=call"	-1
"label=S & word=me"	-1

Q4. Below are two parse trees for the same sentence. Suppose the one above is from a treebank, and the one below is the output of a parser. Write down the (head, indexl , indexr) tuples for each tree, where head is the head non-terminal label for each phrase, indexl is the index of the left most word in the phrase, and indexr is the index of the right most word in the phrase. Then compute the precision, recall, and F1 scores for this tree.

Gold parse:



Predicted parse:



1 2 3 4 5 6
I eat spaghetti with a fork

Gold parse:
<S,1,6>
<NN,1,1>
<VP,2,6>
<VB,2,2>
<NN,3,3>
<PP,4,6>
<P,4,4>
<DT,5,5>
<NN,6,6>

Predicted parse:
<S,1,6>
<NN,1,1>
<VP,2,6>
<VB,2,2>
<NP,3,6>
<NN,3,3>
<PP,4,6>
<P,4,4>
<DT,5,5>
<NN,6,6>

Precision: 9/9
Recall: 9/10
F1 = (2*P*R) / P+R = (2*(9/9)*(9/10))/(9/9+9/10)