# CS378: Natural Language Processing
# Lecture 5: Hidden Markov Model


Eunsol Choi

Parts of this lecture adapted from Greg Durrett, Yejin Choi, Yoav Artzi

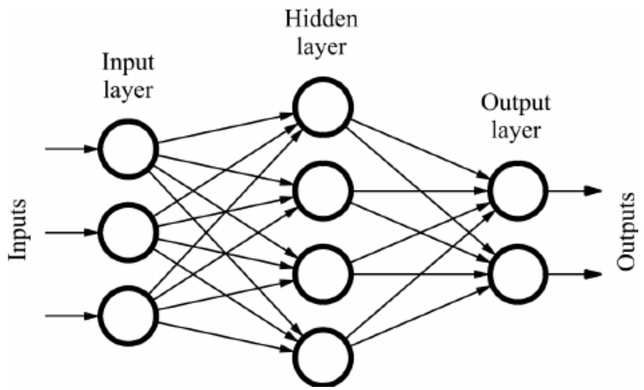
# Today

- Simple feedforward neural network as a classifier
- Sequence Modeling Task
- Hidden Markov Model

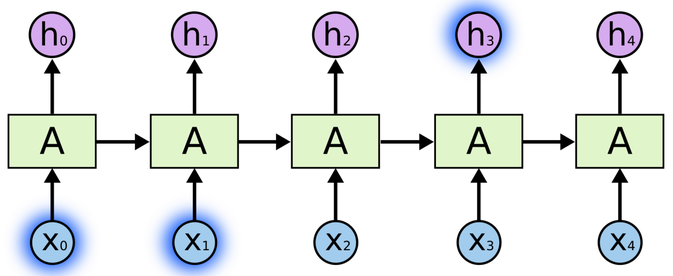
# Neural Networks in NLP

Expects fixed-dimensional input representation

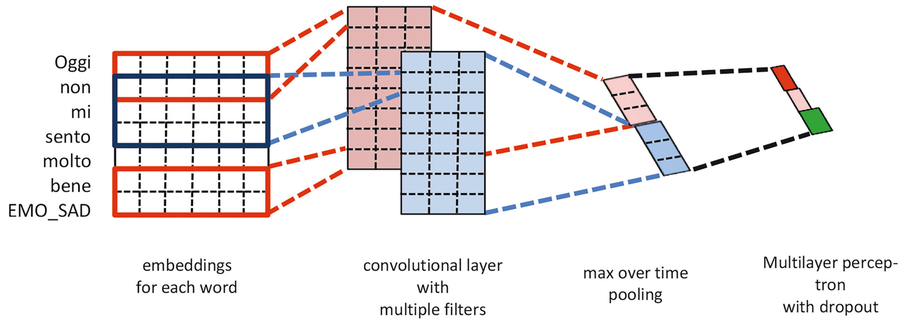
## Feed-forward NNs



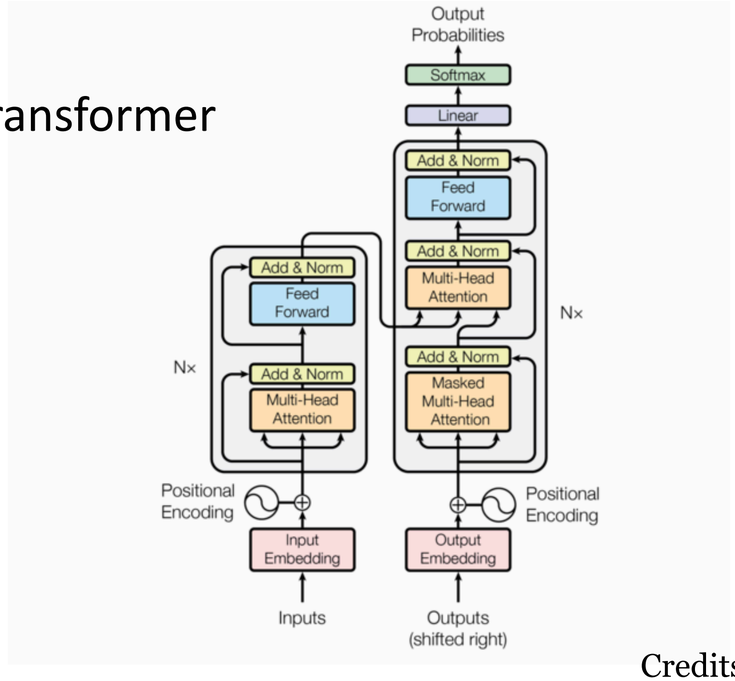

## Recurrent NNs




## Convolutional NNs




## Transformer




Always coupled with word embeddings…

Credits: Princeton NLP course

# Word Embeddings

- So far, we think of words as “one-hot” vectors

*the* = [1, 0, 0, 0, 0, 0, …]

*good* = [0, 0, 0, 1, 0, 0, …]

*great* = [0, 0, 0, 0, 0, 1, …]

# Word Embeddings

- *good* and *great* seem as dissimilar as *good* and *the*
- Neural networks are built to learn sophisticated nonlinear functions of continuous inputs; our inputs is sparse and discrete
- Size of vocabulary grows quickly
  - 500K for broad-coverage domains
  - 13M for Google corpora

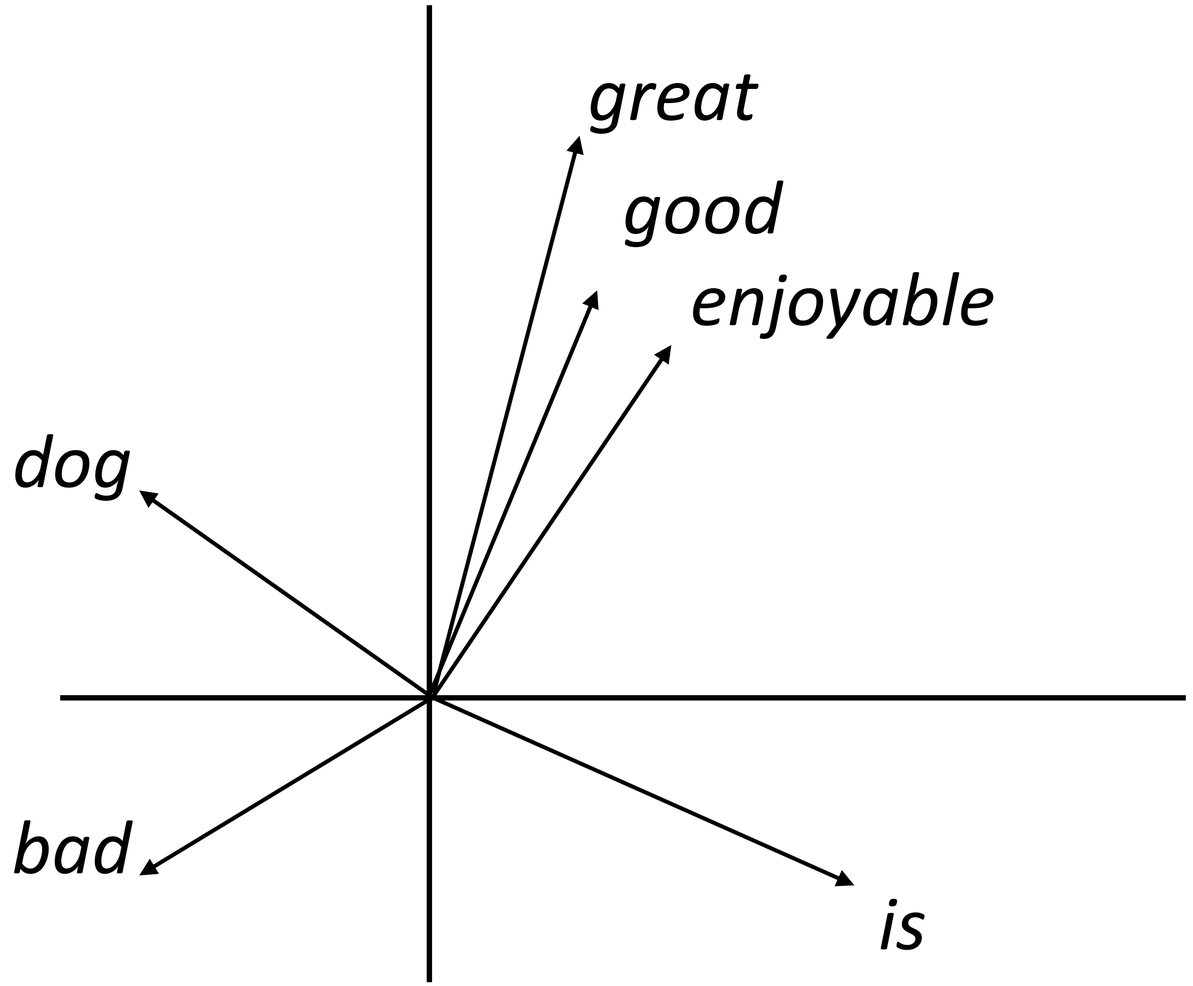
# Word Embeddings

- Want a vector space where similar words have similar embeddings

  *great* ≈ *good*

- In few weeks: come up with a way to produce these word embeddings

- For each word, want "medium" dimensional vector (50-300 dims) representing it

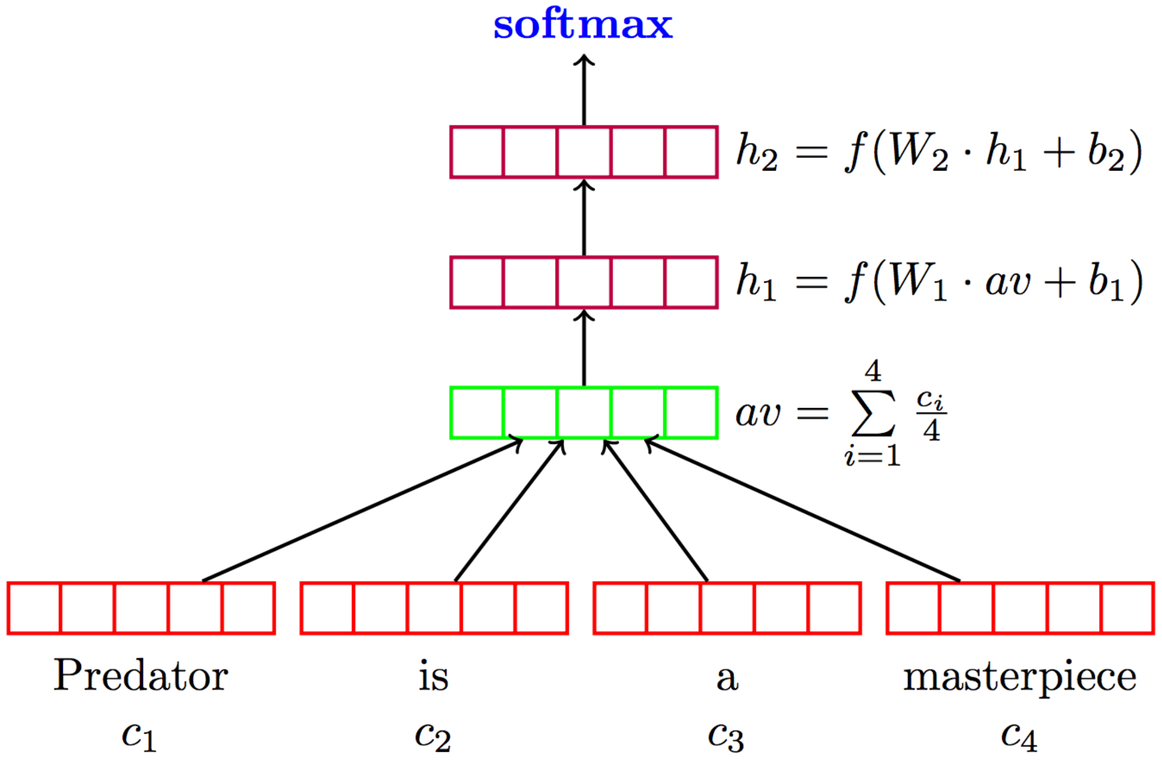


# Using Word Embeddings as feature

- Assume we would like to do sentiment classification again..
- Feature-based models: Bag of words
- How should we use word embeddings?

# Deep Averaging Networks

- Deep Averaging Networks: feedforward neural network on average of word embeddings from input for sentiment classification

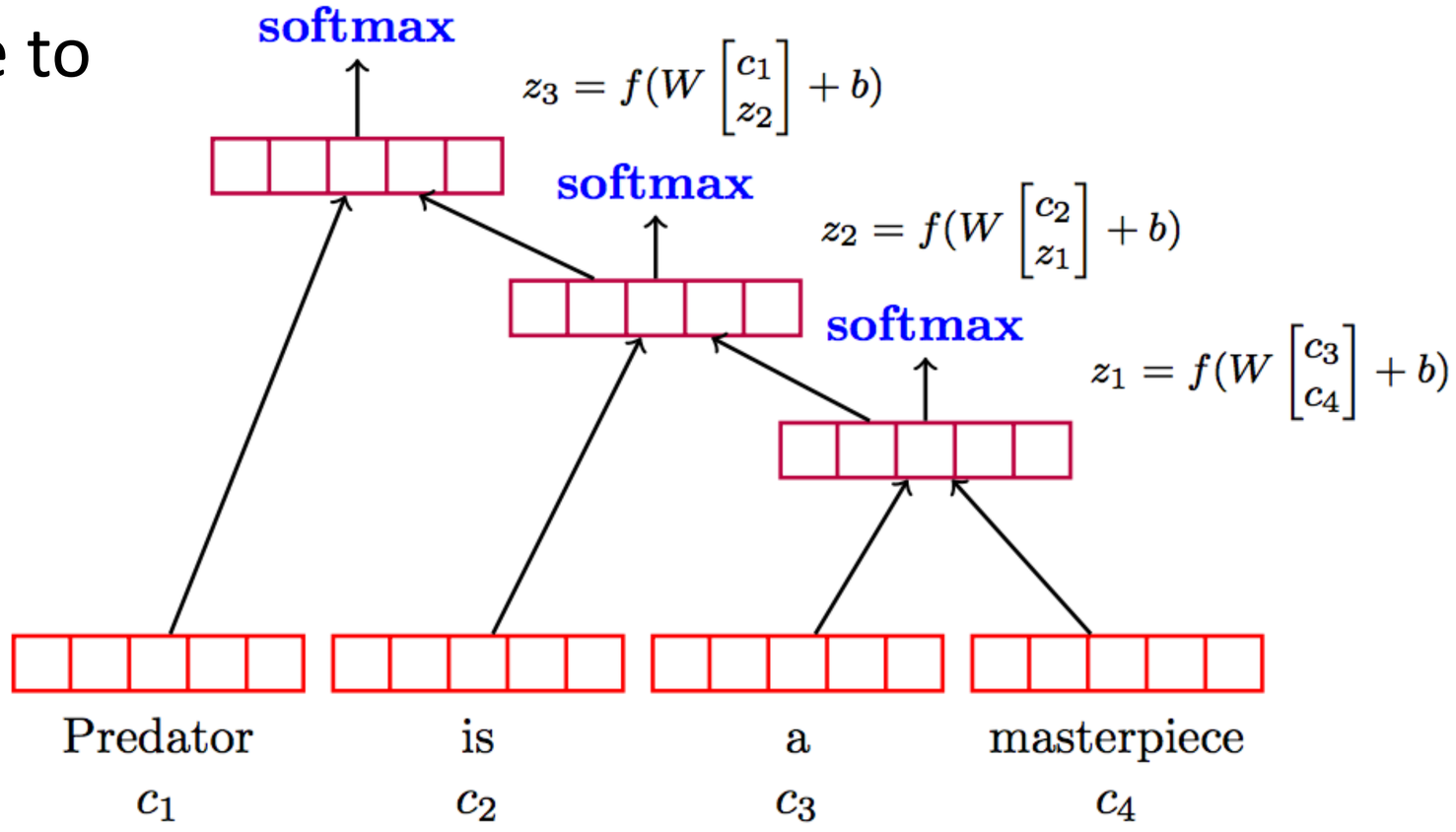



- Limitation?

Iyyer et al. (2015)

# Recursive Neural Network

- Widely-held view: need to model syntactic structure to represent language




Iyyer et al. (2015)

# Deep Averaging Networks

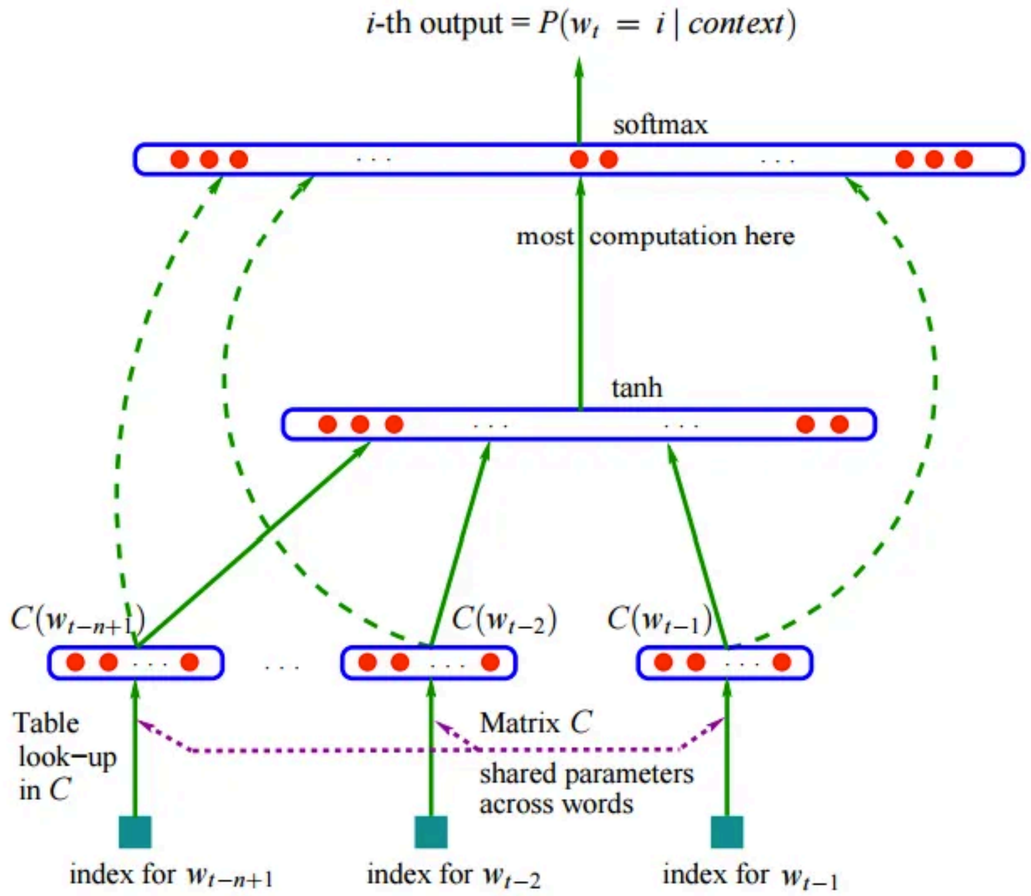
| Sentence | DAN | DRecNN | Ground Truth |
|---|---|---|---|
| who knows what exactly godard is on about in this film, but his words and images do n't have to add up to mesmerize you. | positive | positive | positive |
| it's so good that its relentless, polished wit can withstand not only inept school productions, but even oliver parker's movie adaptation | negative | positive | positive |
| too bad, but thanks to some lovely comedic moments and several fine performances, it's not a total loss | negative | negative | positive |
| this movie was not good | negative | negative | negative |
| this movie was good | positive | positive | positive |
| this movie was bad | negative | negative | negative |
| the movie was not bad | negative | negative | positive |

- Will return to compositionality with syntax and LSTMs

Iyyer et al. (2015)

# Feedforward Neural Language Models

- Given previous n words, predict the next word $P(\text{mat}|\text{the cat sat on the})$




- Input layer (context size n = 5):

$$\mathbf{x} = [\mathbf{e}_{\text{the}}; \mathbf{e}_{\text{cat}}; \mathbf{e}_{\text{sat}}; \mathbf{e}_{\text{on}}; \mathbf{e}_{\text{the}}] \in \mathbb{R}^{dn}$$

concatenation

- Hidden layer

$$\mathbf{h} = \tanh(\mathbf{W}\mathbf{x} + \mathbf{b}) \in \mathbb{R}^{h}$$

- Output layer (softmax)

$$\mathbf{z} = \mathbf{U}\mathbf{h} \in \mathbb{R}^{|V|}$$

$$P(w = i \mid \text{context}) = softmax_i(\mathbf{z})$$

Bengio et al. (2003)

# Logistics

- One extra slip day for everyone.
- HW1 due today, HW2 will be released today

# Assignment 2

- I anticipate it will take a bit longer than assignment 1
- Two programming components
  - Implementing deep averaging network for sentiment classification (with PyTorch).
  - Implementing inference of generative sequence model for part-of-speech tagging task (which we will learn today / Thursday)

# Sequence Models

- Topics for next three lectures and HW2

- We will be back to neural sequence models again in a few weeks

# Overview

- Sequence Modeling Problems in NLP
- Generative Model: Hidden Markov Models (HMM)
- Discriminative Model:
  Maximum Entropy Markov Models (MEMM)
- Unsupervised Learning: Expectation Maximization

# Reading

- Collins: HMMs —> Generative sequence tagging model
- Collins: MEMMs —> Discriminative sequence tagging models
- Collins: EMs —> Expectation Maximization
- J&M: Chapter 8 (optional)
  - Covers both HMM, MEMM

# The Structure of Language

- Language is tree-structured



- But labelled ***sequence*** can provide shallow analysis

| PRP | VBZ | DT | NN | IN | NNS |
|---|---|---|---|---|---|
| I | ate | the | spaghetti | with | chopsticks |

| PRP | VBZ | DT | NN | IN | NNS |
|---|---|---|---|---|---|
| I | ate | the | spaghetti | with | meatballs |

# Sequence Modeling Problems in NLP

- Parts of Speech Tagging (POS)

| PRP | VBZ | DT | NN | IN | NNS |
|---|---|---|---|---|---|
| I | ate | the | spaghetti | with | chopsticks |

| PRP | VBZ | DT | NN | IN | NNS |
|---|---|---|---|---|---|
| I | ate | the | spaghetti | with | meatballs |

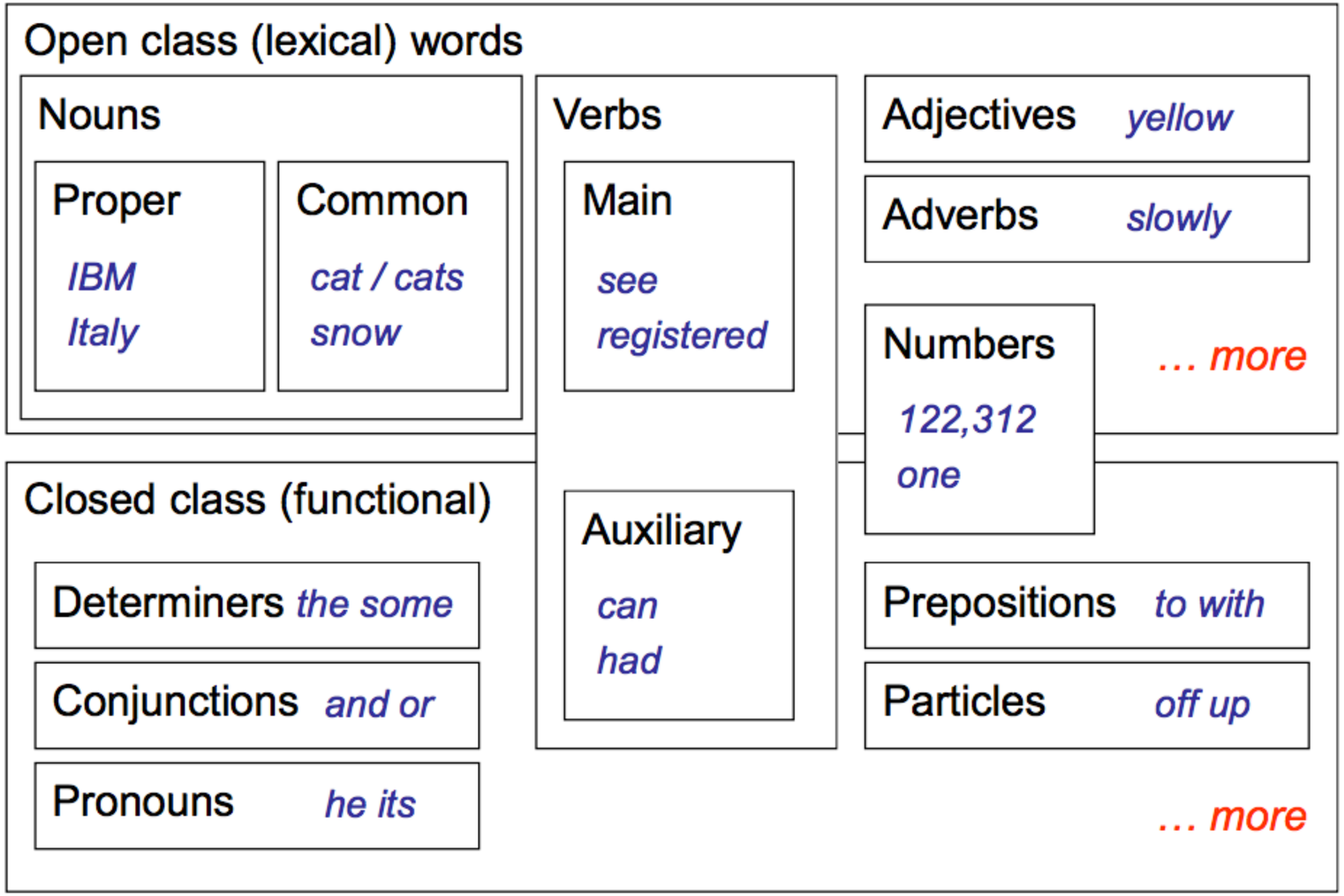
- Named Entity Recognition (NER):
  - Segment text into spans with certain properties (person, organization, )

[Germany]$_{LOC}$ 's representative to the [European Union]$_{ORG}$ 's veterinary committee [Werner Zwingman]$_{PER}$ said on Wednesday consumers should…

Germany/BL 's/NA representative/NA to/NA the/NA European/BO Union/CO 's/NA veterinary/NA committee/NA Werner/BP Zwingman/CP said/NA on/NA Wednesday/NA consumers/NA should/NA…

# Parts of Speech

- Categorization of words into types




Slide credit: Dan Klein

Main Tags

| | | |
|---|---|---|
| CC | conjunction, coordinating | and both but either or |
| CD | numeral, cardinal | mid-1890 nine-thirty 0.5 one |
| DT | determiner | a all an every no that the |
| EX | existential there | there |
| FW | foreign word | gemeinschaft hund ich jeux |
| IN | preposition or conjunction, subordinating | among whether out on by if |
| JJ | adjective or numeral, ordinal | third ill-mannered regrettable |
| JJR | adjective, comparative | braver cheaper taller |
| JJS | adjective, superlative | bravest cheapest tallest |
| MD | modal auxiliary | can may might will would |
| NN | noun, common, singular or mass | cabbage thermostat investment subhumanity |
| NNP | noun, proper, singular | Motown Cougar Yvette Liverpool |
| NNPS | noun, proper, plural | Americans Materials States |
| NNS | noun, common, plural | undergraduates bric-a-brac averages |
| POS | genitive marker | ' 's |
| PRP | pronoun, personal | hers himself it we them |
| PRP$ | pronoun, possessive | her his mine my our ours their thy your |
| RB | adverb | occasionally maddeningly adventurously |
| RBR | adverb, comparative | further gloomier heavier less-perfectly |
| RBS | adverb, superlative | best biggest nearest worst |
| RP | particle | aboard away back by on open through |
| TO | "to" as preposition or infinitive marker | to |
| UH | interjection | huh howdy uh whammo shucks heck |
| VB | verb, base form | ask bring fire see take |
| VBD | verb, past tense | pleaded swiped registered saw |
| VBG | verb, present participle or gerund | stirring focusing approaching erasing |
| VBN | verb, past participle | dilapidated imitated reunifed unsettled |
| VBP | verb, present tense, not 3rd person singular | twist appear comprise mold postpone |
| VBZ | verb, present tense, 3rd person singular | bases reconstructs marks uses |
| WDT | WH-determiner | that what whatever which whichever |
| WP | WH-pronoun | that what whatever which who whom |
| WP$ | WH-pronoun, possessive | whose |

# POS Tagging

- Many words have more than one POS, depending on its context

  The *back* door = JJ (Adjective)

  On my *back* = NN (Noun)

  Win the voters *back* = RB (Adverb)

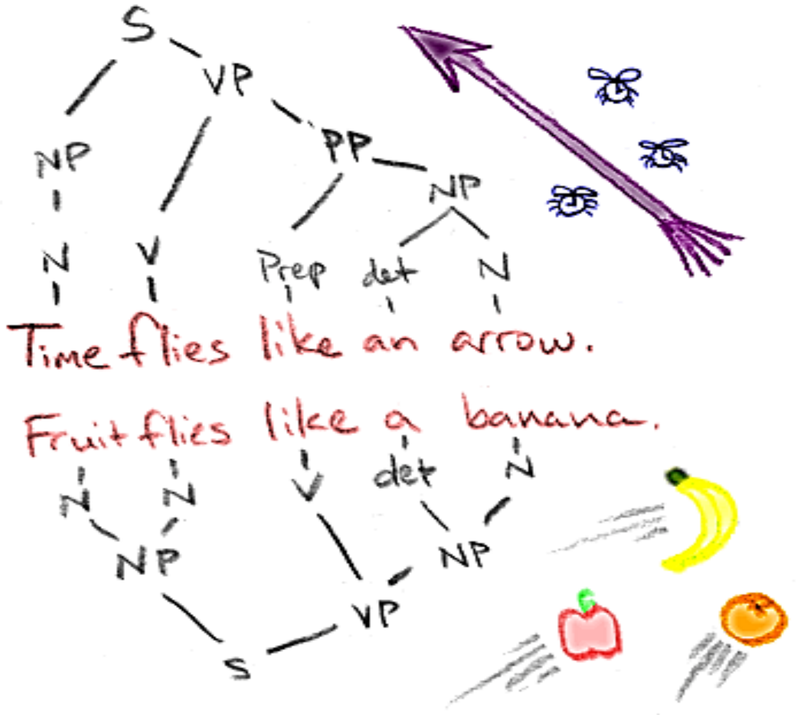
  Promised to *back* the bill = VB (Verb)

- The POS tagging problem is to determine the POS tag for a particular instance of a word.

# Sources of Information

- Knowledge of neighboring words

Time flies like an arrow;
Fruit flies like a banana




- Knowledge of word probabilities
  - ***the, a, an*** is almost always article
  - ***man*** is frequently noun, rarely used as a verb
  - If we choose the most frequent tag, over 90% accuracy
  - About 40% of word tokens are ambiguous

# What is this good for?

- Preprocessing step for syntactic parsers
- Domain-independent disambiguation for other tasks
- (Very) shallow information extraction:
  - write regular expressions like (Det) Adj*N + over the output for phrases

# POS tag sets in different languages

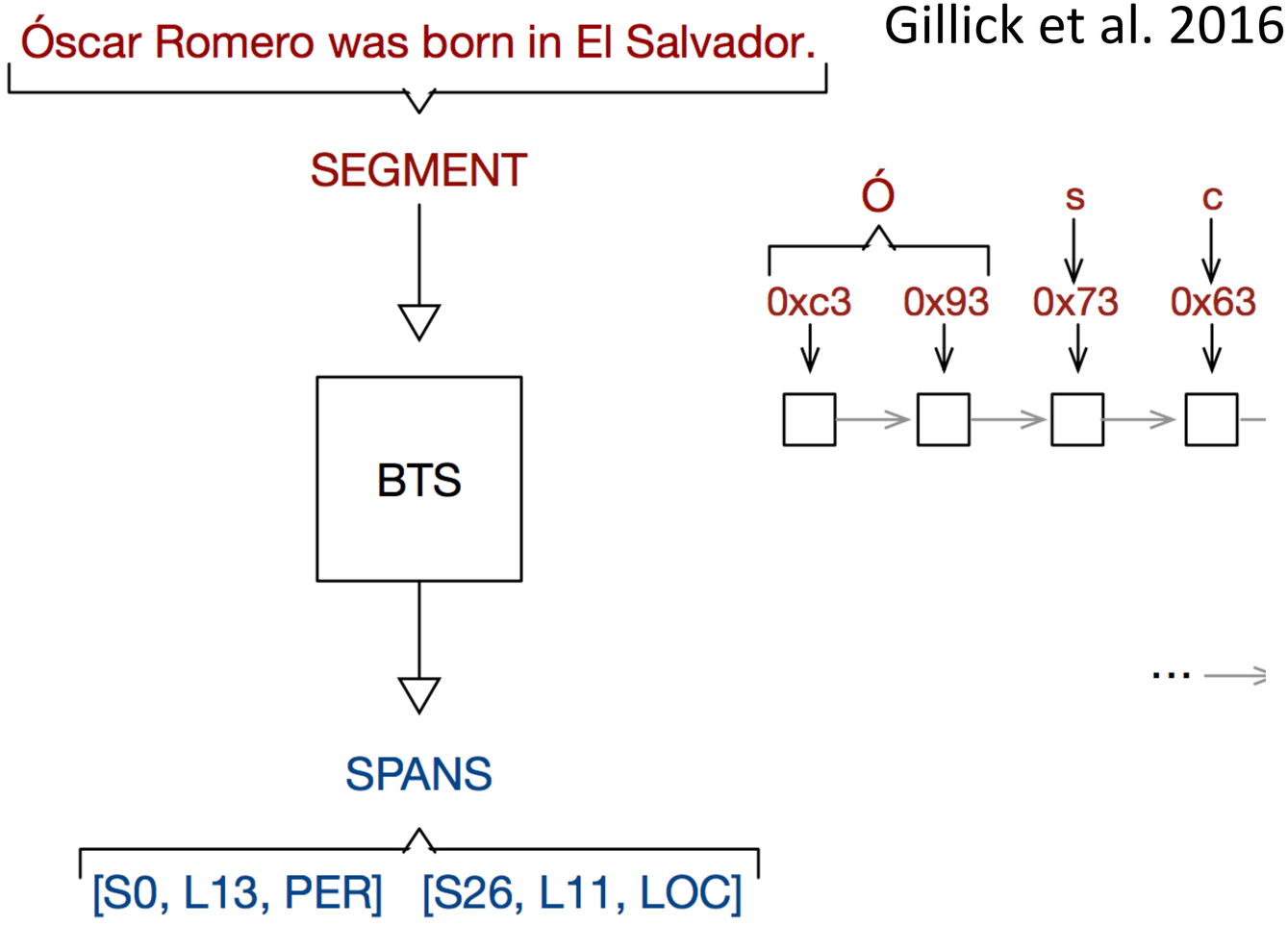
| Language | Source | # Tags |
|---|---|---|
| Arabic | PADT/CoNLL07 (Hajič et al., 2004) | 21 |
| Basque | Basque3LB/CoNLL07 (Aduriz et al., 2003) | 64 |
| Bulgarian | BTB/CoNLL06 (Simov et al., 2002) | 54 |
| Catalan | CESS-ECE/CoNLL07 (Martí et al., 2007) | 54 |
| Chinese | Penn ChineseTreebank 6.0 (Palmer et al., 2007) | 34 |
| Chinese | Sinica/CoNLL07 (Chen et al., 2003) | 294 |
| Czech | PDT/CoNLL07 (Böhmová et al., 2003) | 63 |
| Danish | DDT/CoNLL06 (Kromann et al., 2003) | 25 |
| Dutch | Alpino/CoNLL06 (Van der Beek et al., 2002) | 12 |
| English | PennTreebank (Marcus et al., 1993) | 45 |
| French | FrenchTreebank (Abeillé et al., 2003) | 30 |
| German | Tiger/CoNLL06 (Brants et al., 2002) | 54 |
| German | Negra (Skut et al., 1997) | 54 |
| Greek | GDT/CoNLL07 (Prokopidis et al., 2005) | 38 |
| Hungarian | Szeged/CoNLL07 (Csendes et al., 2005) | 43 |
| Italian | ISST/CoNLL07 (Montemagni et al., 2003) | 28 |
| Japanese | Verbmobil/CoNLL06 (Kawata and Bartels, 2000) | 80 |
| Japanese | Kyoto4.0 (Kurohashi and Nagao, 1997) | 42 |
| Korean | Sejong (http://www.sejong.or.kr) | 187 |
| Portuguese | Floresta Sintá(c)tica/CoNLL06 (Afonso et al., 2002) | 22 |
| Russian | SynTagRus-RNC (Boguslavsky et al., 2002) | 11 |
| Slovene | SDT/CoNLL06 (Džeroski et al., 2006) | 29 |
| Spanish | Ancora-Cast3LB/CoNLL06 (Civit and Martí, 2004) | 47 |
| Swedish | Talbanken05/CoNLL06 (Nivre et al., 2006) | 41 |
| Turkish | METU-Sabanci/CoNLL07 (Oflazer et al., 2003) | 31 |

[Petrov et al. 2012]

# Universal POS Tag Set

| Language | CRF+ | CRF |
|---|---|---|
| Bulgarian | 97.97 | 97.00 |
| Czech | 98.38 | 98.00 |
| Danish | 95.93 | 95.06 |
| German | 93.08 | 91.99 |
| Greek | 97.72 | 97.21 |
| English | 95.11 | 94.51 |
| Spanish | 96.08 | 95.03 |
| Farsi | 96.59 | 96.25 |
| Finnish | 94.34 | 92.82 |
| French | 96.00 | 95.93 |
| Indonesian | 92.84 | 92.71 |
| Italian | 97.70 | 97.61 |
| Swedish | 96.81 | 96.15 |
| AVERAGE | 96.04 | 95.41 |

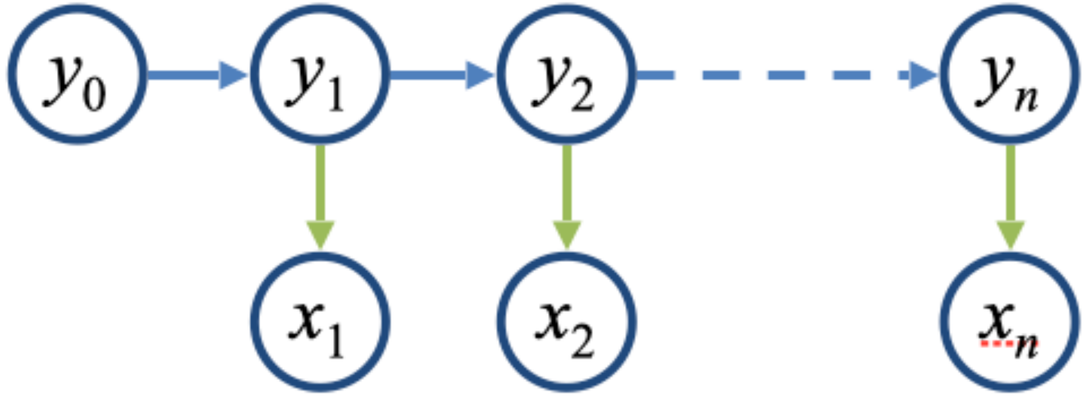
Gillick et al. 2016




- Universal POS tagset (~12 tags), cross-lingual model works well!

# Classic Solution: Hidden Markov Models

- Input $\mathbf{x} = (x_1, ..., x_n)$ Output $\mathbf{y} = (y_1, ..., y_n)$




$$p(x_1 \ldots x_n, y_1 \ldots y_n) =$$

# Two simplifying assumptions

- Markov Assumption (future is conditionally independent of the past given the present)

$$P(y_i \mid y_1, y_2, \cdots, y_{i-1}) = P(y_i \mid y_{i-1})$$

- Independent Assumption:

$$P(x_i \mid \mathbf{x}, \mathbf{y}) = P(x_i \mid y_i)$$

# HMM for POS

The Georgia branch had taken on loan commitments …

DT NNP NN VBD VBN RP NN NNS

- States $Y$ = {DT, NNP, NN, ... } are the POS tags
- Observations $X$ = V are words
- Transition distribution $q(y_i \mid y_{i-1})$ models the tag sequences
- Emission distribution $e(x_i \mid y_i)$ models words given their POS

# HMM Learning and Inference

- Learning:
  - Maximum likelihood: transition q and emission e

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

- Inference:
  - Viterbi: $y^* = \arg \max_{y_1 \dots y_n} p(x_1 \dots x_n, y_1 \dots y_n)$

# Learning: Maximum Likelihood

- Supervised Learning for estimating transitions and emissions

$$q_{ML}(y_i|y_{i-1}) =$$

$$e_{ML}(x|y) =$$

- Any concerns for the quality of any of these estimates?

Sparsity again!

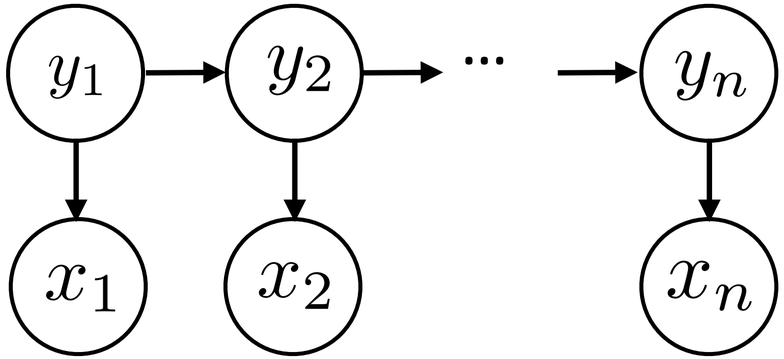
# Learning: Low frequency Words

- Used the following word classes for infrequent words [Bickel et. al, 1999]

| Word class | Example | Intuition |
|---|---|---|
| twoDigitNum | 90 | Two digit year |
| fourDigitNum | 1990 | Four digit year |
| containsDigitAndAlpha | A8956-67 | Product code |
| containsDigitAndDash | 09-96 | Date |
| containsDigitAndSlash | 11/9/89 | Date |
| containsDigitAndComma | 23,000.00 | Monetary amount |
| containsDigitAndPeriod | 1.00 | Monetary amount,percentage |
| othernum | 456789 | Other number |
| allCaps | BBN | Organization |
| capPeriod | M. | Person name initial |
| firstWord | first word of sentence | no useful capitalization information |
| initCap | Sally | Capitalized word |
| lowercase | can | Uncapitalized word |
| other | , | Punctuation marks, all other words |

# Inference (Decoding)

- Input $\mathbf{x} = (x_1, ..., x_n)$ Output $\mathbf{y} = (y_1, ..., y_n)$



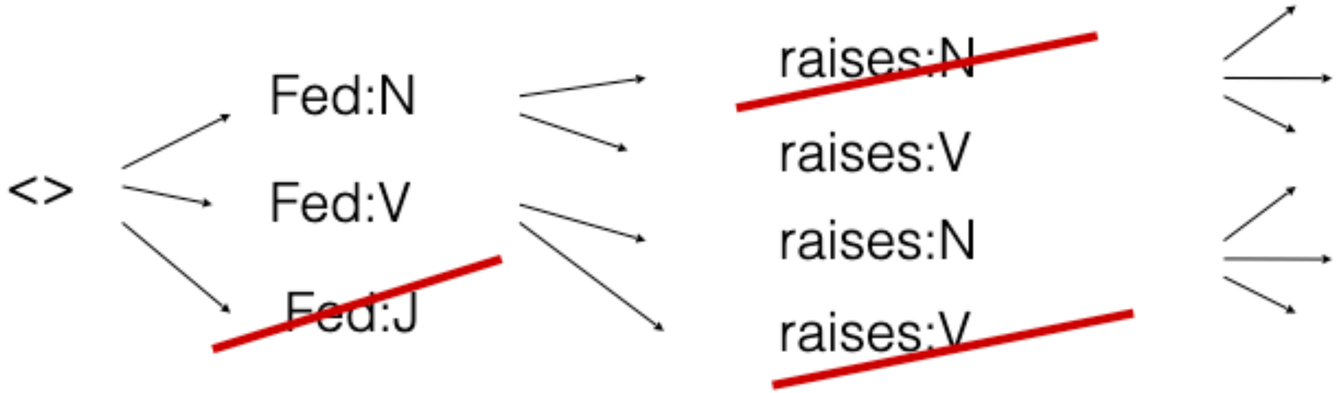

$$p(x_1 \dots x_n, y_1 \dots y_n) = q(STOP|y_n) \prod_{i=1}^{n} q(y_i|y_{i-1}) e(x_i|y_i)$$

- Inference problem: $\text{argmax}_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}) = \text{argmax}_{\mathbf{y}} \frac{P(\mathbf{y}, \mathbf{x})}{\cancel{P(\mathbf{x})}}$

- We can list all possible y and then pick the best one!
  - Any problems?

# Inference (Decoding)

- First solution: Beam Search
  - A beam is a set of partial hypotheses
  - Start with a single empty trajectory
  - At each step, consider all continuation, discard most, keep top K




- But this does not guarantee the optimal answer…

# The Viterbi Algorithm

- Dynamic program for computing the max score of a sequence of length i ending in tag $y_i$

$$\pi(i, y_i) = \max_{y_1 \dots y_{i-1}} p(x_1 \dots x_i, y_1 \dots y_i)$$

$$= \max_{y_{i-1}} e(x_i|y_i) q(y_i|y_{i-1}) \max_{y_1 \dots y_{i-2}} p(x_1 \dots x_{i-1}, y_1 \dots y_{i-1})$$

$$= \max_{y_{i-1}} e(x_i|y_i) q(y_i|y_{i-1}) \quad \pi(i-1, y_{i-1})$$

- Now this is an efficient algorithm!

# The Viterbi Algorithm

- Dynamic program for computing (for all i)

$$\pi(i, y_i) = \max_{y_1 \dots y_{i-1}} p(x_1 \dots x_i, y_1 \dots y_i)$$

- Iterative Computation:

$$\pi(0, y_0) = \begin{cases} 1 \text{ if } y_0 == START \\ 0 \text{ otherwise} \end{cases}$$

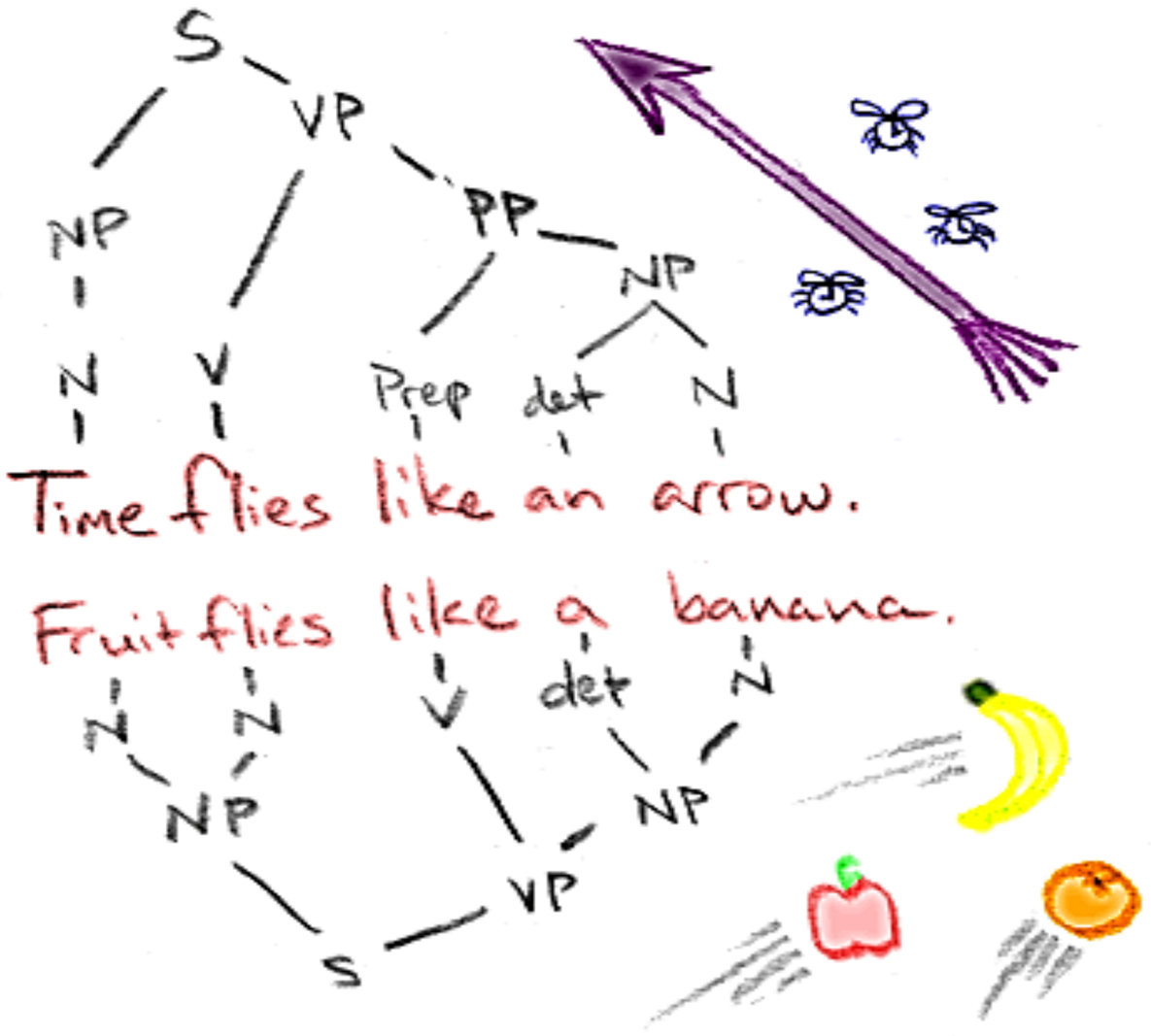
  - For I = 1… n:
    - Store score

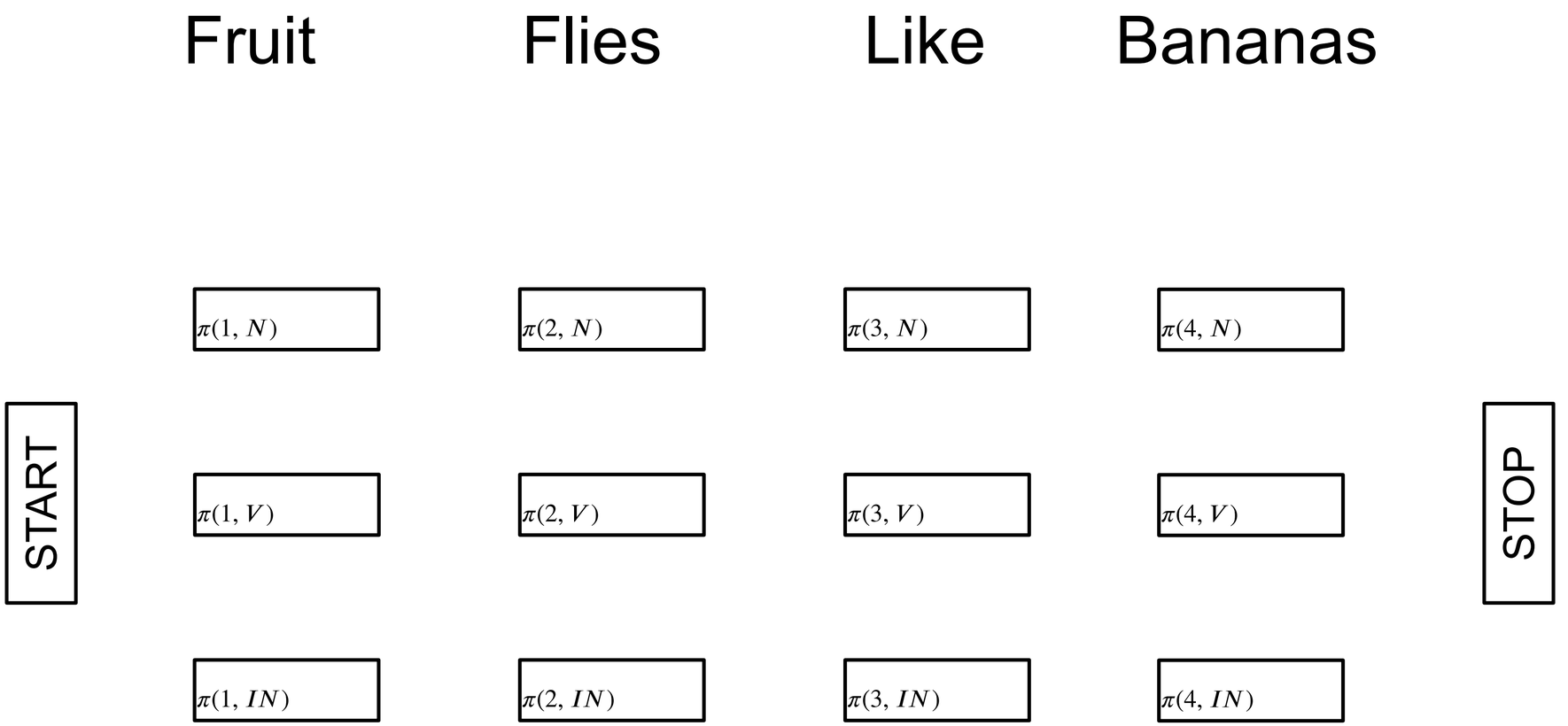
$$\pi(i, y_i) = \max_{y_{i-1}} e(x_i|y_i) q(y_i|y_{i-1}) \pi(i-1, y_{i-1})$$

    - Store back-pointer

$$bp(i, y_i) = \arg\max_{y_{i-1}} e(x_i|y_i) q(y_i|y_{i-1}) \pi(i-1, y_{i-1})$$

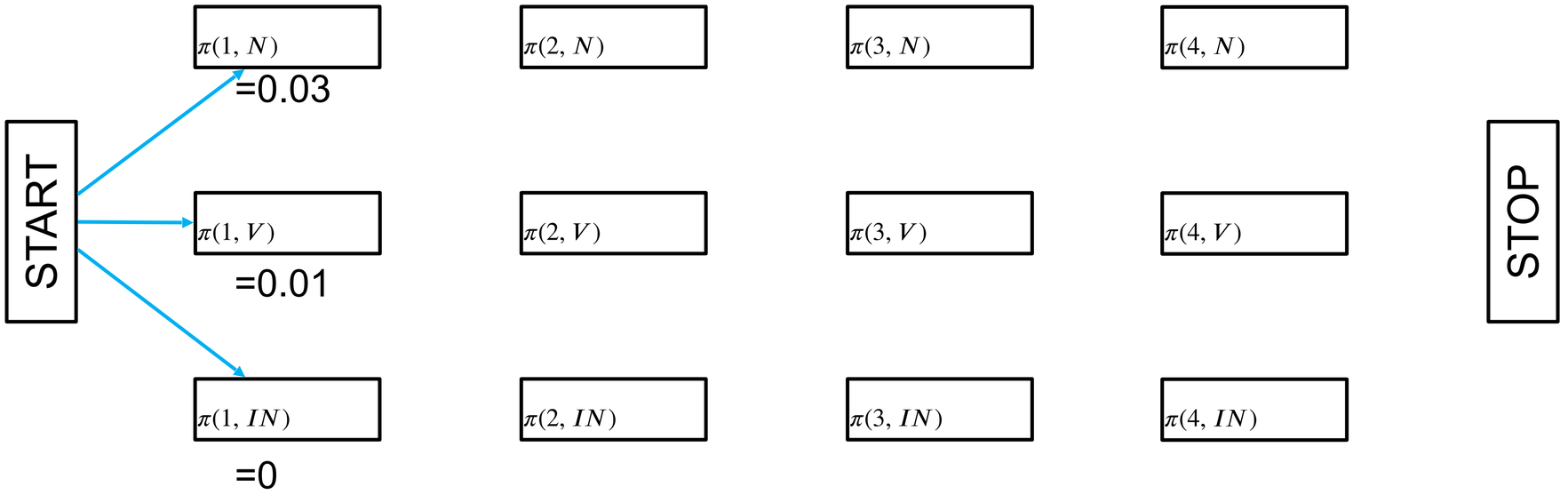
# Time flies like an arrow;
# Fruit flies like a banana





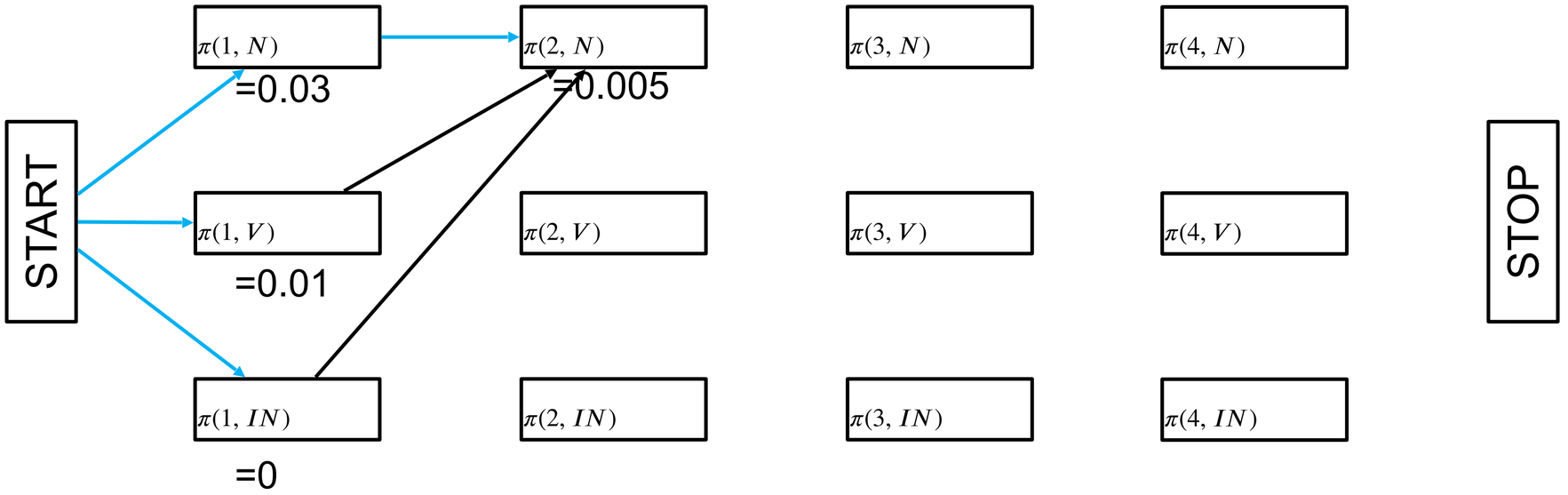

$$\pi(i, y_i) = \max_{y_1 \dots y_{i-1}} p(x_1 \dots x_i, y_1 \dots y_i)$$

Fruit Flies Like Bananas



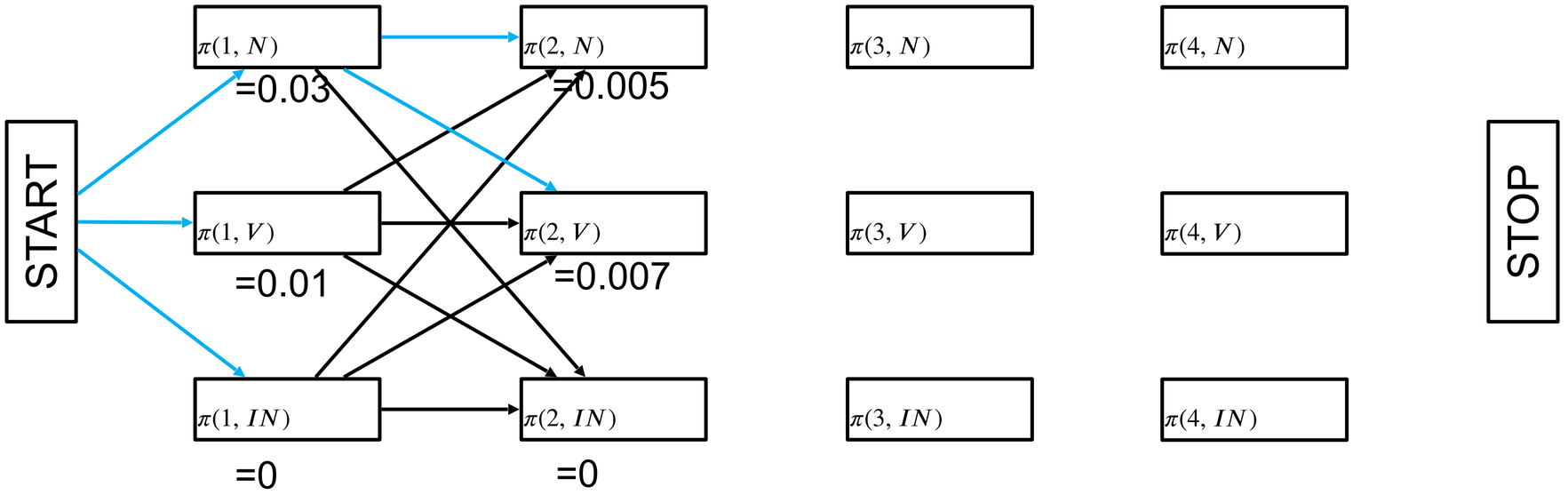

$$\pi(i, y_i) = \max_{y_1 \dots y_{i-1}} p(x_1 \dots x_i, y_1 \dots y_i)$$

Fruit Flies Like Bananas



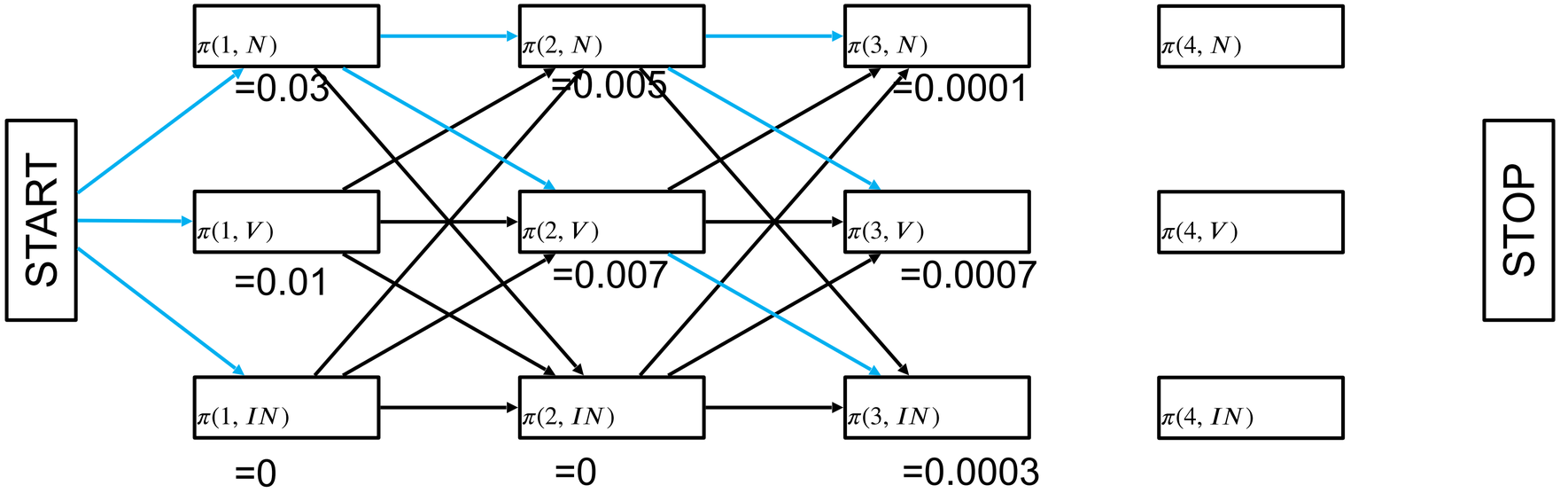

$$\pi(i, y_i) = \max_{y_1 \dots y_{i-1}} p(x_1 \dots x_i, y_1 \dots y_i)$$

Fruit Flies Like Bananas

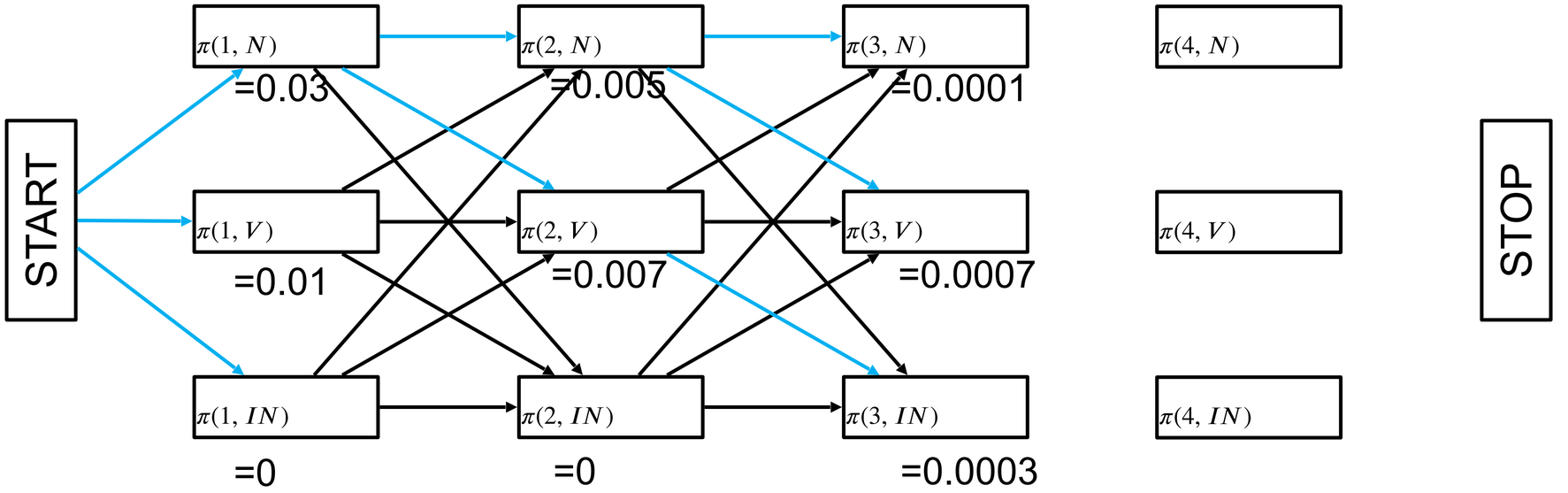



$$\pi(i, y_i) = \max_{y_1 \dots y_{i-1}} p(x_1 \dots x_i, y_1 \dots y_i)$$

Fruit Flies Like Bananas




$$\pi(i, y_i) = \max_{y_1 \dots y_{i-1}} p(x_1 \dots x_i, y_1 \dots y_i)$$

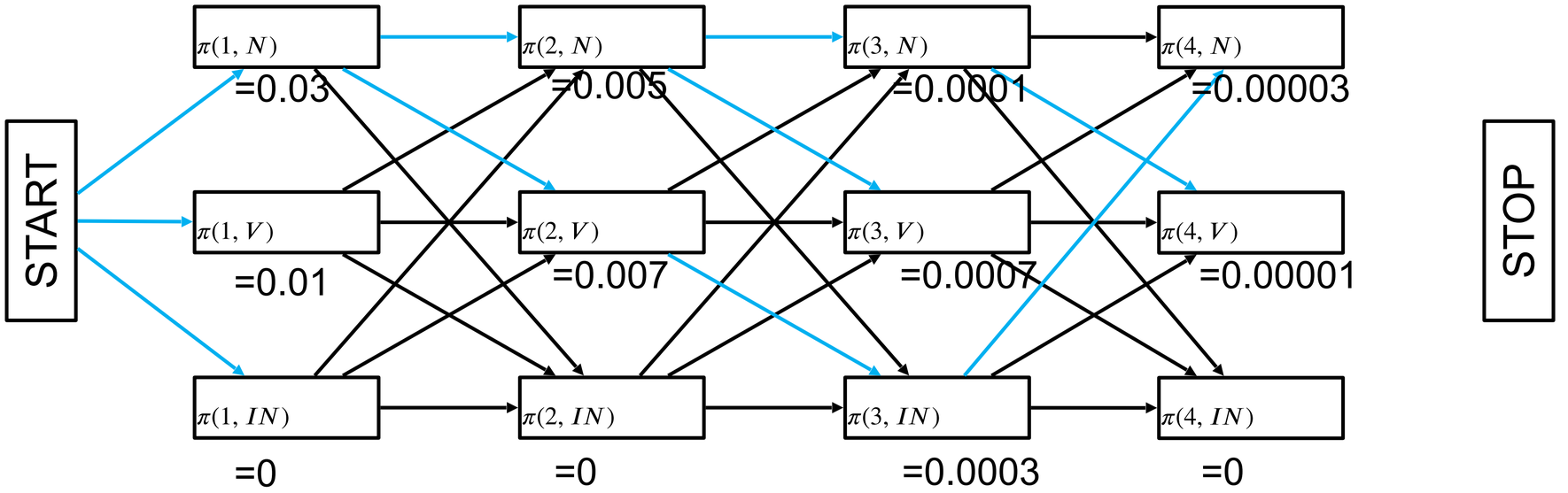



$$\pi(i, y_i) = \max_{y_1 \cdots y_{i-1}} p(x_1 \ldots x_i, y_1 \ldots y_i)$$

$$= \max_{y_1 \cdots y_{i-2}} p(x_1 \ldots x_{i-1}, y_1 \ldots y_{i-1})$$

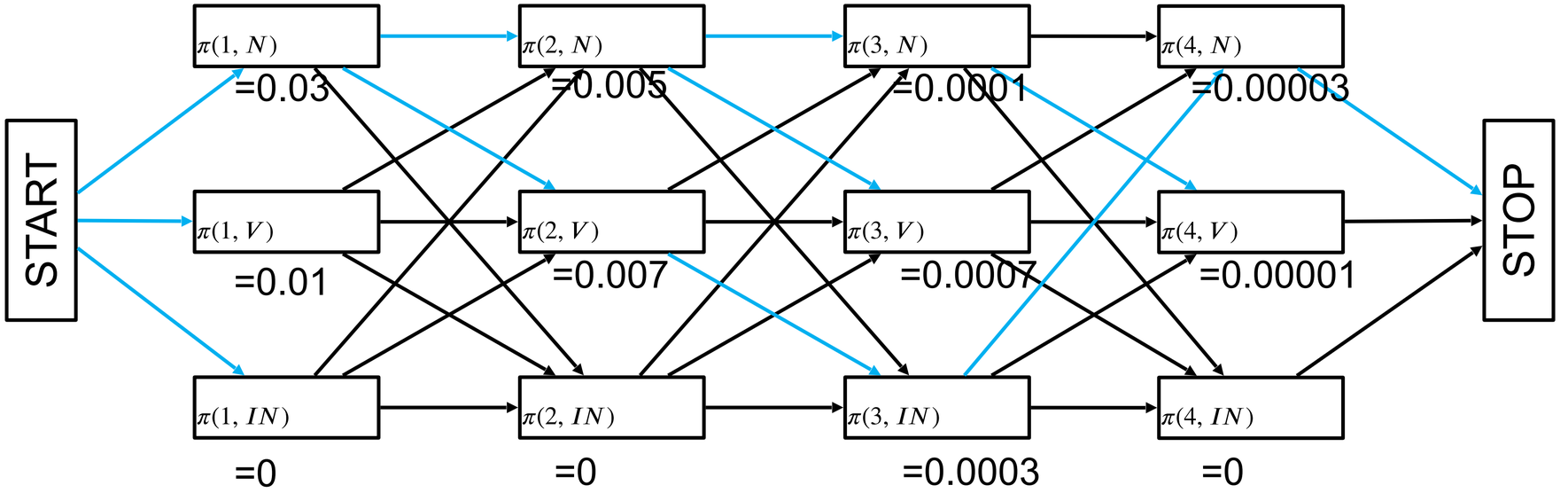
$$= \max_{y_{i-1}} e(x_i|y_i) q(y_i|y_{i-1}) \quad \pi(i-1, y_{i-1})$$

Fruit Flies Like Bananas



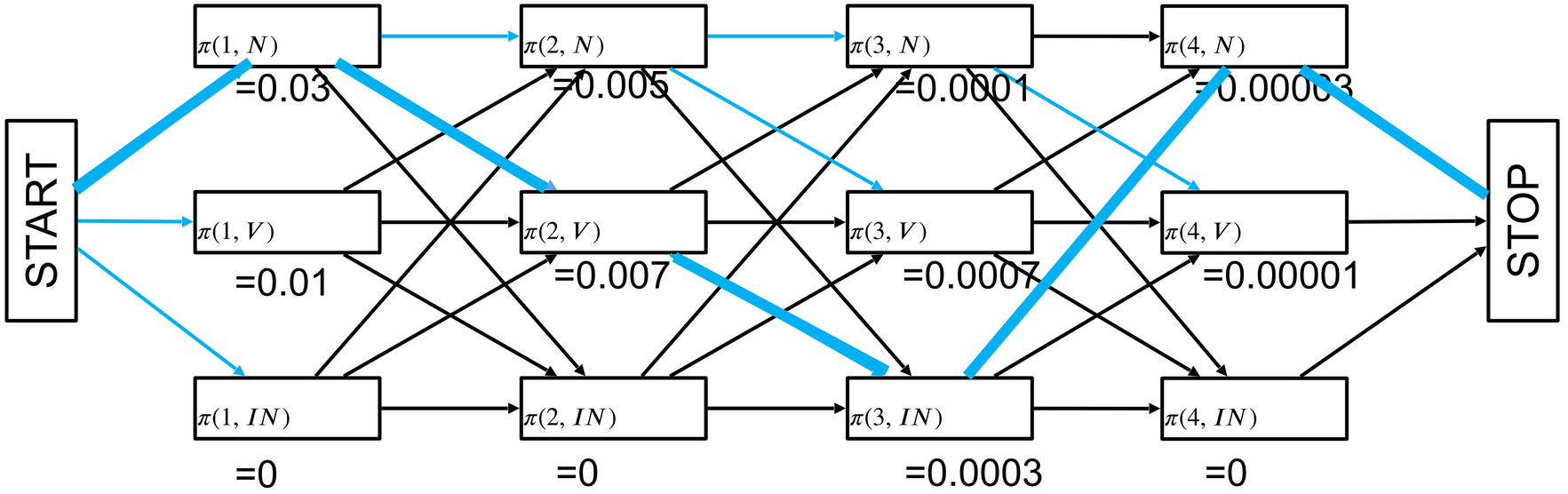

$$\pi(i, y_i) = \max_{y_1 \dots y_{i-1}} p(x_1 \dots x_i, y_1 \dots y_i)$$

Fruit Flies Like Bananas



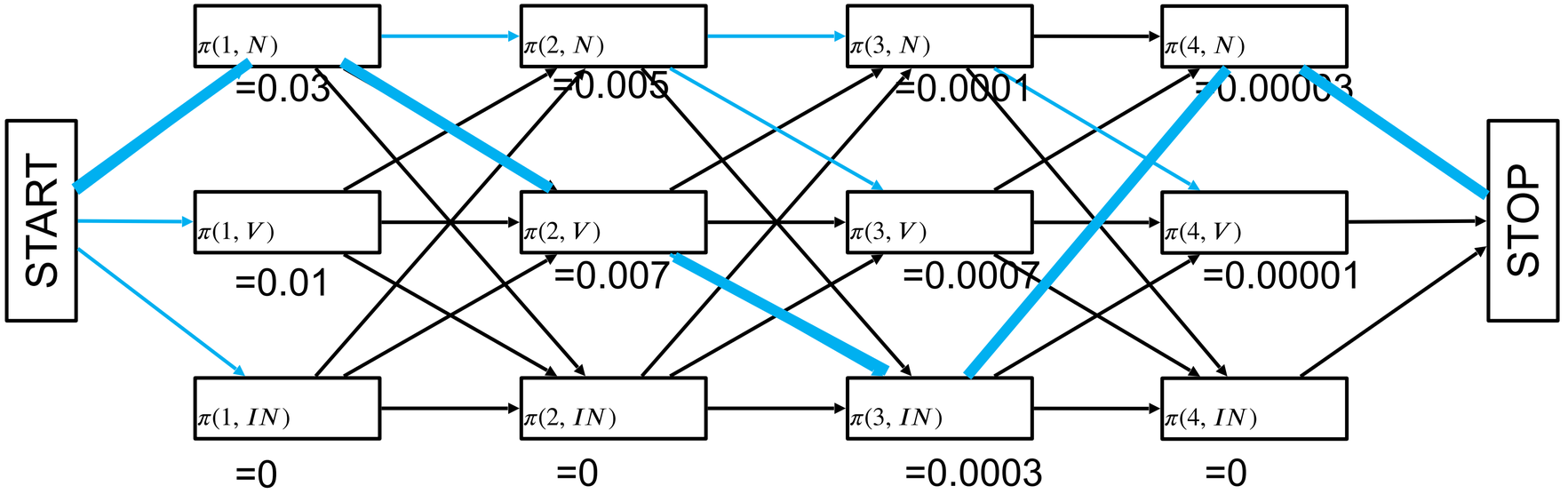

$$\pi(i, y_i) = \max_{y_1 \dots y_{i-1}} p(x_1 \dots x_i, y_1 \dots y_i)$$

Fruit Flies Like Bananas




$$\pi(i, y_i) = \max_{y_1 \dots y_{i-1}} p(x_1 \dots x_i, y_1 \dots y_i)$$

# Why does this find the max p(.)?
# What is the runtime?




$$\pi(i, y_i) = \max_{y_1 \dots y_{i-1}} p(x_1 \dots x_i, y_1 \dots y_i)$$