First, let’s write down the forward pass. The variables are:

- $\mathbf{x}_s$ the input sequence, encoded using one-hot encoding. Denote it by $x_t$.
- $\mathbf{h}_s$ the hidden state (a vector), at each time step. Denote it by $h_t = \tanh(W^{xh}x_t + W^{hh}h_{t-1})$
- $\mathbf{y}_s$ the output layer. Denote it by $y_t = \text{softmax}(y_t)$
- $\mathbf{p}_s$ the output of the softmax. Denote it by $\hat{y}_t = \text{softmax}(y_t)$
- loss the cost/loss function. $\text{Cost} = -\sum_t \log(\sum_k \hat{y}_k^t x_t^k)$

Now, let’s go line by line and interpret those. We will often use e.g. $\partial \text{Cost}_t / \partial h_t$ to denote the contribution from time-step $t$ to the cost function, with $C = \sum_t C_t$ for

$$C_t = \log(\sum_k \hat{y}_k^t x_t^k).$$

```python
dy = np.copy(ps[t])
dy[targets[t]] -= 1 # backprop into y
This is just the derivative of the softmax for $\text{Cost}_t$:
$$\frac{\partial \text{Cost}_t}{\partial \mathbf{y}} = \hat{y} - x$$
Note that $x$ is one-hot encoded, so that it’s mostly zeros, with only a single 1 at coordinate targets[t]. That’s why we first set dy to ps (i.e., the $\hat{y}$), and then subtract $y$.

dWhy += np.dot(dy, hs[t].T)
dby += dy
This corresponds to the $t$-th component of the derivatives wrt $W^{hy}$ and $b^y$:
$$\frac{\partial \text{Cost}_t}{\partial W^{hy}} = \frac{\partial \text{Cost}_t}{\partial \mathbf{y}_t} \frac{\partial \mathbf{y}_t}{\partial W^{hy}} = \frac{\partial \text{Cost}_t}{\partial \mathbf{y}_t} h_t^T$$
$$\frac{\partial \text{Cost}_t}{\partial W^{hy}} = \frac{\partial \text{Cost}_t}{\partial \mathbf{y}_t} \frac{\partial \mathbf{y}_t}{\partial b^y} = \frac{\partial \text{Cost}_t}{\partial \mathbf{y}_t} 1 = \frac{\partial \text{Cost}_t}{\partial \mathbf{y}_t}$$
```

```
dh = np.dot(Why.T, dy) + dhnext
This is tricky. We want to account for the influence of $h_t$ on both $\text{Cost}_t$ and $\text{Cost}_{(t+1):\text{end}}$.
$$\frac{\partial \text{Cost}_{t:\text{end}}}{\partial h_t} = \frac{\partial \text{Cost}_t}{\partial h_t} + \frac{\partial \text{Cost}_{(t+1):\text{end}}}{\partial h_t} = \frac{\partial \text{Cost}_t}{\partial \mathbf{y}_t} \frac{\partial \mathbf{y}_t}{\partial h_t} + dhnext$$
dhraw = (1 - hs[t] * hs[t]) * dh
$$\frac{\partial \text{Cost}_{t:\text{end}}}{\partial \mathbf{h}_t} = (1 - h_t^2) \frac{\partial \text{Cost}_{t:\text{end}}}{\partial h_t}$$
```
The following:

\[
\begin{align*}
\text{dbh} & += \text{dhraw} \\
\text{dWxh} & += \text{np.dot(dhraw, xs[t].T)} \\
\text{dWhh} & += \text{np.dot(dhraw, hs[t-1].T)} \\
\end{align*}
\]

are similar to what we already had. Note that

\[
\frac{\partial \text{Cost}_t}{\partial h_t} = \frac{\partial \text{Cost}}{\partial h_t}
\]

since \( h_t \) cannot influence components of the cost that come before it in time.

Finally, we compute \( \text{dhnext} \), which must be \( \frac{\partial \text{Cost}_{t\text{end}}}{\partial h_{t-1}} \) in order for our earlier definition to work. Now

\[
\frac{\partial \text{Cost}_{t\text{end}}}{\partial h_{t-1}} = \frac{\partial \text{Cost}_{t\text{end}}}{\partial h_{raw_t}} \frac{\partial h_{raw_t}}{\partial h_{t-1}}
\]

This is exactly what the following line does.

\[
\text{dhnext} = \text{np.dot(Whh.T, dhraw)}
\]

The following is self-explanatory:

\[
\text{for dparam in [dWxh, dWhh, dWhy, dbh, dby]:} \\
\quad \text{np.clip(dparam, -5, 5, out=dparam)} \quad \# \text{clip to mitigate exploding gradients}
\]

In the loop, we are adding up all the contributions to the gradients from all the time-steps \( t \).