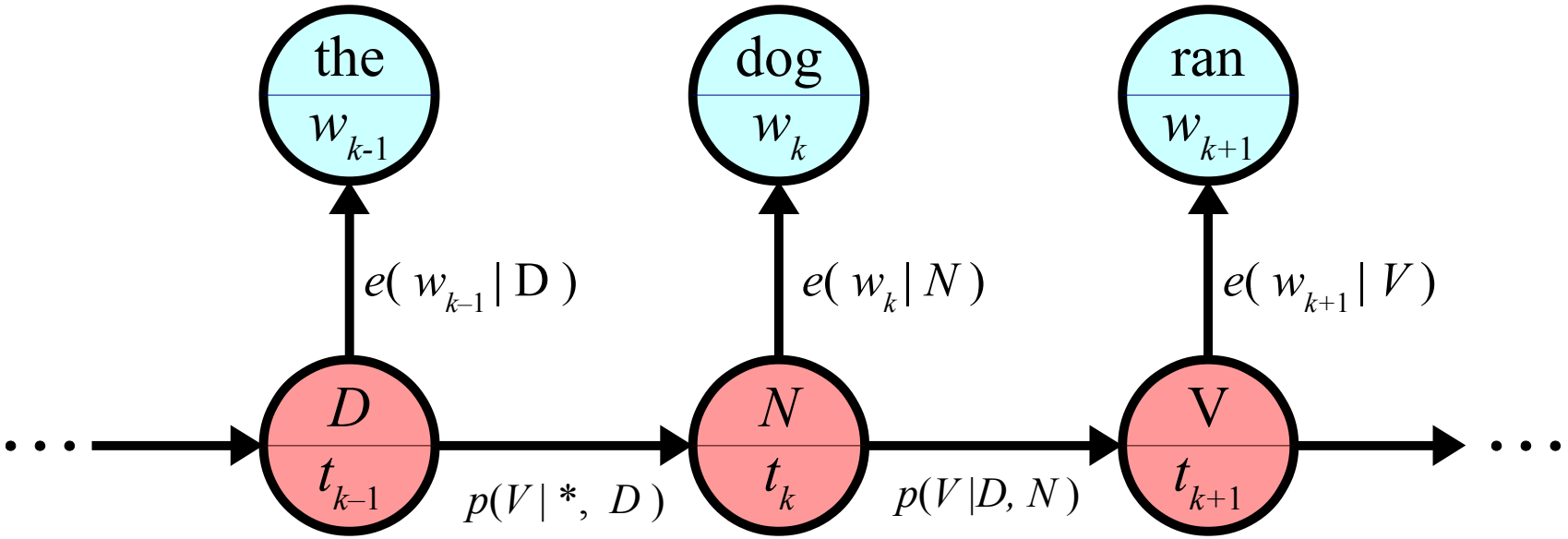
Part of Speech Tagging with HMMs

Abraham Nassar

December 13, 2016

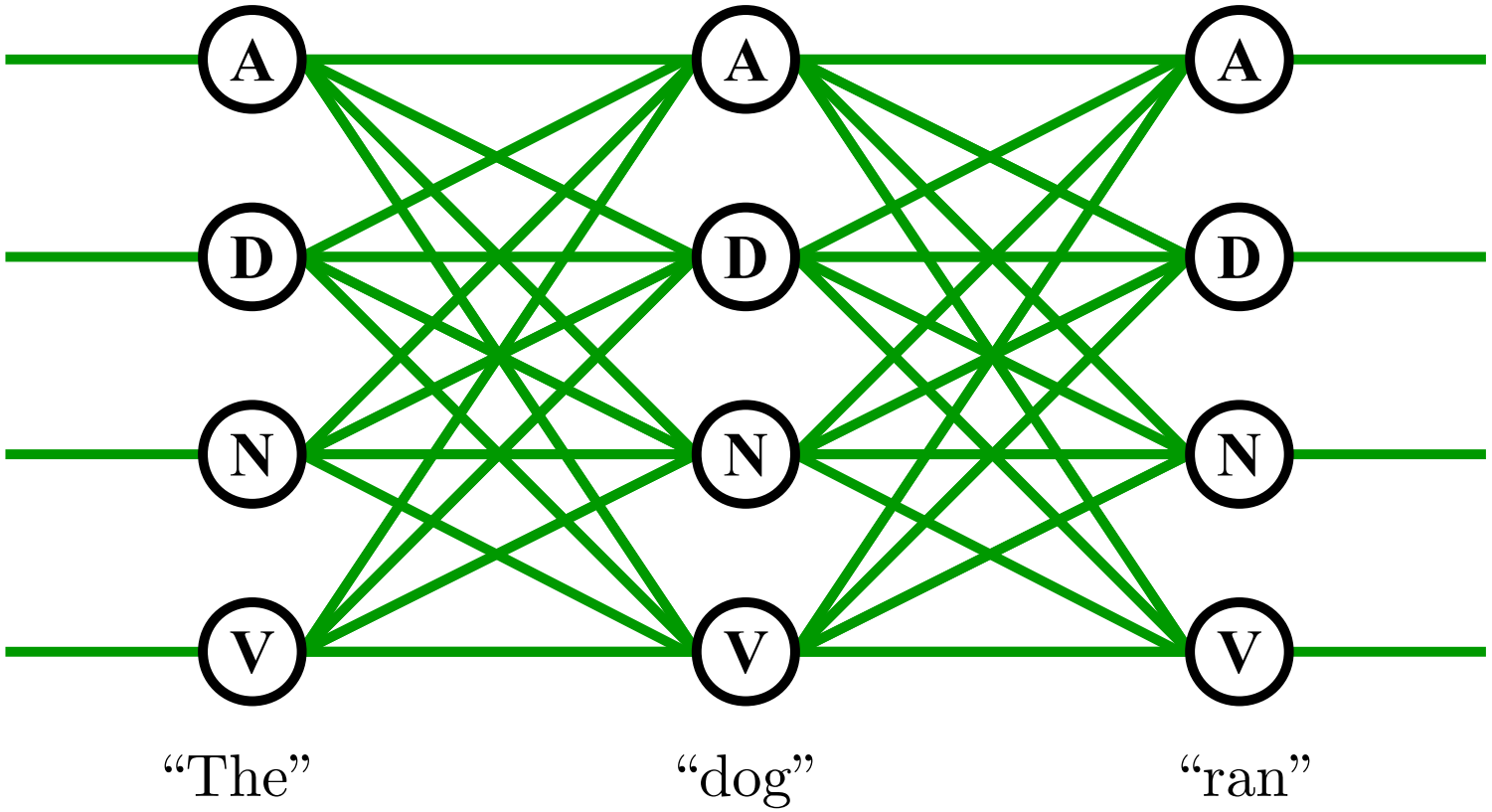
The Problem

Our goal is to take a sentence and find the tag sequence that maximizes probabilities.



The Problem

Brute Force: Each tag would be checked for each word.
The number of calculations: $(\#\text{tags})^{\#\text{words}}$



Definitions

- Let $\mathcal{W}$ be the set of words
- Let $\mathcal{T}$ be the set of tags union with $\{END\}$.
- Let $S(k, u, v)$ the set of all tags of length k such that $y_{k-1} = u$, $y_k = v$.
- For index $k \in \{1, 2, \ldots, n\}$ and tags u, v we can define

$$\pi(k, u, v) = \max_{\{y_i\} \in S(k,u,v)} \prod_{i=1}^{k} q(y_i | y_{i-2}, y_{i-1}) \prod_{i=1}^{k} e(x_i | y_i)$$

The Basic Viterbi Algorithm

Input: A sentence $w_1, w_2, \ldots, w_n$, transition probabilities $q(t_k|t_i, t_j)$ emission probabilities, $e(w_j|t_j)$
Initialization: $\pi(0|*,*) = 1$ and $\pi(0|u,v) = 0, \ \forall u, v \in \mathcal{T}$ where $u \neq *$ or $v \neq *$
Algorithm:

- for $k = 1, 2, \ldots, n$
  - for $(u, v) \in \mathcal{T} \times \mathcal{T}$,

$$\pi(k, u, v) = \max_{t \in \mathcal{T}} \left[\pi(k-1, t, u) \cdot q(v|t, u) \cdot e(w_k|v)\right]$$

- **Return**:

$$\max_{u \in \mathcal{T}, v \in \mathcal{T}} \left[\pi(n, u, v) \cdot q(END|u, v)\right]$$

The Basic Viterbi Algorithm

Input: A sentence $w_1, w_2, \ldots, w_n$, transition probabilities $q(t_k|t_i, t_j)$ emission probabilities, $e(w_j|t_j)$
Initialization: $\pi(0|*,*) = 1$ and $\pi(0|u,v) = 0, \ \forall u, v \in \mathcal{T}$ where $u \neq *$ or $v \neq *$
Algorithm:

- for $k = 1, 2, \ldots, n$
  - for $(u, v) \in \mathcal{T} \times \mathcal{T}$,

$$\pi(k, u, v) = \max_{t \in \mathcal{T}} \left[\pi(k-1, t, u) \cdot q(v|t, u) \cdot e(w_k|v)\right]$$

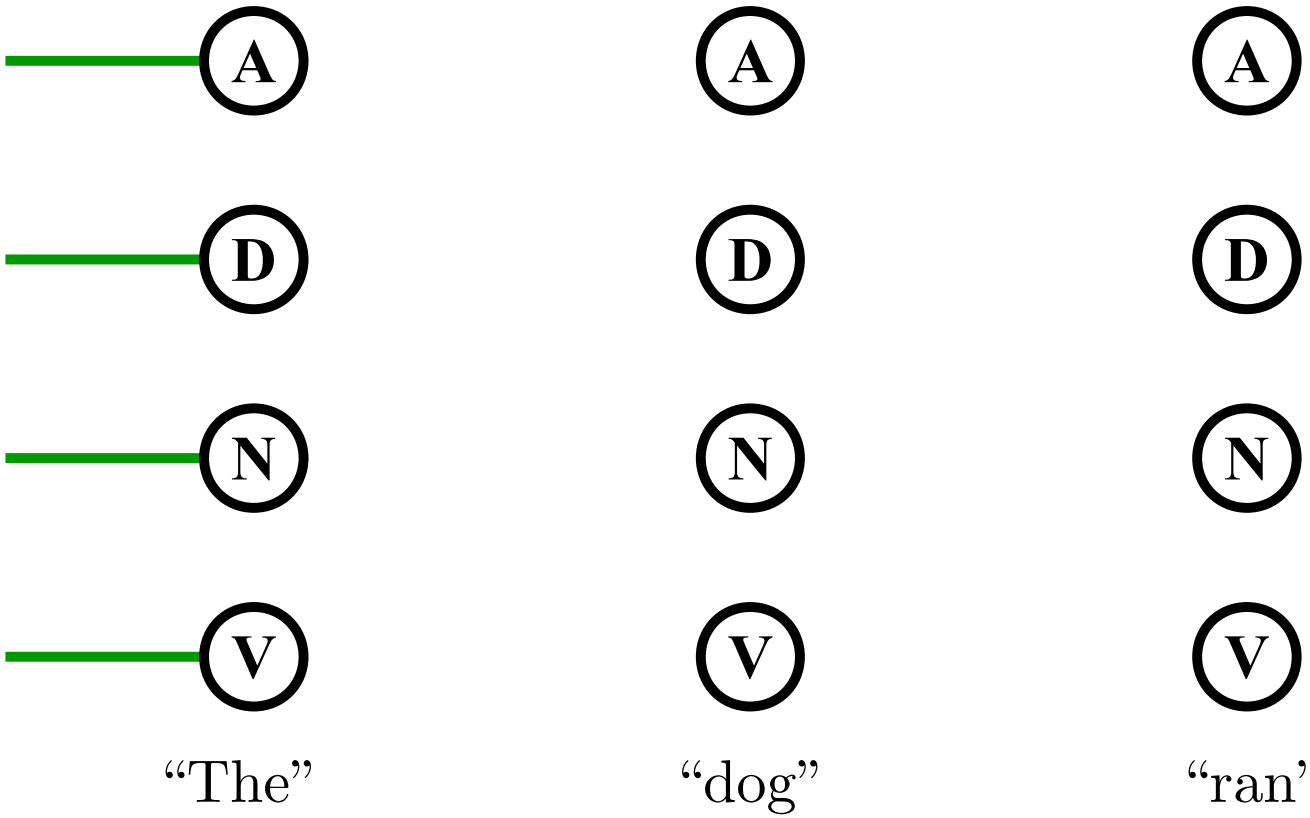
- **Return**:

$$\max_{u \in \mathcal{T}, v \in \mathcal{T}} \left[\pi(n, u, v) \cdot q(END|u, v)\right]$$

NOTE: This returns maximum probability, but we want argmax. So keep track of indices.

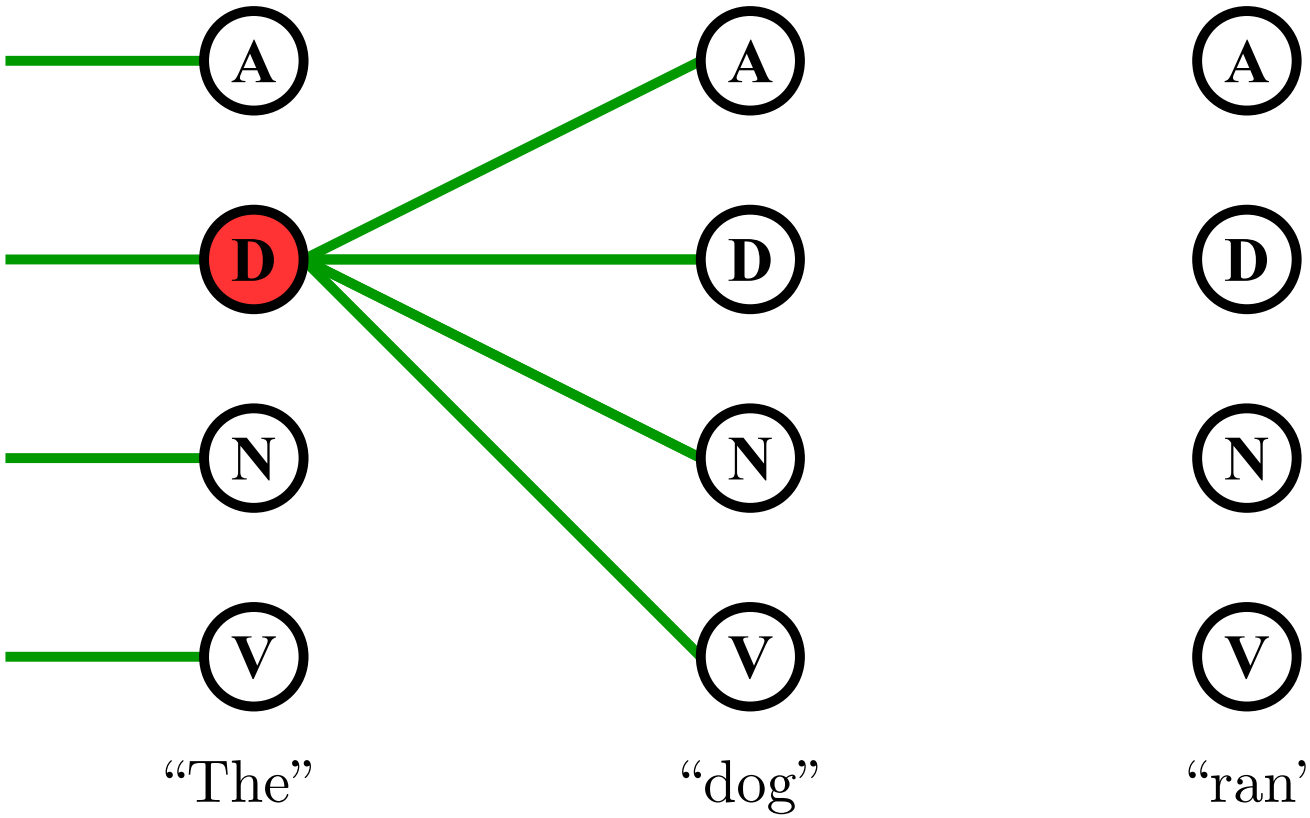
Viterbi Visualized

Viterbi: Find the highest probability word by word.
The number of calculations: $(\#words)\,(\#tags)^3$



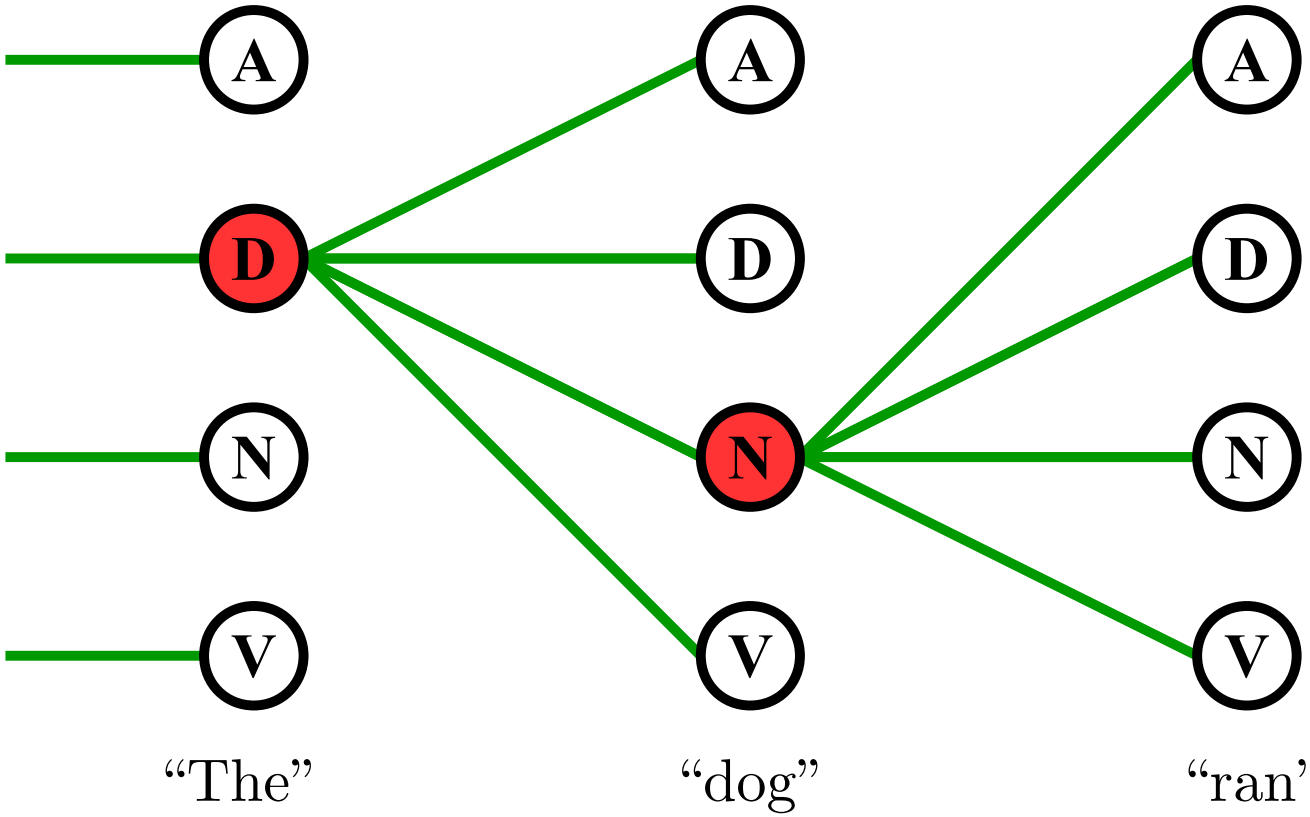
Viterbi Visualized

Viterbi: Find the highest probability word by word.
The number of calculations: $(\#words)\,(\#tags)^3$



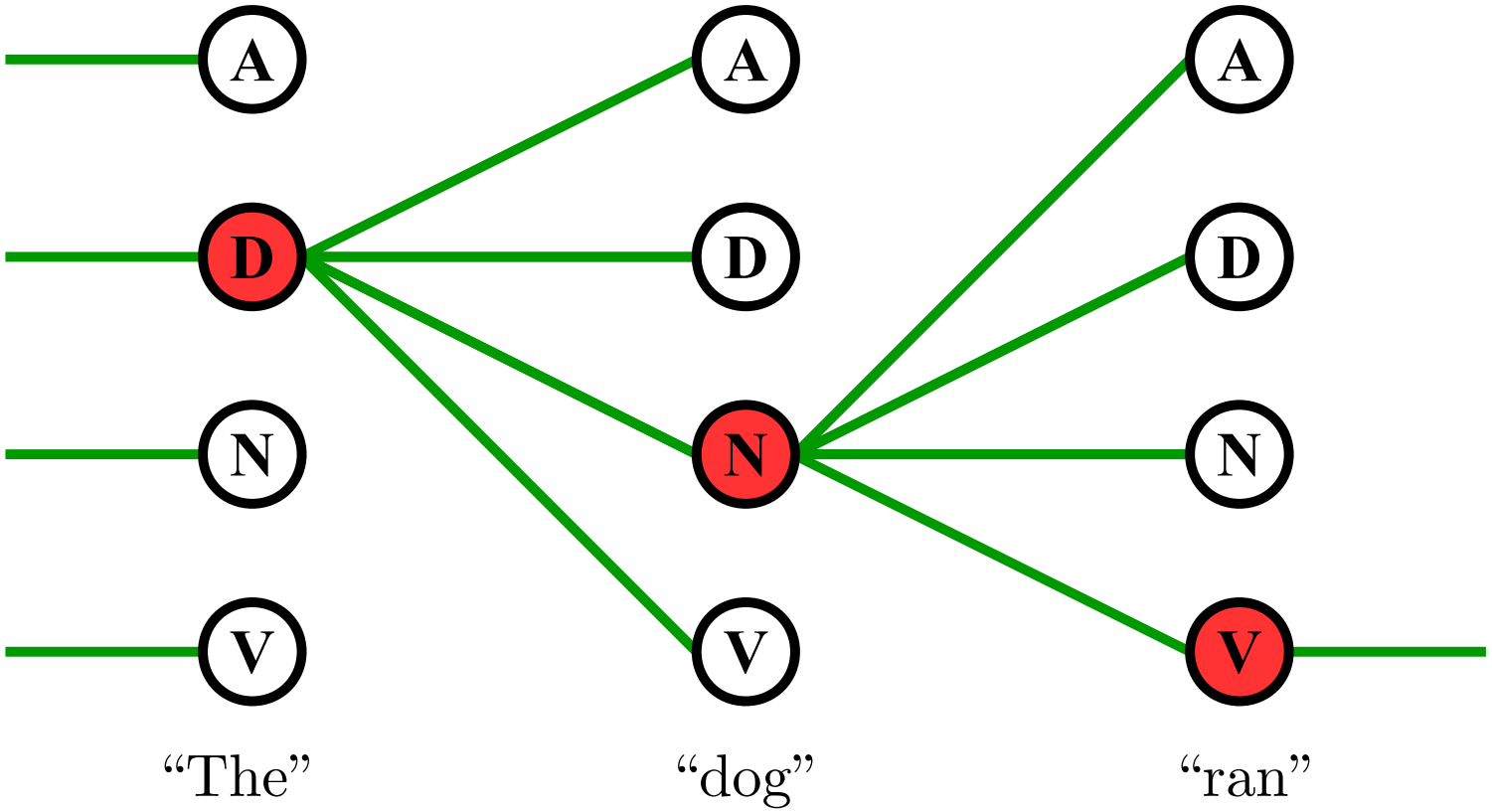
Viterbi Visualized

Viterbi: Find the highest probability word by word.
The number of calculations: $(\#words)\,(\#tags)^3$



Viterbi Visualized

Viterbi: Find the highest probability word by word.
The number of calculations: $(\#words)\,(\#tags)^3$



References and Resources

This presentation was created using the following sources.

- Columbia University's Coursera Course on Natural Language Processing.

  https://www.youtube.com/playlist?list=PLO9y7hOkmmSGSJA8S3gTigcyNDVJ31LLt
- Tagging with Hidden Markov Models by Michael Collins.

  http://www.cs.columbia.edu/~mcollins/courses/nlp2011/notes/hmms.pdf
- A Well-Commented Examples of a Bigram Model By Katrin Erk.

  http://www.katrinerk.com/courses/python-worksheets/hidden-markov-models-for-pos-tagging-in-python