Viterbi pseudocode:

```
def run_viterbi(emission_scores, trans_scores, start_scores, end_scores)
{
 trellis = [LxN]
 emission = emission_scores.transpose() #LxN
 parent = [LxN]

 #Stage 1: initialize first col <s> -> words
 for(j=0 to L-1)
  {
  trellis[j][0] = start_scores[j] + emission[j][0]
  }

 #Stage 2: viterbi dynamic prog
 for(i=1 to N-1)
  {
  for(k=0 to L-1)
   {
   max = -infinity
   r=0
   for(j=0 to L)
    {
    x = trellis[j][i-1] + emission[k][i] + trans_scores[j][k]
              if(x>max):
                max=x; r=j
    }
   trellis[k][i] = max
   parent[k][i] =r
  }
 }

#Stage 3: get the maximum score out by adding last column to the end scores words -> </s>
max = -infinity
p=0
for(j=0 to L-1)
  {
  x = trellis[j][N - 1] + end_scores[j]
  if (x > max):
    max = x; p = j
  }

y=[N]
y[0] =p

#Stage 4: traceback using the parent matrix the parents of the best scores at each stage
for(i=1 to N-1)
   {
      t = y[i - 1]
      c = N - i
      z = int(parent[t][c])
      y[i] = z
              }

y.reverse()
return(max,y)
}
```

Viterbi explanation:

1>Vierbi decoding is picking of the best global hidden state sequence using the dynamic programming approach of getting the next best value using the previous best value.

$T(i, y) = x(y, i, x) + maxy0(t(y0, y)) + T(i - 1, y0)$

2>Viterbi uses the trellis structure to efficiently represent and use all the exponential number of sequences.

The trellis represents the path with maximum probability in position i when we are in a state y(i) = yl and we have observed _x until that position

3>The above pseudocode has 4 stages

4> Stage 1 : is initializing the first column of the trellis using the the transition values from the start tag to all the other L-1 tags. This way we fill out all the L cells of the first column of N columns of words.

5>Stage 2 : Here onwards we calculate the values for columns 1 through N-1 using the previous column values and the corresponding word's emission and transition probabilities of its tag.

At each cell we find the maximum value of the word having a particular tag over all possible previous tags.

This way we fill out till the N-1th column.

6>Stage 3 : After getting the last columns of values, find the sum of those values and the end_scores. This represents the corresponding tags preceding the end tag.And we want the maximum value of the tags that precedes the end tag. This also represents the maximum score for the sequence.

4>stage 4: We get the best sequence of tags by backtracking for the parent of the best value of every column. Initially we would have stored the parents of all the cell values in the parent table(from stage 2).

We use this to traceback until the first tag.