Question 1: Sequence Models

1. Viterbi Algorithm

We first start by proving that,

$$\delta(Q_t) = \max_{Q_{1:t-1}} P(Q_{1:t}, O_{1:t})$$

where we have slightly modified the notation when compared to the lecture notes to improve readability. Therefore, we should understand that $\delta(Q_t = j) = \delta_j(t)$ and make $Q_t = j$ implicit as it is clear that it is a free variable taking values j from 1 to N. We have the base case of our proof by induction,

$$\delta(Q_1) = p(Q_1)P(O_1|Q_1)$$

Let's suppose the claim is true for $1 : t-1$. We will show that it is true at timestep t.

$$\begin{aligned}
\delta(Q_t) &= P(O_t|Q_t) \max_{Q_{t-1}} P(Q_t|Q_{t-1})\delta(Q_{t-1}) \\
&= P(O_t|Q_t) \max_{Q_{t-1}} P(Q_t|Q_{t-1}) \max_{Q_{1:t-2}} P(Q_{1:t-1}, O_{1:t-1}) \\
&= \max_{1:Q_{t-1}} P(O_t|Q_t)P(Q_t|Q_{t-1})P(Q_{1:t-1}, O_{1:t-1})
\end{aligned}$$

We notice that Q_t is conditionally independent of $Q_{1:t-2}$ and $O_{1:t-1}$ given Q_{t-1}. Therefore, by the product rule,

$$\begin{aligned}
\delta(Q_t) &= \max_{1:Q_{t-1}} P(O_t|Q_t)P(Q_t|Q_{1:t-1}, O_{1:t-1})P(Q_{1:t-1}, O_{1:t-1}) \\
&= \max_{1:Q_{t-1}} P(O_t|Q_t)P(Q_{1:t}, O_{1:t-1})
\end{aligned}$$

Moreover, O_t is conditionally independent of $Q_{1:t-1}$ and $O_{1:t-1}$ given Q_t. By the product rule,

$$\begin{aligned}
\delta(Q_t) &= \max_{1:Q_{t-1}} P(O_t|Q_{1:t}, O_{1:t-1})P(Q_{1:t}, O_{1:t-1}) \\
&= \max_{1:Q_{t-1}} P(Q_{1:t}, O_{1:t})
\end{aligned}$$

We have proven that $\delta(Q_t)$ indicates the value of the most likely sequence of hidden states. Let's now prove that we can recover the optimal hidden states sequence, defined as,

$$\arg\max_{Q_t} \max_{1:Q_{t-1}} P(Q_{1:t}, O_{1:t})$$

The base case of our proof by induction is,

$$Q_T^* = \arg\max_{Q_T} \delta(Q_T)$$

We suppose that the claim is true for $t+1$ where Q^*_{t+1} is the optimal hidden state at timestep $t+1$ for the most likely hidden state sequence. Let's prove that it is also true for timestep t,

$$\begin{aligned}
Q^*_t &= \arg\max_{Q_t} P(Q^*_{t+1}|Q_t)\delta(Q_t) \\
&= \arg\max_{Q_t} P(Q^*_{t+1}|Q_t) \max_{1:Q_{t-1}} P(Q_{1:t}, O_{1:t}) \\
&= \arg\max_{Q_t} \max_{1:Q_{t-1}} P(Q^*_{t+1}|Q_t)P(Q_{1:t}, O_{1:t}) \\
&= \arg\max_{Q_t} \max_{1:Q_{t-1}} P(Q^*_{t+1}|Q_{1:t}, O_{1:t})P(Q_{1:t}, O_{1:t}) \\
&= \arg\max_{Q_t} \max_{Q_1:Q_{t-1}} P(Q^*_{t+1}, Q_{1:t}, O_{1:t}) \\
&= \arg\max_{Q_t} \max_{Q_1:Q_{t-1}} \max_{Q_{t+2:T}} P(Q_{t+2:T}, O_{t+1:T}|Q_{t+1}*)P(Q^*_{t+1}, Q_{1:t}, O_{1:t}) \\
&= \arg\max_{Q_t} \max_{Q_1:Q_{t-1}} \max_{Q_{t+2:T}} P(Q_{t+2:T}, O_{t+1:T}|Q^*_{t+1}, Q_{1:t}, O_{1:t})P(Q_{t+1}*, Q_{1:t}, O_{1:t}) \\
&= \arg\max_{Q_t} \max_{Q_1:Q_{t-1}} \max_{Q_{t+2:T}} P(Q_{1:t}, Q_{t+1}*, Q_{t+2:T}, O_{1:T})
\end{aligned}$$

where we have used the fact that Q^*_{t+1} is conditionally of $Q_{1:t-1}$ and $O_1 : t$ given Q_t and $Q_{t+2:T}, O_{t+1:T}$ is conditionally independent of $Q_{1:t}$ and$O_{1:t}$ given Q^*_{t+1}. The final equality tells us that Q^*_t is the hidden state at timestep t for the most likely hidden state sequence. We have therefore proven the claim.

2. Linear Chain Conditional Random Field

We will disprove the claim by showing that the the objective maximized by the sum of cross-entropy terms at every timestep is not equivalent to negative log likelihood loss. We rewrite the negative log likelihood as the likelihood in order to clarify the derivation. Therefore,

$$\begin{aligned}
l(\theta) &= \prod_i P(Y^{(i)}|X^{(i)}) \\
&= \prod_i \frac{1}{Z(X^{(i)})} \exp\left(\sum_t \sum_k \theta_k f_k(y^{(i)}_t, y^{(i)}_{t-1}, x^{(i)}_t)\right) \qquad (1)
\end{aligned}$$

The cross-entropy terms can also be rewritten in terms of likelihood (as the cross-entropy is equivalent to the negative log likelihood), and we get,

$$l(\theta) = \prod_i \prod_t p(y^{(i)}_t|X^{(i)})$$

We notice a lose resemblance to Eq.1, as,

$$\exp\left(\sum_t \sum_k \theta_k f_k(y^{(i)}_t, y^{(i)}_{t-1}, x^{(i)}_t)\right) = \prod_t \exp\left(\sum_k \theta_k f_k(y^{(i)}_t, y^{(i)}_{t-1}, x^{(i)}_t)\right)$$

For the objectives to be equivalent, we would need that $p(y^{(i)}_t|X^{(i)})$ simplifies to $\exp\left(\sum_k \theta_k f_k(y^{(i)}_t, y^{(i)}_{t-1}, x^{(i)}_t)\right)$. However, this is not the case as,

$$p(y^{(i)}_t|X^{(i)}) = \frac{1}{Z(X^{(i)})} \sum_{y^{(i)}_{t'}} \exp\left(\sum_t \sum_k \theta_k f_k(y^{(i)}_t, y^{(i)}_{t-1}, x^{(i)}_t)\right)$$

where $y_{t'}^{(i)}$ represents the set of all $y^{(i)}$ in a sequence except for timestep $y_t^{(i)}$. This final expression does not simplify further. We can see this through the fact that each factor f_k depends on the previous timestep and the current timestep through y_{t-1} and y_t respectively. This is a direct consequence of the fact that the data is not i.i.d. but instead sequential.

Question 2: Decipherment with HMMs

Instances of correct sentences

- Je vous aime
- Il regarde Jackie
- Les beaux chiens regardent ces plantes colorées

Instances of incorrect sentences

- Jesus regardent ces noirs chiens
- Il aime cet plante colorée
- Les beaux chiens mangeons des plantes extraordinaires

Answers to questions

1. What are some advantages of modelling French grammar with a CFG?
The French language is sometimes very complicated and its grammar tries to characterize every possible situation. For example, this leads to a situation where there are more than 8 different types of pronouns. When modelling the French grammar with a CFG, we perhaps lose some of those details but gain in simplicity. This most likely can still lead to a model that performs well and is more easy to implement for a specific task, where only a certain part of the French grammar is relevant. Moreover, the CFG can better prescribe the position of certain elements in the sentence. For example, pronouns like "le", "la" or "me" are considered as personal pronouns by the French grammar, but this does not make explicit the fact that they have to be placed before the the verb when used as direct object pronouns. A CFG can better model these relationships and even more complicated ones through recurrence.

2. What are some disadvantages of modelling French grammar with a CFG?
Trying to accurately model the French grammar can lead to a exponential complexity. This comes from the fact that verbs, nouns, determiners and adjectives must all respect the gender and/or the number. Implementing agreement between all these entities could perhaps be more easy to do through feature structure and unification (Chapter 16 from JM's book) rather than our naive approach. Moreover, in the French language, exceptions to the rules are common (if not more common than the rules themselves). An obvious example is the way verbs are conjugated. These exceptions would lead to impractical implementations through a CFG.

3. What are some aspects of French grammar that your CFG does not handle?
For the verb tenses, we only model the present tense, which is the most simple tense. For example,

if we were to try to model the *participe passé*, we would need to consider the the gender and number of the direct object pronouns. Moreover, we do not handle cases where the adjective and the noun simply do not make logical sense when together, such as "des rues généreuses" (unless you are considering writing poetry, where anything can make sense). Moreover, some adjectives can both be placed before or after the verb, but only when they come together with the right nouns. For example, "la semaine dernière" is correct, but "le chat dernier" is incorrect (while "le dernier chat" is correct). There are many more basic aspects of French that we don't cover, such as conjunction or adverbs.

Question 3: Decipherment with HMMs

	Standard		Laplace		Extra data		Laplace+Extra Data	
	Train	Test	Train	Test	Train	Test	Train	Test
Cipher 1	100.0	09.77	99.51	97.66	100.0	69.17	98.76	**98.10**
Cipher 2	93.45	14.19	90.06	83.12	88.76	62.16	86.23	**87.88**
Cipher 3	34.20	21.30	33.09	21.30	31.23	20.78	30.62	**21.65**

Table 1: Results for training accuracy and test accuracy presented in percentage for each of the configurations and for each of the ciphers. For each cipher (row), we highlight in **bold** the configuration giving the best test accuracy.

When we train a standard HMM, we notice that the model clearly overfits for cipher 1 and cipher 2, but underfits on cipher 3 (as the train and test accuracies are both very low). When we train the HMM with Laplace smoothing we notice it successfully reduces the overfitting for both cipher 1 and 2 and dramatically improves their test accuracy. This means that the complexity of our HMM models was high enough but the training dataset was too small, inducing overfitting. In the case of cipher 3, adding Laplace smoothing does not improve or worsen the test data, it simply reduces the the training accuracy. This is evidence that the complexity of an HMM is not high enough for this cipher.

When we add extra data to train the transitions probabilities, we confirm our hypothese that the original training dataset was too small for cipher 1 and 2, as for both datasets we significantly increase the test accuracy. This increase is not as efficient as adding regularization in the form of the Laplace smoothing, which could be seen as surprising. However, if we were allowed to also train the emission and starting probabilities with extra data, it would be very likely that the test accruacy would improve further. As a simple experiment, we tried to also train the starting probabilities with the extra data and noticed test accuracies of **85.10%** and **90.12%** for cipher 1 and 2 respectively. In the case of cipher 2, this improvement actually leads to the best test accuracy overall (even higher than combining Laplace smoothing and extra data). Given that the parameters of the HMM are trained by MLE, this is not surprising as the more data we have, the better estimates we can obtain. When we add extra data for the cipher 3, we actually observe a decrease in accuracy for both train and test accuracies. This is yet more evidence that the Gilbert Vernam's encoding strategy is far too clever for our current models. In the last experiment, when we combine Laplace smoothing and extra data, we observe the best accuracy for all the ciphers. This experiments also showcases that cipher 1 is more simple to decipher as we achieve close to perfect test accuracy whilecipher 2 achieves pretty good accuracy but could still be improved by adjusting the smoothing parameter and adding more data. Finally, cipher 3 is much more difficult to decode than the other two. For cipher 3, we would actually need a more complex model, perhaps involving neural networks.