CSC246 Sequence Project Report

Zihan Xie — zxie13
Ningyuan Xiong — nxiong

We just use 1000 training data files because of too long running time of all 50000 files

1. Iterations to converge

We define ε (the minimum absolute change in the likelihood in a given iteration) as 1. The

initial likelihood	-2569.82738
1	-1646.8983
2	-1642.7818
3	-1638.5867
4	-1633.5873
5	-1629.2138
6	-1626.47678
7	-1823.1508
8	-1619.2962
9	-1614.9618
10	-1610.1033
11	-1603.7123
12	-1598.6652
13	-1582.5392
14	-1586.7330
15	-1581.5269
16	-1577.0083
17	-1573.1798
18	-1569.9615
19	-1567.2394
20	-1564.9172

We run the HMM model with 5 hidden units. It takes almost 3 hours to run 1000 data files with 5 hidden units and 20 iterations. From the likelihood above, the decrease of likelihood in an iteration is around 3.5. I think our definition of ε as 1 makes sense. Because the loglikelihood is around 1500. One decrease in likelihood means that there are about 1^-1500 increase in likelihood which is very small. So when the absolute change in the likelihood in a given iteration is less than one, it makes sense that the EM algorithm converges.

2. Number of hidden states and model effectiveness

run below one hour

Hidden states	iterations	initial likelihood	fina likelihood
1	2(converge)	-2479.0133	-1636.18
2	2(converge)	-2441.8486	-1636.2774
3	10	-2425.23	-1633.35
5	5	-2569.82738	-1629.2138
6	5	-2414.66	-1627.3675
10	3	-2413.7946	-1634.4352

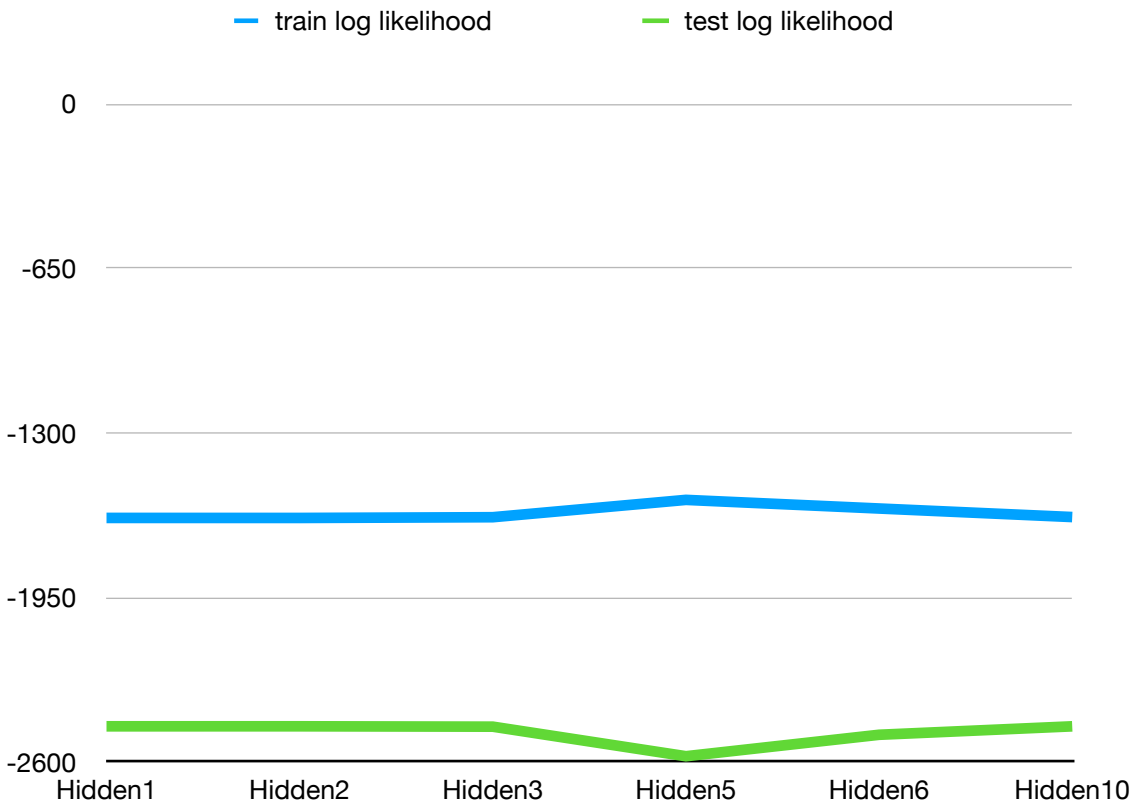
With fewer hidden states, the EM algorithm to train the model runs relative faster. In addition, with few hidden states such as 1 or 2, it takes very few iterations to converge. However, those HMMs with larger than 5 hidden states, it is relative harder to converge, which means the absolute change in likelihood is smaller than ε.
From the result table above, I think HMM models with hidden states 6 is the most effective. The first reason is that in a given period of time(one hour), it does the greatest decrease in log likelihood. And the final log likelihood using the same training data files is the lowest.

3.Results to be theory bounds, compute bounds or memory bounds; overfitting or not

From the table in question1, it iterates 20 rounds. For the first round, it makes the log likelihood decrease almost half. However, the decrease is relative small in the rest iterations(around 3/iteration). So it maybe relates to some theory bounds.

In addition, it is obvious that there is compute bounds. Under one hour, the compute can perform only several iterations. The log likelihood is still decreasing all the time. However, it takes too much time to train the model even with only 1000 training files.

Comparison between train and test data



We can see that the log likelihood for test data is much lower than log likelihood for training data, which means the trained HMM model is a bit overfitting.

Although there are no signals showing that there is under-fitting in our models, it is possible that there exists under-fitting if we use small data set. When our data set is very small and we use vocabularies as the smallest unit, it is highly possible that there exists any vocabularies which is test data but not in training data.

5.additional questions

We can also use sequence model to deal with chatbots, which is automatically reply human messages. This is similar to movie review sequence model. When we handle the data set, we build the similar vocabulary dictionary for each vocabularies. And convert the word inputs to number inputs. Each input units has an emission which means the likely replying words when we see this particular vocabulary. By measuring the success, we can build a dictionary which contains some key words appeared in chat and corresponding possible answers. For example, if the model detects “date”, then the reply message should contain year, month or day.