Notes on Underflow Problems in the Viterbi Algorithm, and a Solution using Log Probabilities

Michael Collins

This note describes a simple modification to the Viterbi algorithm for HMMs that is very important in practice. You'll need to make this modification when implementing your solution to the programming problem for Homework 1. Figure 1 shows the original algorithm as presented in the notes and lectures. Figure 2 shows the modified algorithm, with changes highlighted in red.

The motivation for the change is as follows. In the original algorithm we are taking probabilities and multiplying them, for example in the expression

$$\pi(k-1, w, u) \times q(v|w, u) \times e(x_k|v)$$

The result is that as k gets larger, the values $\pi(k, u, v)$ will be calculated as products of more and more probabilities $q(\ldots)$ and $e(\ldots)$. In some cases, we may eventually see underflow problems, where the $\pi(k, u, v)$ values become too small for the floating-point precision of the underlying implementation. This will lead to very small $\pi(k, u, v)$ values being set to zero, making the underlying dynamic program inexact to the point that it fails.

A simple solution to this problem is to modify the dynamic program to search for

$$\max_{y_1 \cdots y_{n+1}} \log p(x_1 \ldots x_n, y_1 \ldots y_{n+1})$$

and

$$\arg \max_{y_1 \cdots y_{n+1}} \log p(x_1 \ldots x_n, y_1 \ldots y_{n+1})$$

rather than

$$\max_{y_1 \cdots y_{n+1}} p(x_1 \ldots x_n, y_1 \ldots y_{n+1})$$

and

$$\arg \max_{y_1 \cdots y_{n+1}} p(x_1 \ldots x_n, y_1 \ldots y_{n+1})$$

Note that we are now using $\log$ probabilities. We clearly have

$$\arg \max_{y_1 \cdots y_{n+1}} p(x_1 \ldots x_n, y_1 \ldots y_{n+1}) = \arg \max_{y_1 \cdots y_{n+1}} \log p(x_1 \ldots x_n, y_1 \ldots y_{n+1})$$

because $\log$ is a strictly monotonically increasing function. So the tag sequence given as output from the two algorithms will be identical.

In addition, because of the identity $\log(a \times b) = \log a + \log b$ for any positive values a and b, if we have

$$p(x_1 \ldots x_n, y_1 \ldots y_{n+1}) = \prod_{i=1}^{n+1} q(y_i|y_{i-2}, y_{i-1}) \prod_{i=1}^{n} e(x_i|y_i)$$

then

$$\log p(x_1 \ldots x_n, y_1 \ldots y_{n+1}) = \sum_{i=1}^{n+1} \log q(y_i|y_{i-2}, y_{i-1}) + \sum_{i=1}^{n} \log e(x_i|y_i)$$

The algorithm in Figure 2 therefore replaces $q(\ldots)$ and $e(\ldots)$ by $\log q(\ldots)$ and $\log e(\ldots)$, and replaces multiplication by addition. Log probabilities do not in general suffer from the underflow problem.

One issue with the new algorithm is that $\log 0 = -\infty$. Some care is required to deal with probability values equal to 0. A simple solution is to set $\log 0 = -B$ where B is a large number.

Input: a sentence $x_1 \ldots x_n$, parameters $q(s|u,v)$ and $e(x|s)$.
Definitions: Define $\mathcal{K}$ to be the set of possible tags. Define $\mathcal{K}_{-1} = \mathcal{K}_0 = \{*\}$, and $\mathcal{K}_k = \mathcal{K}$ for $k = 1 \ldots n$.
Initialization: Set $\pi(0, *, *) = 1$.
Algorithm:

```
• For $k = 1 \ldots n$,
    – For $u \in \mathcal{K}_{k-1}$, $v \in \mathcal{K}_k$,
        $\pi(k,u,v) = \max_{w \in \mathcal{K}_{k-2}} (\pi(k-1,w,u) \times q(v|w,u) \times e(x_k|v))$
        $bp(k,u,v) = \arg\max_{w \in \mathcal{K}_{k-2}} (\pi(k-1,w,u) \times q(v|w,u) \times e(x_k|v))$
• Set $(y_{n-1}, y_n) = \arg\max_{u \in \mathcal{K}_{n-1}, v \in \mathcal{K}_n} (\pi(n,u,v) \times q(\text{STOP}|u,v))$
• For $k = (n-2) \ldots 1$,
        $y_k = bp(k+2, y_{k+1}, y_{k+2})$
• **Return** the tag sequence $y_1 \ldots y_n$
```

Figure 1: The Viterbi Algorithm with backpointers.

Input: a sentence $x_1 \ldots x_n$, parameters $q(s|u,v)$ and $e(x|s)$.
Definitions: Define $\mathcal{K}$ to be the set of possible tags. Define $\mathcal{K}_{-1} = \mathcal{K}_0 = \{*\}$, and $\mathcal{K}_k = \mathcal{K}$ for $k = 1 \ldots n$.
Initialization: Set $\pi(0, *, *) = 0$.
Algorithm:

```
• For $k = 1 \ldots n$,
    – For $u \in \mathcal{K}_{k-1}$, $v \in \mathcal{K}_k$,
          $\pi(k,u,v) = \max_{w \in \mathcal{K}_{k-2}} \left(\pi(k-1,w,u) + \log q(v|w,u) + \log e(x_k|v)\right)$
          $bp(k,u,v) = \arg\max_{w \in \mathcal{K}_{k-2}} \left(\pi(k-1,w,u) + \log q(v|w,u) + \log e(x_k|v)\right)$
• Set $(y_{n-1}, y_n) = \arg\max_{u \in \mathcal{K}_{n-1}, v \in \mathcal{K}_n} \left(\pi(n,u,v) + \log q(\text{STOP}|u,v)\right)$
• For $k = (n-2) \ldots 1$,
          $y_k = bp(k+2, y_{k+1}, y_{k+2})$
• **Return** the tag sequence $y_1 \ldots y_n$
```

Figure 2: The Viterbi Algorithm with backpointers, using log probabilities.