Natural Language Processing: Assignment 2

Isaac Jefferson Lee

Email: isalee@student.ethz.ch

Question 1

Question 1 :: Part a)

We are asked to prove that the expectation semiring satisfies the semiring axioms. So first we prove that $\langle \mathbb{R} \times \mathbb{R}, \oplus, \mathbf{0} \rangle$ is a commutative monoid.

Suppose that we have some arb. $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in \mathbb{R} \times \mathbb{R}$ where we have defined

$$\boldsymbol{x} := \langle x_1, x_2, \rangle, \boldsymbol{y} := \langle y_1, y_2, \rangle, \boldsymbol{z} := \langle z_1, z_2, \rangle, .$$

Where $x_i, y_i, z_i \in \mathbb{R}$, then it follows that:

$$\begin{aligned} \boldsymbol{x} \oplus (\boldsymbol{y} \oplus \boldsymbol{z}) &= \langle x_1, x_2 \rangle \oplus (\langle y_1, y_2 \rangle \oplus \langle z_1, z_2 \rangle) \\ &= \langle x_1, x_2 \rangle \oplus (\langle y_1 + z_1, y_2 + z_2 \rangle) \\ &= \langle x_1, x_2 \rangle \oplus \langle y_1 + z_1, y_2 + z_2 \rangle \\ &= \langle x_1 + (y_1 + z_1), x_2 + (y_2 + z_2) \rangle \\ &= \langle x_1 + y_1 + z_1, x_2 + y_2 + z_2 \rangle \qquad (\dagger) \end{aligned}$$

.

where the last line follows from the associativity of $+$ over $\mathbb{R}$. Similarly:

$$\begin{aligned} (\boldsymbol{x} \oplus \boldsymbol{y}) \oplus \boldsymbol{z} &= (\langle x_1, x_2 \rangle \oplus \langle y_1, y_2 \rangle) \oplus \langle z_1, z_2 \rangle \\ &= (\langle x_1 + y_1, x_2 + y_2 \rangle) \oplus \langle z_1, z_2 \rangle \\ &= \langle x_1 + y_1, x_2 + y_2 \rangle \oplus \langle z_1, z_2 \rangle \\ &= \langle (x_1 + y_1) + z_1, (x_2 + y_2) + z_2 \rangle \\ &= \langle x_1 + y_1 + z_1, x_2 + y_2 + z_2 \rangle \qquad (\dagger\dagger) \end{aligned}$$

.

Where again, the last line follows from associativity of $+$ over $\mathbb{R}$. So we see that:

$$(\dagger) = (\dagger\dagger) \implies \text{Associativity.}$$

Next we prove the existence of the identity element $= \mathbf{0}$. Choose some arb. $\boldsymbol{x} \in \mathbb{R} \times \mathbb{R}$, then:

$$\boldsymbol{x} \oplus \mathbf{0} := \langle x_1 + x_2 \rangle \oplus \langle 0, 0 \rangle = \langle 0 + x_1, 0 + x_2 \rangle.$$

And by the fact that $\mathbf{0}$ is the identity for $+$ over $\mathbb{R}$ it follows that:

$$\boldsymbol{x} + \mathbf{0} = \boldsymbol{x} = \mathbf{0} + \boldsymbol{x}.$$

So it remains to prove commutativity.

Suppose we choose some arb. $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R} \times \mathbb{R}$, then

$$\boldsymbol{x} \oplus \boldsymbol{y} := \langle x_1, x_2 \rangle \oplus \langle y_1, y_2 \rangle = \langle x_1 + y_1, x_2 + y_2 \rangle.$$

$$\boldsymbol{y} \oplus \boldsymbol{x} := \langle y_1, y_2 \rangle \oplus \langle x_1, x_2 \rangle = \langle y_1 + x_1, y_2 + x_2 \rangle.$$

By the commutativity of $+$ over $\mathbb{R}$, $x_i + y_i = y_i + x_i \ \forall i \implies \boldsymbol{x} + \boldsymbol{y} = \boldsymbol{y} + \boldsymbol{x}$ And since our choice of $\boldsymbol{x}$ and $\boldsymbol{y}$ were arb. we have proved that $\langle \mathbb{R} \times \mathbb{R}, \oplus, \mathbf{0} \rangle$ is a commutative monoid.

So now we need to prove that $\langle \mathbb{R} \times \mathbb{R}, \otimes, \mathbf{1} \rangle$ is a monoid. Similar to before, we first verify associativity: Consider arb. $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in \mathbb{R} \times \mathbb{R}$,

$$\begin{aligned}
\boldsymbol{x} \otimes (\boldsymbol{y} \otimes \boldsymbol{z}) &= \langle x_1, x_2 \rangle \otimes (\langle y_1, y_2 \rangle \otimes \langle z_1, z_2 \rangle) \\
&= \langle x_1, x_2 \rangle \otimes (\langle y_1 * z_1, y_1 * z_2 + y_2 * z_1 \rangle) \\
&= \langle x_1, x_2 \rangle \otimes \langle y_1 * z_1, y_1 * z_2 + y_2 * z_1 \rangle \\
&= \langle x_1 * (y_1 * z_1), x_1 * (y_1 * z_2 + y_2 * z_1) + x_2 * (y_1 * z_1) \rangle
\end{aligned}$$

By distributivity, associativity and commutativity of $*$ and $+$ over $\mathbb{R}$ we have

$$= \langle x_1 * y_1 * z_1, x_1 * y_1 * z_2 + x_1 * y_2 * z_1 + x_2 * y_1 * z_1 \rangle \qquad (\dagger).$$

Then RHS is given by:

$$\begin{aligned}
(\boldsymbol{x} \otimes \boldsymbol{y}) \otimes \boldsymbol{z} &= (\langle x_1, x_2 \rangle \otimes \langle y_1, y_2 \rangle) \otimes \langle z_1, z_2 \rangle \\
&= (\langle x_1 * y_1, x_1 * y_2 + x_2 * y_1 \rangle) \otimes \langle z_1, z_2 \rangle \\
&= \langle x_1 * y_1, x_1 * y_2 + x_2 * y_1 \rangle \otimes \langle z_1, z_2 \rangle \\
&= \langle (x_1 * y_1) * z_1, (x_1 * y_1) * z_2 + (x_1 * y_2 + x_2 * y_1) * z_1 \rangle
\end{aligned}$$

Again by distributivity, associativity and commutativity of $*$ and $+$ over $\mathbb{R}$ we have

$$= \langle x_1 * y_1 * z_1, x_1 * y_1 * z_2 + x_1 * y_2 * z_1 + x_2 * y_1 * z_1 \rangle \qquad (\dagger\dagger).$$

And therefore we get:

$$(\dagger\dagger) = (\dagger) \implies \text{Associativity.}$$

Now we must prove that the identity element exists (and $= \mathbf{1}$). So suppose that we have some arb. $\boldsymbol{x} := \langle x_1, x_2 \rangle \in \mathbb{R} \times \mathbb{R}$ where we have $\mathbf{1} := \langle 1, 0 \rangle \in \mathbb{R} \times \mathbb{R}$ then

$$\boldsymbol{x} \otimes \mathbf{1} = \langle x_1, x_2 \rangle \otimes \langle 1, 0 \rangle \in \mathbb{R} \times \mathbb{R}) = \langle x_1 * 1, x_1 * 0 + x_2 * 1 \rangle.$$

Since 1 is the identity of $*$ over $\mathbb{R}$ and 0is the annihilator $\implies = \langle x_1, x_2 \rangle$

Similarly $\mathbf{1} \otimes \boldsymbol{x} = \langle 1, 0 \rangle \otimes \langle x_1, x_2 \rangle = \langle 1 * x_1, 1 * x_2 + 0 * x_1 \rangle = \langle x_1, x_2 \rangle$

So we have now proved that $\langle \mathbb{R} \times \mathbb{R}, \otimes, \mathbf{1} \rangle$ is a monoid.

Now we need to prove that $\otimes$ distributes over $\oplus$, i.e

$$\forall \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in \mathbb{R} \times \mathbb{R} (\boldsymbol{x} \oplus \boldsymbol{y}) \otimes \boldsymbol{z} = (\boldsymbol{x} \otimes \boldsymbol{z}) \oplus (\boldsymbol{y} \otimes \boldsymbol{z}) \qquad (*).$$

and

$$\boldsymbol{z} \otimes (\boldsymbol{x} \oplus \boldsymbol{y}) = (\boldsymbol{z} \otimes \boldsymbol{x}) \oplus (\boldsymbol{z} \otimes \boldsymbol{y}) \qquad (**).$$

Consider the LHS of $(*)$:

$$\begin{aligned}
(\boldsymbol{x} \oplus \boldsymbol{y}) \otimes \boldsymbol{z} &= (\langle x_1, x_2 \rangle \oplus \langle y_1, y_2 \rangle) \otimes \langle z_1, z_2 \rangle \\
&= (\langle x_1 + y_1, x_2 + y_2 \rangle) \otimes \langle z_1, z_2 \rangle \\
&= \langle x_1 + y_1, x_2 + y_2 \rangle \otimes \langle z_1, z_2 \rangle \\
&= \langle x_1 + y_1, x_2 + y_2 \rangle \otimes \langle z_1, z_2 \rangle \\
&= \langle (x_1 + y_1) * z_1, (x_1 + y_1) * z_2 + (x_2 + y_2) * z_1 \rangle
\end{aligned}$$

Again by distributivity, associativity and commutativity of $*$ and $+$ over $\mathbb{R}$ we have:

$$= \langle x_1 * z_1 + y_1 * z_1, x_1 + z_2 + y_1 * z_2 + x_2 * z_1 + y_2 * z_1 \rangle \qquad (\dagger).$$

Now consider the RHS of $(*)$:

$$\begin{aligned}(\boldsymbol{x} \otimes \boldsymbol{z}) \oplus (\boldsymbol{y} \otimes \boldsymbol{z})) &= (\langle x_1, x_2 \rangle \otimes \langle z_1, z_2 \rangle) \oplus (\langle y_1, y_2 \rangle \otimes \langle z_1, z_2 \rangle) \\ &= (\langle x_1 * z_1 + x_1 * z_2 + x_2 * z_1 \rangle) \oplus (\langle y_1 * z_1, y_1 * z_2 + y_2 * z_1 \rangle) \\ &= \langle x_1 * z_1 + x_1 * z_2 + x_2 * z_1 \rangle \oplus \langle y_1 * z_1, y_1 * z_2 + y_2 * z_1 \rangle \\ &= \langle x_1 * z_1 + y_1 * z_1, x_1 * z_2 + x_2 * z_1 + y_1 * z_2 + y_2 * z_1 \rangle \qquad (\dagger\dagger)\end{aligned}$$

$(\dagger) = (\dagger\dagger)$ so we have the first part of distributivity proved.

Now consider $(**)$. The LHS of $(**)$ is given by:

$$\begin{aligned}\boldsymbol{z} \otimes (\boldsymbol{x} \oplus \boldsymbol{y}) &= \langle z_1, z_2 \rangle \otimes (\langle x_1, x_2 \rangle \oplus \langle y_1, y_2 \rangle) \\ &= \langle z_1, z_2 \rangle \otimes (\langle x_1 + y_1, x_2 + y_2 \rangle) \\ &= \langle z_1, z_2 \rangle \otimes \langle x_1 + y_1, x_2 + y_2 \rangle \\ &= \langle z_1 * (x_1 + y_1), z_1 * (x_2 + y_2) + z_1 * (x_1 + y_1) \rangle\end{aligned}$$

Again by distributivity, associativity and commutativity of $*$ and $+$ over $\mathbb{R}$ we have:

$$= \langle x_1 * z_1 + y_1 * z_1, x_2 * z_1 + y_2 * z_1 + x_1 * z_2 + y_1 * z_2 \rangle \qquad (\dagger).$$

The RHS of $(**)$ is given by:

$$\begin{aligned}(\boldsymbol{z} \otimes \boldsymbol{x}) \oplus (\boldsymbol{z} \otimes \boldsymbol{y}) &= (\langle z_1, z_2 \rangle \otimes \langle x_1, x_2 \rangle) \oplus (\langle z_1, z_2 \rangle \otimes \langle y_1, y_2 \rangle) \\ &= (\langle z_1 * x_1, z_1 * x_2 + z_2 * x_1 \rangle) \oplus (\langle z_1 * y_1, z_1 * y_2 + z_2 * y_1 \rangle) \\ &= \langle z_1 * x_1, z_1 * x_2 + z_2 * x_1 \rangle \oplus \langle z_1 * y_1, z_1 * y_2, z_2 * y_1 \rangle \\ &= \langle z_1 * x_1 + z_1 * y_1, z_1 * x_2 + z_2 * x_1 + z_1 * y_2 + z_2 * y_1 \rangle \qquad (\dagger\dagger)\end{aligned}$$

So $(\dagger) = (\dagger\dagger) \implies$ we have proved distributivity.

It remains to verify that $\mathbf{0}$ is an annihilator for $\otimes$. Suppose we have some arb. $\boldsymbol{x} \in \mathbb{R} \times \mathbb{R}$, then:

$$\begin{aligned}\boldsymbol{x} \otimes \mathbf{0} &= \langle x_1, x_2 \rangle \otimes \langle 0, 0 \rangle \\ &= \langle x_1 * 0, x_1 * 0 + x_2 * 0 \rangle \\ &= \langle 0, 0 \rangle\end{aligned}$$

$$\begin{aligned}\mathbf{0} \otimes \boldsymbol{x} &= \langle 0, 0 \rangle \otimes \langle x_1, x_2 \rangle \\ &= \langle 0 * x_1, 0 * x_2 + 0 * x_1 \rangle \\ &= \langle 0, 0 \rangle\end{aligned}$$

And we are done.

QED.

Question 1 :: Part b)

Recall that the forward algorithm calculates $\alpha(\boldsymbol{w}, EOS)$ which is defined to be the sum of the scores of all paths starting from BOS and ending at EOS.

Suppose we define:

$$\omega_n := \exp\left(\text{score}(\langle t_{n-1}, t_n \rangle), \boldsymbol{w}\right).$$

Algorithm 1 Lifted Forward Algorithm

lifted_forward($\boldsymbol{w}, \mathcal{T}, N$)
for $t_1 \in \mathcal{T}$ **do**
  $\alpha(\boldsymbol{w}, t_1, 1) \leftarrow \mathbf{1} := \langle 1, 0 \rangle$
end for
for $n \in 2, ..., N$ **do**
  for $t_n \in \mathcal{T}$ **do**
    $\alpha(\boldsymbol{w}, t_n, n) \leftarrow \bigoplus_{t_{n-1} \in \mathcal{T}} \langle \exp\left(\text{score}(\langle t_{n-1}, t_n \rangle, \boldsymbol{w})\right),$
    $-\exp\left(\text{score}(\langle t_{n-1}, t_n \rangle, \boldsymbol{w})\right) \log\left(\exp\left(\text{score}(\langle t_{n-1}, t_n \rangle, \boldsymbol{w})\right)\right) \rangle \otimes \alpha(\boldsymbol{w}, t_{n-1}, n-1)$
  end for
end for

Then lifting into the expectation semi-ring we have:

$$\omega_n \mapsto \langle \omega_n, -\omega_n \log(\omega_n) \rangle.$$

Recall the forward algorithm:
For ease of notation, let us define:

$$\phi_n := \text{score}(\langle t_{n-1}, t_n \rangle).$$

So our lifted tuple is:

$$\langle \exp(\phi_n), -\exp(\phi_n) * \phi_n \rangle.$$

So for the first iteration of our lifted algorithm, when $n = 2$, we get

$$\begin{aligned}
\forall t_2 \in \mathcal{T}, \alpha(\boldsymbol{w}, t_2, 1) &= \bigoplus_{t_1 \in \mathcal{T}} \langle \exp(\phi_2), -\exp(\phi_2) * \phi_2 \rangle \otimes \langle 1, 0 \rangle \\
&= \bigoplus_{t_1 \in \mathcal{T}} \langle \exp(\phi_2), -\exp(\phi_2) * \phi_2 \rangle \\
&= \langle \sum_{t_1 \in \mathcal{T}} \exp(\phi_2), -\sum_{t_1 \in \mathcal{T}} \phi_2 \exp(\phi_2) \rangle
\end{aligned}$$

Then for $n = 3$, we have:

$$\begin{aligned}
\forall t_3 \in \mathcal{T}, \alpha(\boldsymbol{w}, t_3, 1) &= \bigoplus_{t_2 \in \mathcal{T}} \langle \exp(\phi_3), -\phi_3 \exp(\phi_3) \rangle \otimes \langle \sum_{t_1 \in \mathcal{T}} \exp(\phi_2), -\sum_{t_1 \in \mathcal{T}} \phi_2 \exp(\phi_2) \rangle \\
&= \bigoplus_{t_2 \in \mathcal{T}} \langle \exp(\phi_3) \sum_{t_1 \in \mathcal{T}} \exp(\phi_2), \\
&\quad -\exp(\phi_3) \sum_{t_1 \in \mathcal{T}} \phi_2 \exp(\phi_2) - \phi_3 \exp(\phi_3) \sum_{t_1 \in \mathcal{T}} \exp(\phi_2) \rangle \\
&= \bigoplus_{t_2 \in \mathcal{T}} \langle \exp(\phi_3) \sum_{t_1 \in \mathcal{T}} \exp(\phi_2), \\
&\quad -\sum_{t_1 \in \mathcal{T}} \phi_2 \exp(\phi_3) \exp(\phi_2) - \sum_{t_1 \in \mathcal{T}} \phi_3 \exp(\phi_3) \exp(\phi_2) \rangle
\end{aligned}$$

.

Then combining sums and grouping terms we get:

$$= \left\langle \sum_{t_2 \in \mathcal{T}} \left[\exp(\phi_3) \sum_{t_1 \in \mathcal{T}} \exp(\phi_2) \right], -\sum_{t_2 \in \mathcal{T}} \sum_{t_1 \in \mathcal{T}} (\phi_2 + \phi_3) \exp(\phi_3) \exp(\phi_2) \right\rangle.$$

We can apply this procedure for $n = 4, 5, ..., N$ and thus show by induction that if

$$\alpha(\boldsymbol{w}, \text{EOS}, N+1) := \langle \gamma_1, \gamma_2 \rangle.$$

then it follows that:

$$\gamma_2 = -\sum_{t_N \in \mathcal{T}} \sum_{t_{N-1} \in \mathcal{T}} \dots \sum_{t_1 \in \mathcal{T}} \left[\sum_{n'=1}^{N} \left[\phi_{n'} \prod_{n=2}^{N} \exp(\phi_n) \right] \right].$$

Note: $n = 1 \implies \phi_n = 0 \implies \exp(\phi_n) = 1$

$$\implies \prod_{n=2}^{N} \exp(\phi_n) = \prod_{n=1}^{N} \exp(\phi_n).$$

$$\implies \gamma_2 = -\sum_{\boldsymbol{t} \in \mathcal{T}^N} \left[\sum_{n'=1}^{N} \left[\phi_{n'} \prod_{n=1}^{N} \exp(\phi_n) \right] \right] = \sum_{\boldsymbol{t} \in \mathcal{T}^N} \left[\left(\prod_{n=1}^{N} \exp(\phi_n) \right) \left(\sum_{n'=1}^{N} \phi_{n'} \right) \right].$$

$$= -\sum_{\boldsymbol{t} \in \mathcal{T}^N} [\exp(\text{score}(\boldsymbol{t}, \boldsymbol{w})) * \text{score}(\boldsymbol{t}, \boldsymbol{w})] = H_U(T_{\boldsymbol{w}}).$$

So we see that the second component of the lifted tuple will give us $H_U(T_{\boldsymbol{w}})$. Also it is clear by distributitivy that the first element in the tuple, γ_1 will give us our normalizing constant $Z(\boldsymbol{w})$.

Question 1 :: Part c)

We want to show:

$$H(T_{\boldsymbol{w}}) = \frac{1}{Z(\boldsymbol{w})} H_U(T_{\boldsymbol{w}}) + \log(Z(\boldsymbol{w})).$$

Recall that:

$$H_U(T_{\boldsymbol{w}}) := -\sum_{\boldsymbol{t} \in \mathcal{T}^N} \exp(\text{score}(\boldsymbol{t}, \boldsymbol{w})) \text{score}(\boldsymbol{t}, \boldsymbol{w}).$$

Substituting this into the Right Hand Side we get:

$$RHS = -\frac{1}{Z(\boldsymbol{w})} \sum_{\boldsymbol{t} \in \mathcal{T}^N} [\exp(\text{score}(\boldsymbol{t}, \boldsymbol{w})) \text{score}(\boldsymbol{t}, \boldsymbol{w})] + \log(Z(\boldsymbol{w})).$$

Now shifting our focus to the Left Hand Side:

$$LHS = H(T_{\boldsymbol{w}}) := -\sum_{\boldsymbol{t} \in \mathcal{T}^N} P(\boldsymbol{t}|\boldsymbol{w}) \log(P(\boldsymbol{t}|\boldsymbol{w})).$$

Recall that: $P(\boldsymbol{t}|\boldsymbol{w}) := \exp(\text{score}(\boldsymbol{t}, \boldsymbol{w}))/Z(\boldsymbol{w})$

$$\implies H(T_{\boldsymbol{w}}) = -\sum_{\boldsymbol{t} \in \mathcal{T}^N} \left[\frac{\exp(\text{score}(\boldsymbol{t}, \boldsymbol{w}))}{Z(\boldsymbol{w})} \log \left(\frac{\exp(\text{score}(\boldsymbol{t}, \boldsymbol{w}))}{Z(\boldsymbol{w})} \right) \right].$$

Then by properties of logs we have:

$$= -\sum_{\boldsymbol{t} \in \mathcal{T}^N} \left[\frac{\exp(\text{score}(\boldsymbol{t}, \boldsymbol{w}))}{Z(\boldsymbol{w})} (\log(\exp(\text{score}(\boldsymbol{t}, \boldsymbol{w}))) - \log(Z(\boldsymbol{w}))) \right]$$

$$= -\sum_{\boldsymbol{t} \in \mathcal{T}^N} \left[\frac{\exp(\text{score}(\boldsymbol{t}, \boldsymbol{w}))}{Z(\boldsymbol{w})} (\text{score}(\boldsymbol{t}, \boldsymbol{w}) - \log(Z(\boldsymbol{w}))) \right]$$

$$= -\frac{1}{Z(\boldsymbol{w})} \sum_{\boldsymbol{t} \in \mathcal{T}^N} [\exp(\text{score}(\boldsymbol{t}, \boldsymbol{w})) \text{score}(\boldsymbol{t}, \boldsymbol{w}) - \exp(\text{score}(\boldsymbol{t}, \boldsymbol{w})) \log(Z(\boldsymbol{w}))]$$

$$= -\frac{1}{Z(\boldsymbol{w})} \sum_{\boldsymbol{t} \in \mathcal{T}^N} [\exp(\text{score}(\boldsymbol{t}, \boldsymbol{w})) \text{score}(\boldsymbol{t}, \boldsymbol{w})] + \frac{1}{Z(\boldsymbol{w})} \sum_{\boldsymbol{t} \in \mathcal{T}^N} \exp(\text{score}(\boldsymbol{t}, \boldsymbol{w}) \log(Z(\boldsymbol{w})))$$

So now we just have to show that:

$$\frac{1}{Z(\boldsymbol{w})} \sum_{\boldsymbol{t} \in \mathcal{T}^N} [\exp(\text{score}(\boldsymbol{t}, \boldsymbol{w})) \log(Z(\boldsymbol{w}))] = \log(Z(\boldsymbol{w})).$$

and then we get that $LHS = RHS$.

Note: Since $\log(Z(\boldsymbol{w}))$ has no dependence on $\boldsymbol{t}$, we can then take it out of the sum to get:

$$\frac{1}{Z(\boldsymbol{w})} \sum_{\boldsymbol{t} \in \mathcal{T}^N} [\exp(\text{score}(\boldsymbol{t}, \boldsymbol{w})) \log(Z(\boldsymbol{w}))] = \frac{\log(Z(\boldsymbol{w}))}{Z(\boldsymbol{w})} \underbrace{\sum_{\boldsymbol{t} \in \mathcal{T}^N} \exp(\text{score}(\boldsymbol{t}, \boldsymbol{w}))}_{:= Z(\boldsymbol{w})}.$$

$$= \frac{\log(Z(\boldsymbol{w}))}{Z(\boldsymbol{w})} * Z(\boldsymbol{w}) = \log(Z(\boldsymbol{w})) \implies LHS = RHS.$$

and we are done.

QED.

Question 1 :: Part d)

We know that for the forward, we sum over $|\mathcal{T}|$ items for each $t_n \in \mathcal{T}$, for each $n \in \{2, ..., N\}$ therefore it is clear that we have $O(N * |\mathcal{T}|^2)$ time complexity.

In part b), we have shown that we can calculate both $H_U(T_{\boldsymbol{w}})$ and $Z(\boldsymbol{w})$ in a single pass of the forward algorithm using the specified lifting strategy. This means we can calculate $H_U(T_{\boldsymbol{w}})$ and $Z(\boldsymbol{w})$ in $O(N * |\mathcal{T}|^2)$ time and then by part c) it follows that since log, + and / are all operations that can be run in $O(1)$, time that $H(T_{\boldsymbol{w}})$ can also be computed in $O(N * |\mathcal{T}|^2)$ time.

Also we know that the gradient of $H(T_{\boldsymbol{w}})$ w.r.t $\boldsymbol{\theta}$ can be computed in $O(N * |\mathcal{T}|^2)$ time using backpropagation since thanks to the forward algorithm, the forward pass can be computed in $O(N * |\mathcal{T}|^2)$ time, then the backpass just multiplies and sums the computed values so by the *magic of backpropagation*, the overall time complexity is the same. I.e the gradient can also be computed in $O(N * |\mathcal{T}|^2)$ time.

Question 2

Question 2 :: Part a)

By contradiction, suppose the first POS tagging of length N popped from the priority queue is not the best POS tagging. I.e There exists a POS tagging of length N that has a higher (less negative) score. Clearly this tagging could not have been in the queue at the time the first length N POS tagging was popped, or else it would itself have been popped instead. This must mean that we have not "visited" some of its nodes yet.

Clearly if none of the nodes for this alternative tagging are in the queue at the time of popping the first length N tagging, the either we have visited them all (in which case we just get the alternative tagging is equal to the first length N tagging) or the tagging is obviously worse scoring than the first length N tagging because a single one of its edges has a more negative score than the entire length N tagging which gets first popped. So we can assume that a non-empty subset of the alternative taggings nodes are in the queue at the time of popping the first length N tagging. However if this is the case, then since we choose to pop the length N tagging over this shorted tagging, and adding more tags to a tagging will only make the score more negative, then it is clear that the alternative tagging must have a more negative score and we arrive at a contradiction.

So we can see that clearly the first length N tagging popped from the queue must be the best (least negative) scoring complete (length N) tagging.

Question 2 :: Part b)

Proceed by induction on $i :=$ row index of γ, If we consider the initial step of Dijkstra's (ignoring the first row of γ) then $\forall t' \in \mathcal{T}$ we do:

$$\gamma[1, t'] \leftarrow max(-\infty, \omega(BOT, t', \boldsymbol{w}) + 0).$$
$$\implies \forall t' \in \mathcal{T} \quad \gamma[1, t'] = \omega(BOT, t', \boldsymbol{w}).$$

This is what we instantiated the first of γ to be in the Viterbi algorithm, so we see that the first row computed is clearly the same. I.e we have proved the base case for $i = 1$.

Now assume true for $i = k \in \mathbb{N}$, i.e

$$\forall t' \in \mathcal{T} \quad \gamma_D[k, t'] = \gamma_V[k, t'].$$

Now if we run Dijkstra's until the queue is empty, then it is clear that:

$$\forall t' \in \mathcal{T} \quad \gamma_D[k+1, t'] = max_{t'' \in \mathcal{T}}(\gamma_D[k, t''] + \omega[t'', t']).$$

And then the forward Viterbi gives:

$$\gamma_V[k+1, t'] = max_{t'' \in \mathcal{T}}(\omega(t'', t', \boldsymbol{w}) + \gamma_V[k, t']).$$

So clearly by our induction hypothesis if follows that:

$$\forall t'' \in \mathcal{T} \quad \gamma_D[k, t''] = \gamma_V[k, t''] \implies \forall t' \in \mathcal{T} \quad \gamma_D[k+1, t'] = \gamma_V[k+1, t'].$$

and we are done QED.

Question 2 :: Part c)

Let us implement the priority queue as a **heap**. This means push will have $O(\log(m))$ time complexity and pop will have $O(1)$, where m is the number of items in the queue. Since we want to compute an upper bound on the runtime, then we must consider the worst case example. I.e when we don't stop early. In this worst case we must clearly push onto the queue for every edge in the graph, and the queue will be linear in the number of vertices, so it follows that if we have $|\mathcal{T}||\boldsymbol{w}|$ vertices and $|\mathcal{T}|^2|\boldsymbol{w}|$ edges (ignoring BOT and EOT) then we will have a runtime complexity of:

$$O(|\mathcal{T}|^2 |\boldsymbol{w}| \log(|\mathcal{T}| |\boldsymbol{w}|)).$$

Comparing this with the Viterbi algorithm, which has a runtime of $O(|\mathcal{T}|^2 |\boldsymbol{w}|)$ we see that in the worst case scenario, Dijkstras is slower.

However worse case runtime aside, since Dijkstra's is a greedy algorithm and has the early stopping ability, then in reality the average runtime of Dijkstra's could be a lot faster depending on the shape of the problem. In applications where we can be more lenient about consistent speeds, but just want overall faster runtime, Dijkstra's could be favourable.

Question 2 :: Part d)

No. Dijkstra's does **NOT** always calculate the best POS tagging in the:

$$\mathbb{R} = \langle \mathbb{R} \cup \{-\infty, +\infty\}, \oplus_{\log}, +, +\infty, 0 \rangle.$$

semiring. We can see this since inside this semi-ring we can have both positive and negative edge-weights, but our queue update method ($\oplus_{\log}$) always returns a negative score, and so if we want to find the "shortest path" in the context of the semi-ring then clearly this will not work. E.g if we start with positive edge weights, then as soon as we start updating the queue, these positive edge weights initially in the queue will either never get revisited, or always be visited before the new negative scores in the queue, depending on how we order the queue/how we define "best scoring". Either way we have the potential to miss the best scoring path.

More rigorously we note the $\oplus$ does not satisfy superiority and therefore dijkstra's will not work in this instance.

Question 2 :: Part e)

As we just discussed, we require **Superiority**. I.e we either need that:

$$\forall a, b \quad a \leq a \oplus b \wedge b \leq a \oplus b.$$

or

$$\forall a, b \quad a \geq a \oplus b \wedge b \geq a \oplus b.$$

One way to ensure this is to only allow monotonic semirings with the same sign edge weights, i.e all negative or all positive weights.

Question 3

See uploaded .ipynb.