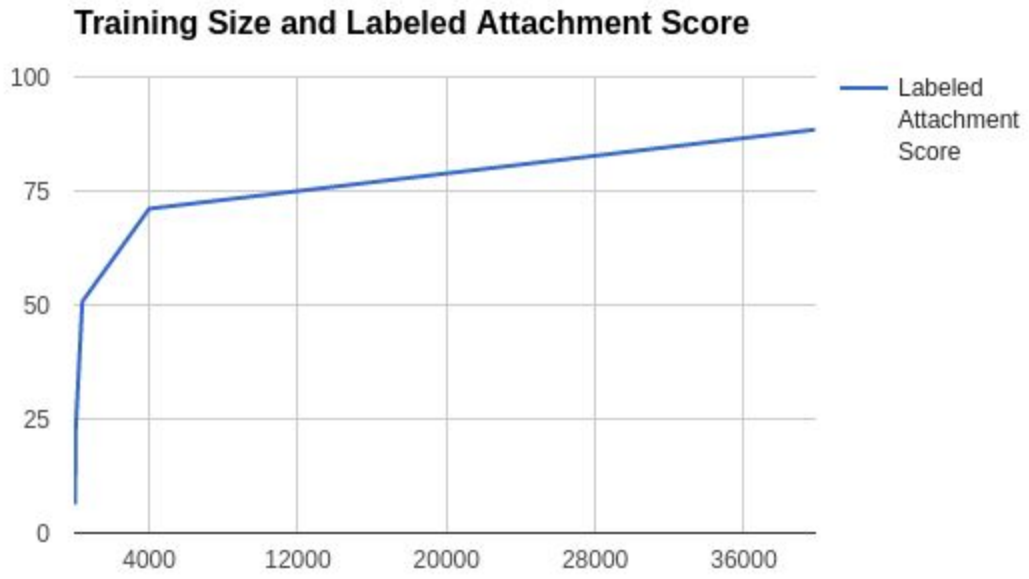
Question 1:

Training Size	Labeled Attachment Score
4	6.25 %
40	22.59 %
400	50.76 %
4000	71.11 %
39,832	88.45 %

Table 1: Training size vs LAS.



Graph 1: LAS vs Trainings Size.

As one might have guessed the bigger the training set the better the accuracy. However as could be seen from the graph there is a a huge diminishing returns and the plotted data is similar to an ln graph. This suggests that even though adding more data to the training set would increase the accuracy, the increase would not be substantial.

Question2 :

Parser	**Labeled Attachment Score**
Default	93.11 %
Nivre Eager	93.11 %
Nivre Standard	93.35 %
Stack Eager	93.32 %
Stack Lazy	93.32 %
Stack Proj	93.35 %

Table 2: Different parsers vs LAS

As seen by the values there is not much of a difference between the parsers. However the best performing one seems to be Stack Proj. (the unlabeled attachment score being the tiebreaker.)

Question 3:

Training Model	POS Score
t4e2s47f14	96.37%
t1e5s1f41	95.63%
t2e2s11f11	96.25%
t1e1s1f1	95.76%
t3e2s11f11	96.30%
t3e1s11f11	96.22%

Table 3: Different tagger training models vs POS score

The best performing parameters were as follows:

Tag order_t: 4
Emission order_e: 2
Unknown word handling_s: 47
Unknown word handling_f: 14

This number is reached by creating a short bash script / python script to iterate through all (within reason) possibilities of the training models and looking at the evaluations results of each and one of them.

Increasing the tag order and emission order increased the evaluation time of each model which made me initially assume that they would return a higher accuracy. However just cranking up the t and e values did not result in a significant accuracy bump. In the end brute-forcing the values gave me much better results compared to my hand crafted attempts, which is definitely not surprising.

Question 4:

POS Source	LAS
Golden	93.35 %
Manual	90.98 %

Table 4: Different tagger training models vs POS score

The result is definitely not surprising. The manual POS tagger resulted in a worse overall LOS score. The result should be obvious as errors that happened in the POS tagging carried over to the LAS score.

The best POS tagger had a 96.37% accuracy which means it was 3.63% inaccurate. We can assume that the parser would make an overall mistake if it is given a wrong POS. From this we can assume the difference between Golden and Manual should be around 3.63% and that is somewhat true with 2.37% difference. (the 1% difference between 3.63% and 2.37% probably comes from distribution and that the POS tagger is better at handling more common POS and wıth the same logic those appear more often)

Question 5:

When I look at the first 20 entries of the results from user trained POS/Parser and compare it to the gold results I see the following:

Only 1/20 crude Tags were off which results in 95% accuracy in the crude tagging.
3/20 were off when it comes to fine POS, and if we assume the error carries over this number is reduced to 2/20.

When we look at the data we can see that both of the errors happen in the same case. The parser instead of choosing NOUN NNP, chose NOUN NNPS.

One option to fix this would be to ensure in the training set there is enough data that correctly differentiates NOUN NNP vs NOUN NPPS. Seeing that the same the only error that happened in fine POS (assuming error was carried over from coarse) in only one type of POS, increasing its training data should in a way to differentiate between the two should hopefully affect the results in a positive way.

All the errors in the parsing section happened in the words that have a hyphen ("-") included in them. Another way to look at this is to say the parsing accuracy is 0% when it comes to phrases that includes a hyphen. ("-") In order combat this problem we can selectively include more correct data that includes hyphens to the training set to target this specific deficiency of our model.