Lecture 21: Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks, part 2

Mark Hasegawa-Johnson
All content CC-BY 4.0 unless otherwise specified.

ECE 537, Fall 2022
1 Review: CTC in Testing Mode

2 Maximum Likelihood Training of a CTC Network

3 Summary
Outline

1. Review: CTC in Testing Mode
2. Maximum Likelihood Training of a CTC Network
3. Summary
Temporal Classification Example: Speech

Temporal classification maps from a sequence of speech frames (top) to a sequence of phoneme or character labels (bottom).

Graves et al., 2006, Figure 1. (c) ICML
Variables in CTC

- **\(\mathbf{x} = [x_1, \ldots, x_T]\)** is the input. It is a sequence of vectors, \(x_t = [x_{1t}, \ldots, x_{mt}]\).

- **\(\mathbf{y} = [y_1, \ldots, y_T]\)** is the network output. It is a sequence of probability vectors, \(y_t = [y_{1u}, \ldots, y_{|L|+1}]\).

- **\(\pi = [\pi_1, \ldots, \pi_T]\)** is the path. It is a sequence of characters, \(y_k^t = p(\pi_t = k | \mathbf{x})\).

- **\(\mathbf{l} = [l_1, \ldots, l_U]\)** = \(B(\pi)\) is the label sequence, \(U \leq T\), which should be compared to the correct label sequence, \(\mathbf{z}\).
In order to express the CTC forward algorithm, we need to define a modified label sequence, $l'$. $l'$ is equal to $l$ with blanks inserted between every pair of letters. Thus if

$$l = [f, e, d],$$

then

$$l' = [-, f, -, e, -, d, -].$$

If the length of $l$ is $|l|$, then the length of $l'$ is $2|l| + 1$. 
CTC Forward Algorithm: Partial Sequences

We also need to define the following partial sequences:

\[ \mathbf{x}_{1:t} = [x_1, \ldots, x_t] \]
\[ \pi_{1:t} = [\pi_1, \ldots, \pi_t] \]
\[ l'_{1:s} = [l'_1, \ldots, l'_s] \]

\[
= \begin{cases} 
  [-, l_1, -, l_2, \ldots, l_{s/2}] & s \text{ even} \\
  [-, l_1, -, l_2, \ldots, l_{(s-1)/2}, -] & s \text{ odd}
\end{cases}
\]
The CTC Forward Algorithm

Definition:

\[ \alpha_t(I'_{1:s}) \equiv p(I'_{1:s} | x_{1:t}) \]
The CTC Forward Algorithm

1. Initialize:

\[ \alpha_t([-]) = y^1_1 \]
\[ \alpha_t([- , l_1]) = y^1_{l_1} \]

2. Iterate:

\[ \alpha_t(l'_1:s) = \begin{cases} 
(\alpha_{t-1}(l'_1:s) + \alpha_{t-1}(l'_1:s-1)) \times y_t^{l'_s} \\
\text{................. if } l'_s = - \text{ or } l'_s = l'_s-2 \\
(\alpha_{t-1}(l'_1:s) + \alpha_{t-1}(l'_1:s-1) + \alpha_{t-1}(l'_1:s-2)) \times y_t^{l'_s} \\
\text{................. otherwise}
\end{cases} \]

3. Terminate:

\[ p(l_{1:U}|x) = \alpha_T(l'_1:2U) + \alpha_T(l'_1:2U+1) \]
The CTC Forward Algorithm

Graves et al., 2006, Fig. 3. (c) ICML
Outline

1. Review: CTC in Testing Mode

2. Maximum Likelihood Training of a CTC Network

3. Summary
The CTC Loss

The CTC loss function is the negative log probability of the correct label sequence given the waveform:

$$\mathcal{L}_{CTC} = - \ln p(z|x)$$

This is similar to cross entropy, but differentiating it is more complicated, since the correct label sequence is $z = [z_1, \ldots, z_U]$, while the speech sequence is $x = [x_1, \ldots, x_T]$.
We want to train the network to maximize the probability of the correct labeling, \( p(z|x) \).

The most computationally efficient way to calculate \( p(z|x) \) is the forward algorithm:

\[
p(z|x) = \alpha_T(z'_{1:2U}) + \alpha_T(z'_{1:2U+1}),
\]

... but that form is not easy to differentiate.

The expansion over all possible paths is not a computationally efficient way to calculate \( p(z|x) \), but it’s easier to differentiate:

\[
p(z|x) = \sum_{\pi \in B^{-1}(z)} p(\pi|x) = \sum_{\pi \in B^{-1}(z)} \prod_{t=1}^{T} y_{\pi_t}^{t}
\]
Differentiating the CTC Loss

Remember that the basic principle of back-propagation is the chain rule. If we want \( \frac{d\mathcal{L}}{dw} \), we can find it as

\[
\frac{d\mathcal{L}}{dw} = \sum_{\tau=1}^{T} \sum_{k=1}^{|L|+1} \left( \frac{d\mathcal{L}}{dy_{\tau}^{k}} \right) \left( \frac{\partial y_{\tau}^{k}}{\partial w} \right),
\]

- \( \frac{\partial y_{\tau}^{k}}{\partial w} \) is the same as for any other RNN, so it’s uninteresting.
- \( \frac{d\mathcal{L}}{dy_{\tau}^{k}} \) is unique to CTC. Let’s derive it.
Differentiating the CTC Loss

\[ \mathcal{L}_{\text{CTC}} = - \ln p(z|x) = - \ln \left( \sum_{\pi \in B^{-1}(z)} \prod_{t=1}^{T} y_{\pi_t}^t \right) \]

Therefore

\[ \frac{d \mathcal{L}}{dy_{k}^T} = \left( \frac{-1}{p(z|x)} \right) \left( \frac{dp(z|x)}{dy_{k}^T} \right) = \left( \frac{-1}{p(z|x)} \right) \left( \frac{1}{y_{k}^T} \right) \left( \sum_{\pi \in B^{-1}(z), \pi_T = k} \prod_{t=1}^{T} y_{\pi_t}^t \right) \]

- The sum in the last line is over all paths that are valid expansions of the correct transcription, and for which \( \pi_T = k \).
- The \( \frac{1}{y_{k}^T} \) comes from the derivative of the product:

\[ \frac{d}{dy} xyz = xz = \frac{1}{y} xyz \]
Differentiating the CTC Loss

\[
\frac{dL}{dy^T_k} = \left( \frac{-1}{p(z|x)} \right) \left( \frac{1}{y^T_k} \sum_{\pi \in B^{-1}(z), \pi_\tau = k} \prod_{t=1}^{T} y^{t}_{\pi_t} \right)
\]

The sum in the last line is over all paths that are valid expansions of the correct transcription, and for which \( \pi_\tau = k \). This has a nice Bayesian interpretation:

\[
\frac{dL}{dy^T_k} = \left( \frac{-1}{p(z|x)} \right) \left( \frac{1}{y^T_k} p(z, \pi_\tau = k | x) \right)
\]

\[
= \frac{-1}{y^T_k} p(\pi_\tau = k | z, x)
\]
The CTC Gamma Probability

Just as for any other HMM, let’s define a gamma probability, $\gamma_\tau(k) = p(\pi_\tau = k | z, x)$. Then

$$\frac{dL}{dy^\tau_k} = -\frac{\gamma_\tau(k)}{y^\tau_k},$$

where

$$\gamma_\tau(k) = p(\pi_\tau = k | z, x) = \frac{1}{y^\tau_k} \alpha_\tau(z'_1:s) \beta_\tau(z'_{s:(2U+1)}) \quad (1)$$

- $\beta_t(z'_{s:2U+1}) = p(z'_{s:(2U+1)} | x_t:T)$

Notice that $\alpha_\tau(z'_{1:s})$ and $\beta_\tau(z'_{s:(2U+1)})$ both include the fact that the network is producing $z'_s = k$ at time $\tau$. To compensate for that duplication, Eq. (1) has a $\frac{1}{y^\tau_k}$ factor.
The CTC Forward-Backward Algorithm

$$\alpha_t(z'_{1:s})$$ is the probability of the best path up to (and including) state $$z'_s$$ at time $$t$$. $$\beta_t(z'_{s:(2U+1)})$$ is the probability of the best path starting from state $$z'_s$$ at time $$t$$. 

Graves et al., 2006, Fig. 3. (c) ICML
At the start of training (a), $y_k^t = 0$ for all $k$ except the blank symbol. $\gamma_t(k)$, is determined mostly by the known sequence of the correct labels, $z$.

After some training (b), $y_k^t$ has started to converge. Its convergence guides the forward-backward algorithm, so $\gamma_t(k)$ is also much more localized.

When fully converged, $y_k^t \approx \gamma_t(k)$, so the error is nearly zero.
Outline

1. Review: CTC in Testing Mode
2. Maximum Likelihood Training of a CTC Network
3. Summary
Conclusions

- A CTC network is trained to minimize

\[ \mathcal{L}_{\text{CTC}} = -\ln p(z|x) \]

- Differentiating, we discover that

\[ \frac{d\mathcal{L}}{dy_k^\tau} = \frac{1}{y_k^\tau} p(\pi_\tau = k|z, x) \]

- \( p(\pi_\tau = k|z, x) \) can be computed using the forward-backward algorithm.

- Even in the very first epoch of training, the known sequence \( z \) distributes error uniformly across the waveform. For this reason, CTC training converges smoothly and quickly (compared, e.g., to Transformer loss).