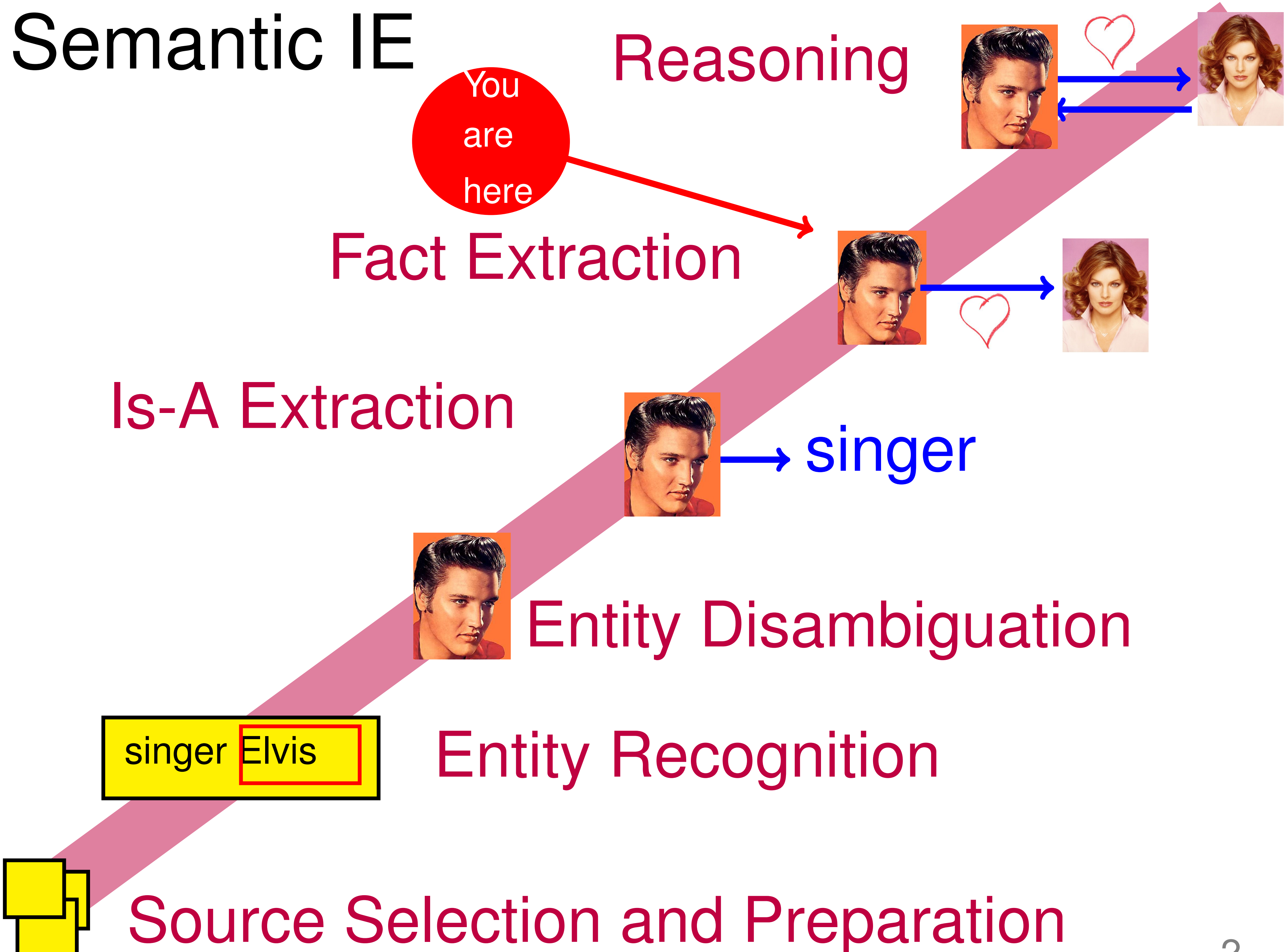
Information Extraction

Lecture 4: IE from unstructured text

Fabian M. Suchanek

Semantic IE


Reasoning
You are here
Fact Extraction
Is-A Extraction
singer
Entity Disambiguation
singer Elvis
Entity Recognition
Source Selection and Preparation

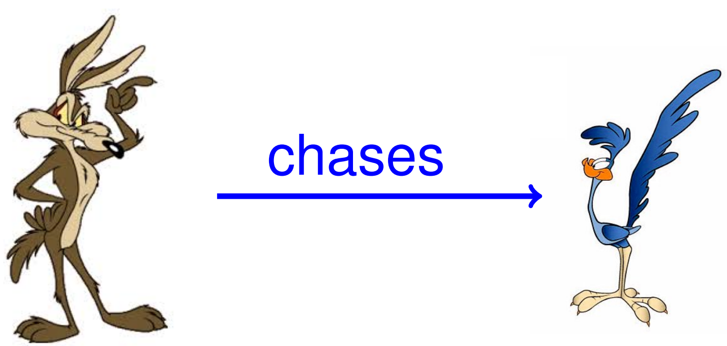
Overview

- Fact extraction from text
  - Difficulties
  - POS Tagging
  - Dependency grammars

Def: Fact Extraction

Fact extraction is the extraction of facts about entities from a corpus.

Coyote chasse Roadrunner.



Def: Extraction Pattern

An extraction pattern for a binary relation r is a string that contains two place-holders X and Y, indicating that two entities stand in relation r.

$X \xrightarrow{\text{chases}} Y$

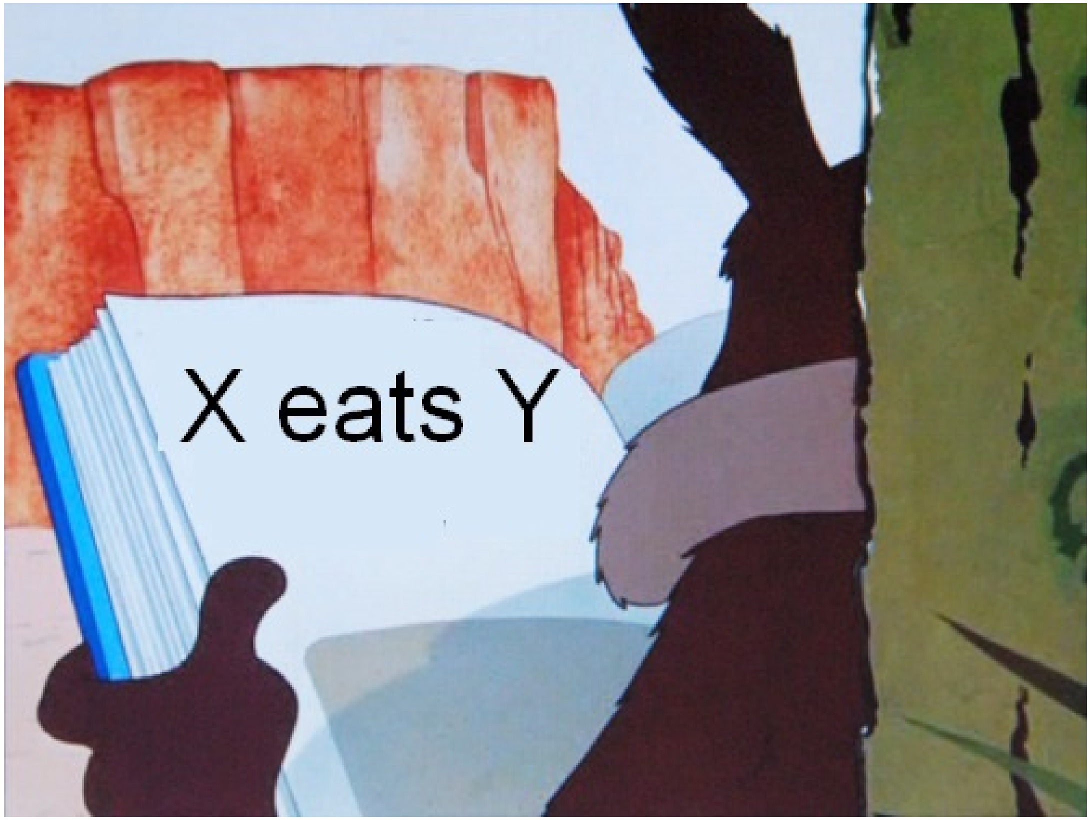
Extraction patterns for

- “X chasse Y”
- “X poursuit Y”
- “la chasse d’X pour Y”
- “Y est chassé par X”

Where do we get the patterns?

- Option 1:

  Manually compile patterns.



Where do we get the patterns?

- Option 2:

  Manually annotate patterns in text

Quand Coyote chasse Roadrunner, il recontre certains inconvenients.

“X chasse Y”

Where do we get the patterns?

- Option 3: Pattern Deduction

  Retrieve patterns by help of known facts.

Known facts:

$X \xrightarrow{\text{chases}} Y$

Obama chasse Osama depuis 2009.

"X chasse Y"

Def: Pattern Deduction

Given a corpus, and given a KB, pattern deduction is the process of finding extraction patterns that produce facts of the KB when applied to the corpus.

KB

chases

+

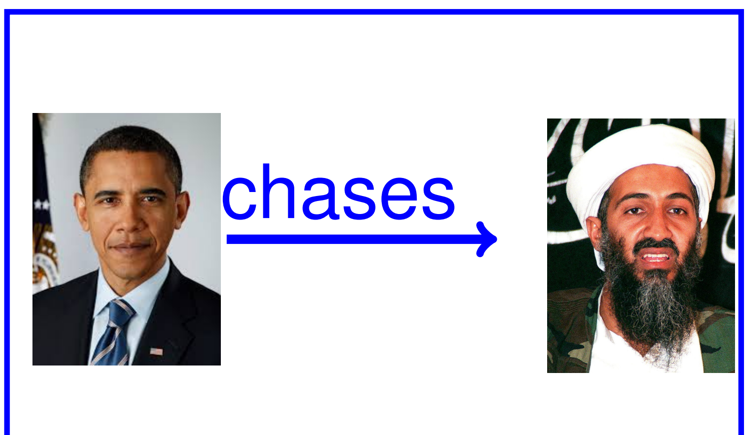
Corpus

Obama chasse Osama.

Def: Pattern Deduction

Given a corpus, and given a KB, pattern deduction is the process of finding extraction patterns that produce facts of the KB when applied to the corpus.

KB



\+

Corpus

Obama chasse Osama.

=> “X chasse Y” is a pattern for chases(X,Y)

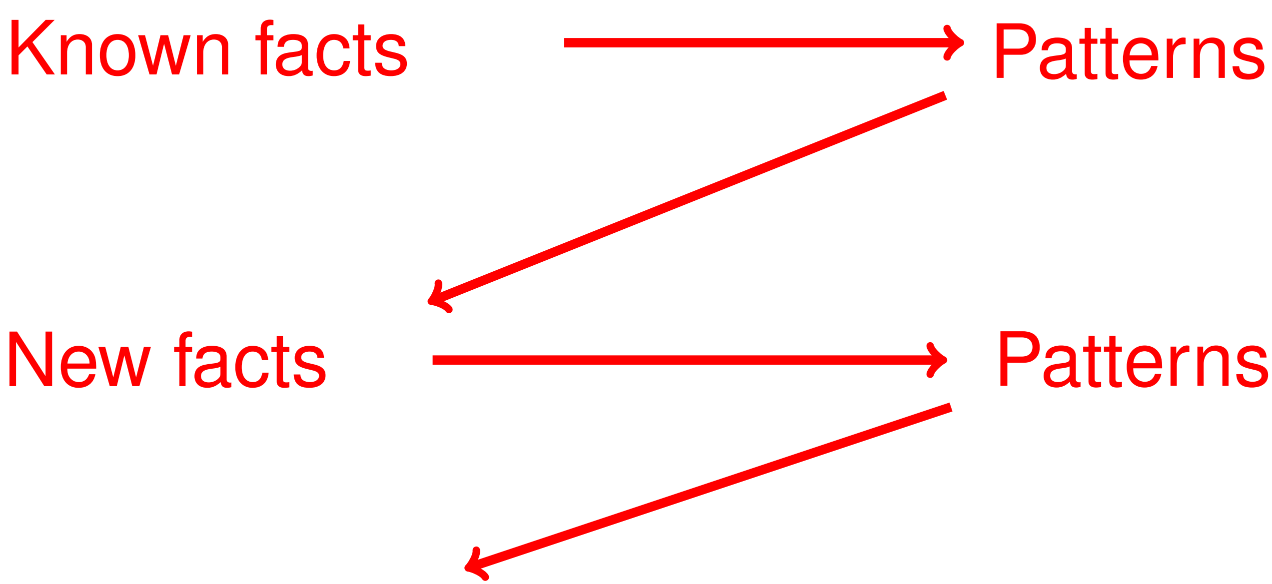
Def: Pattern iteration/DIPRE

important

Pattern iteration (also: DIPRE) is the process of repeatedly

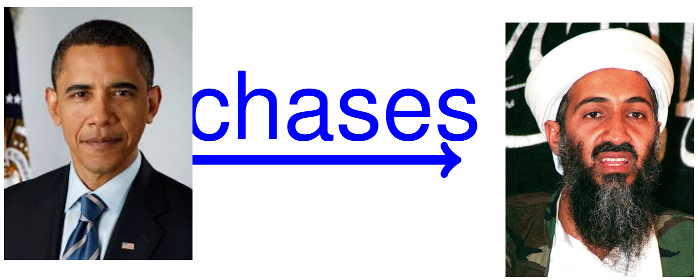
- applying pattern deduction
- using the patterns to find new facts

... thus continuously augmenting the KB.



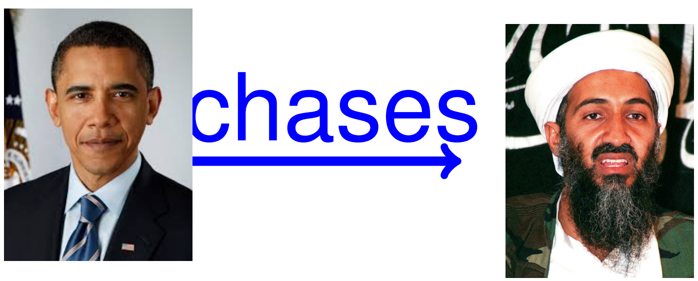
Example: DIPRE

Obama chases Osama. Coyote runs after RR. Coyote chases RR.



Example: DIPRE

Obama chases Osama. Coyote runs after RR. Coyote chases RR.



"X chases Y"

Example: DIPRE

Obama chases Osama. Coyote runs after RR. Coyote chases RR.

"X chases Y"

Example: DIPRE

Obama chases Osama. Coyote runs after RR. Coyote chases RR.

chases

chases

...

"X chases Y"

"X runs after Y"

Task: DIPRE

Given this KB,

apply DIPRE to this corpus:

Obama likes Michelle.

Coyote wants to eat Roadrunner.

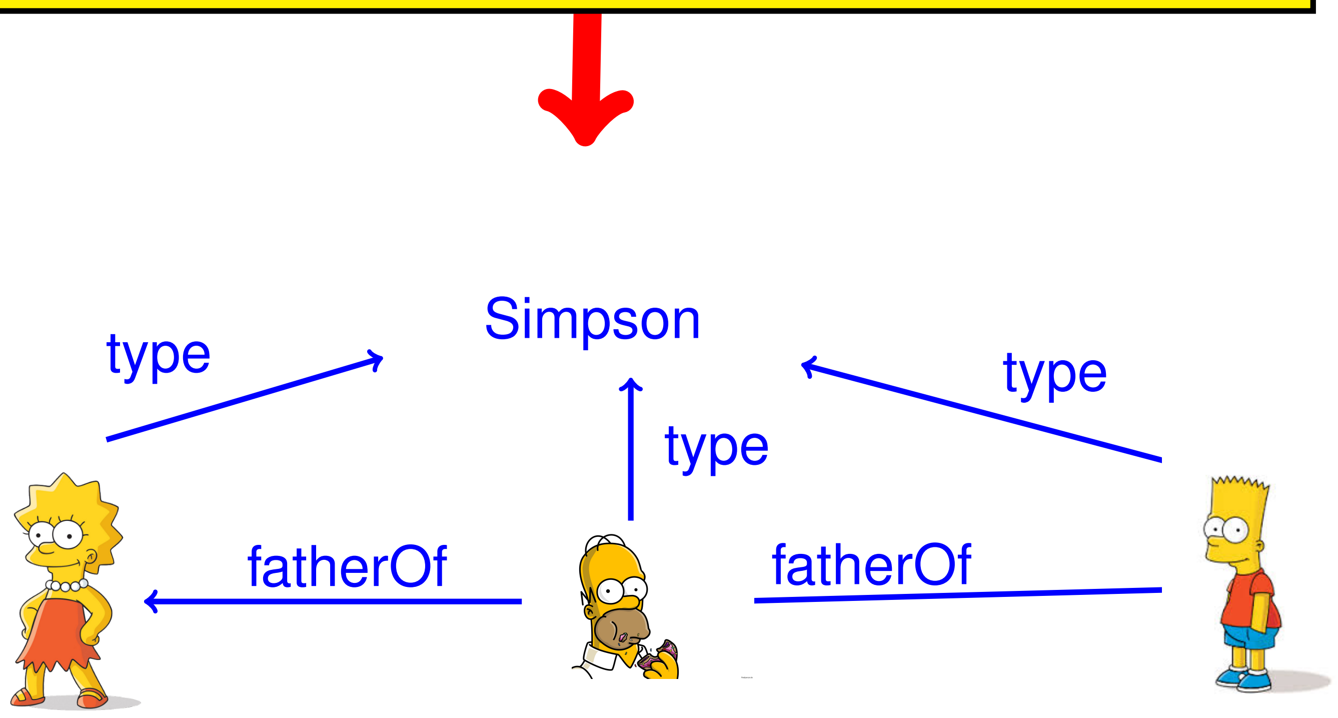
Harry wants to eat cornflakes.

Harry likes cornflakes for breakfast.

Summary: Information Extraction

Congratulations, you can now transform (parts of) natural language text into structured information!

I love Simpsons such as Bart, Lisa, and Homer.
Homer is the father of Bart.
Homer is the father of Lisa.

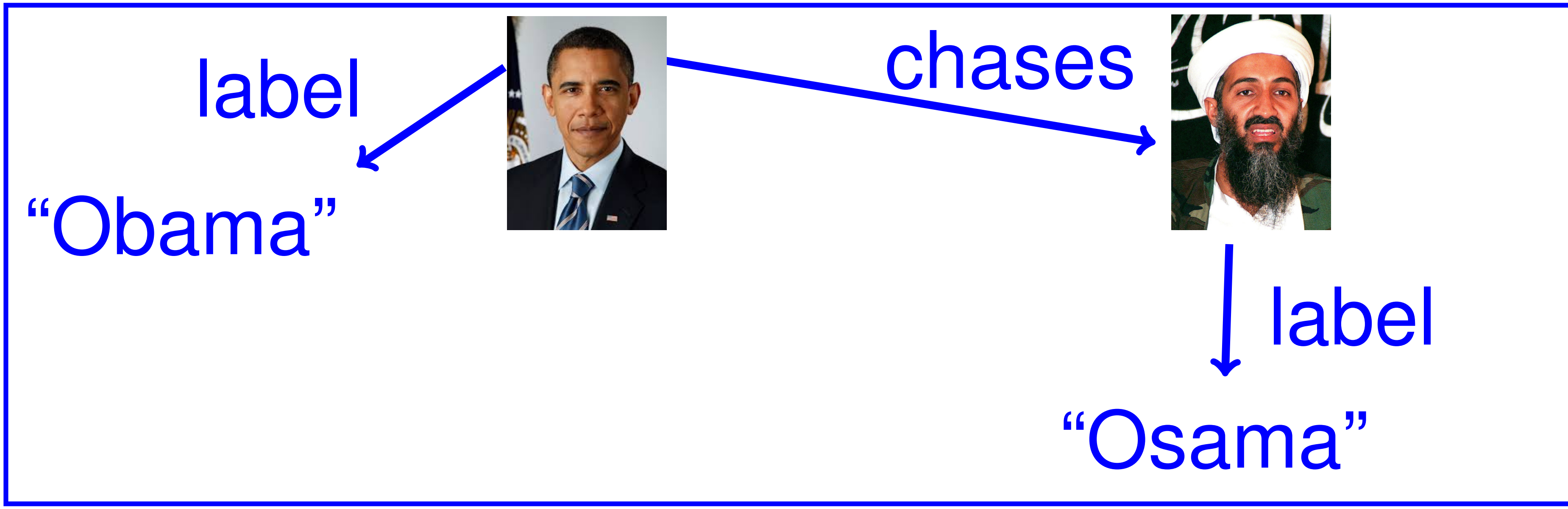


Overview

- Fact extraction from text
  - Difficulties
  - POS Tagging
  - Dependency grammars

We use labels to find patterns

KB



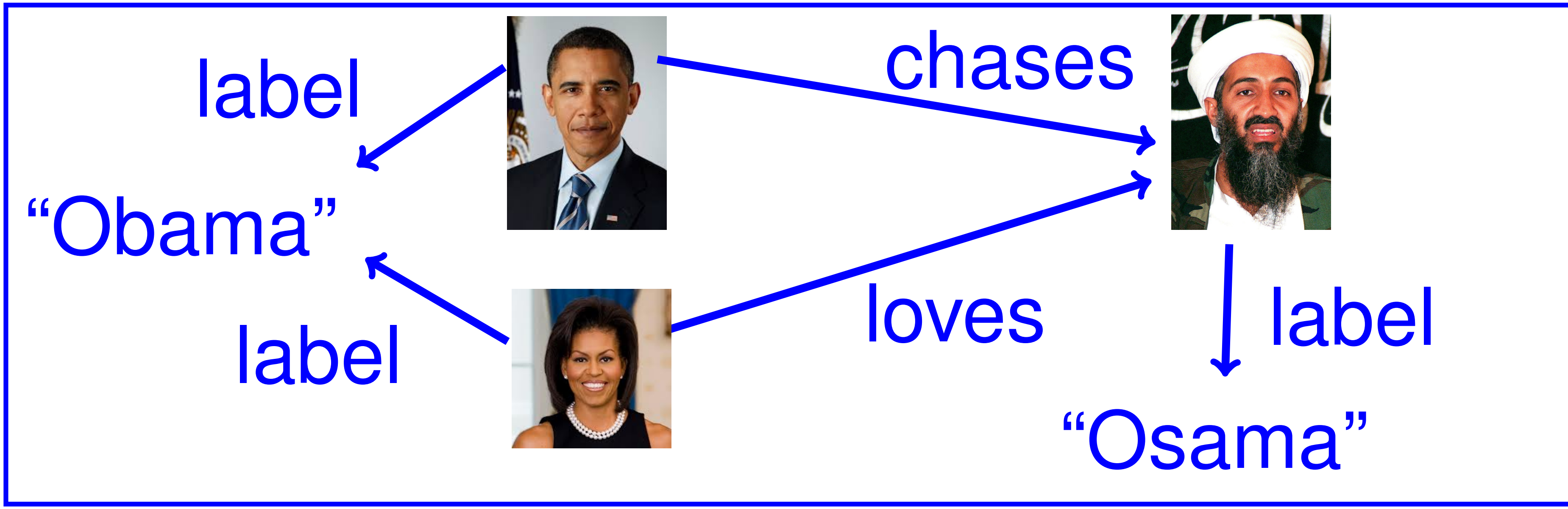
Corpus

+ **Obama chases Osama.**

=> "X chases Y" is a pattern for chases(X,Y)

Ambiguity is a problem

KB

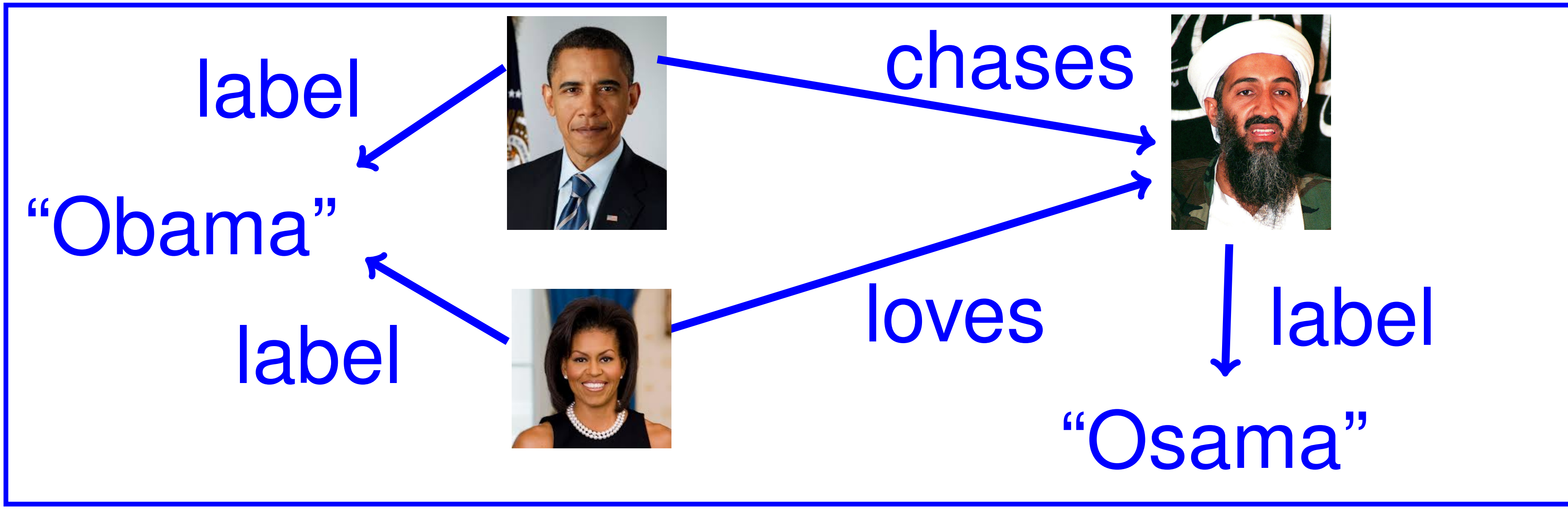


Corpus

\+ **Obama chases Osama.**

Ambiguity is a problem

KB



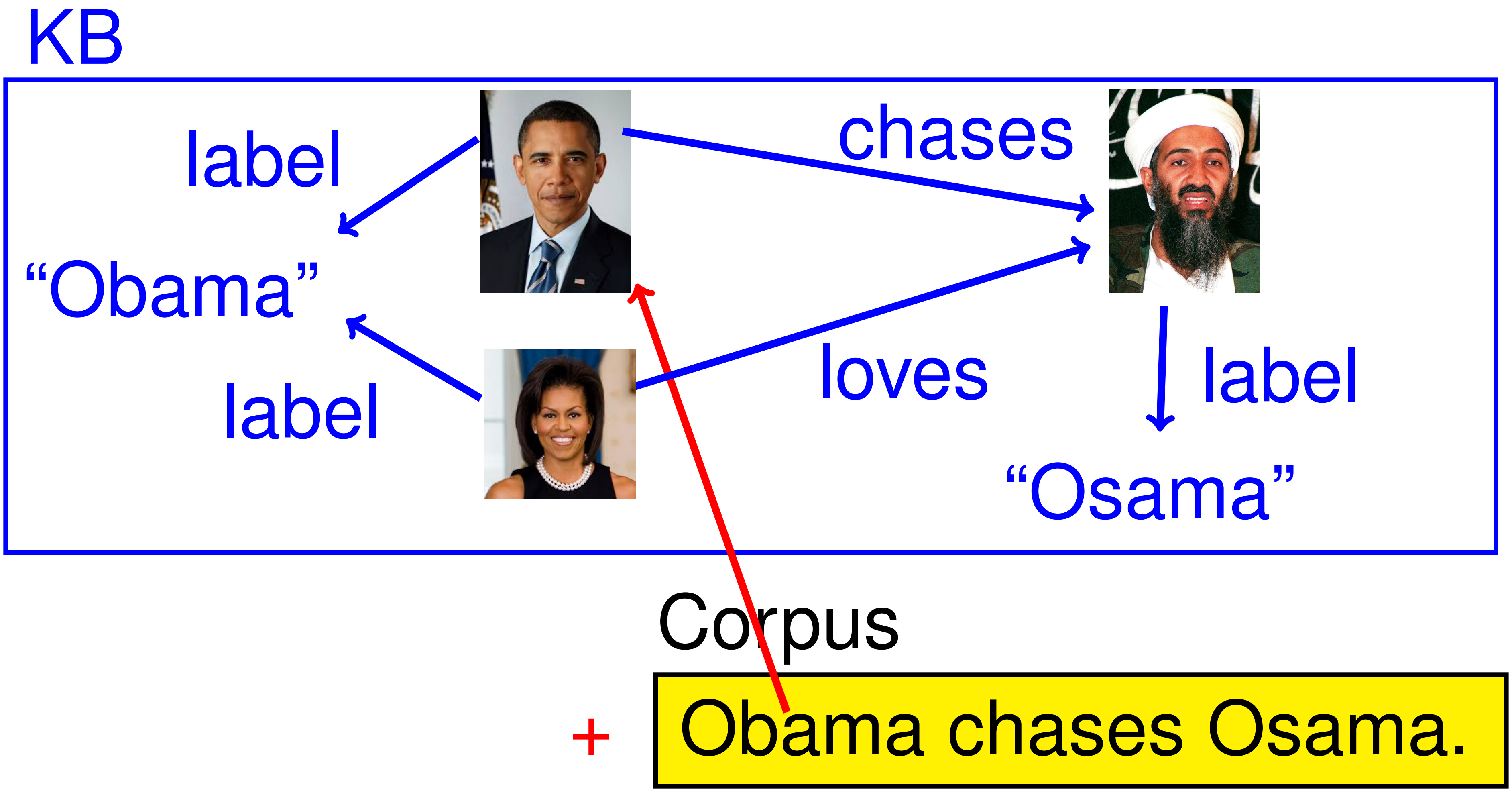
Corpus

\+ Obama chases Osama.

=> “X chases Y” is a pattern

for loves(X,Y) or for chases(X,Y)?

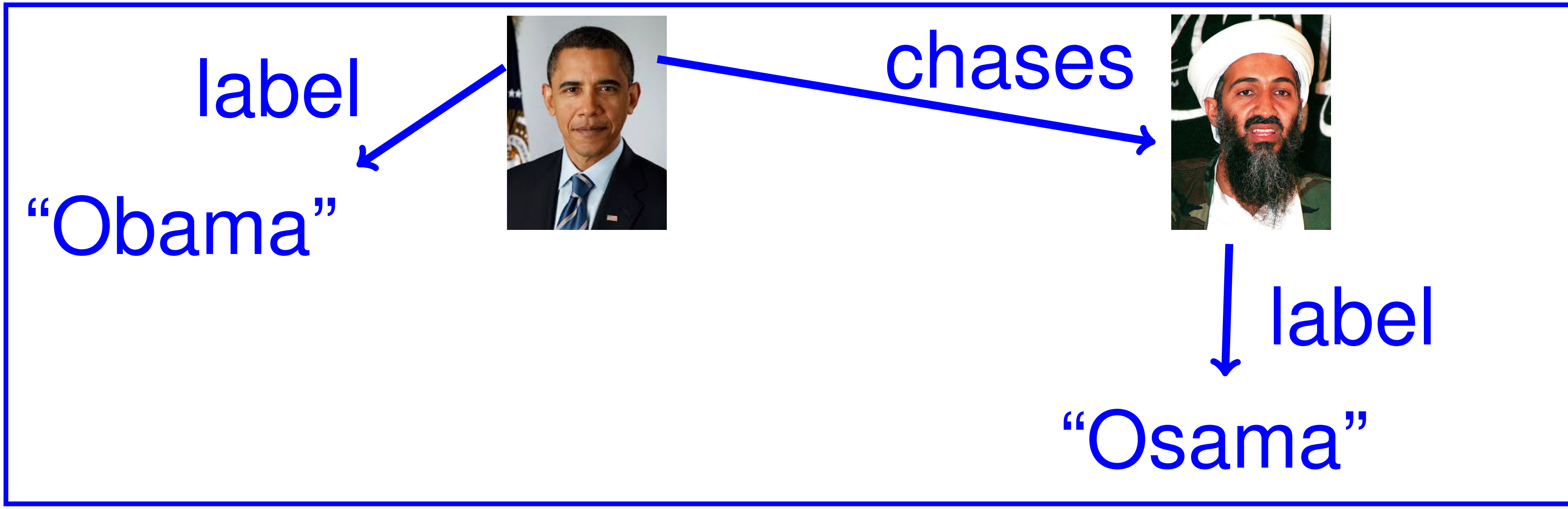
Disambiguation helps



=> "X chases Y" is a pattern
for chases(X,Y)

Phrase structure can be a problem

KB

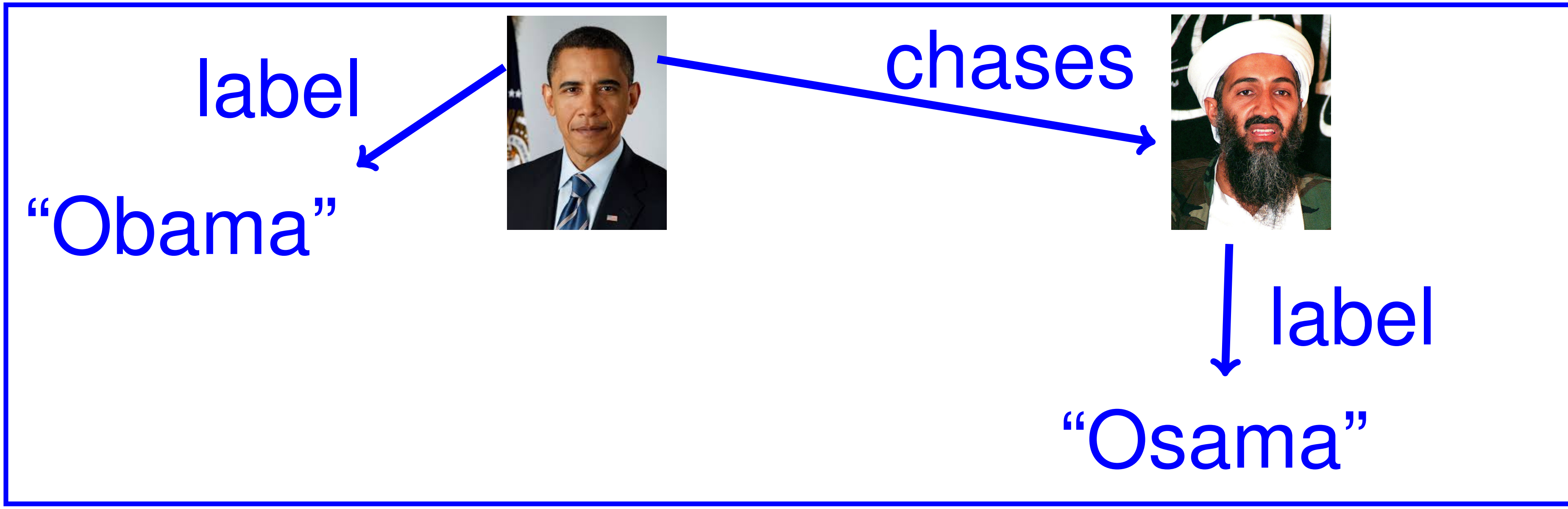


Corpus

\+ Obama hat Osama gejagt.

Phrase structure can be a problem

KB



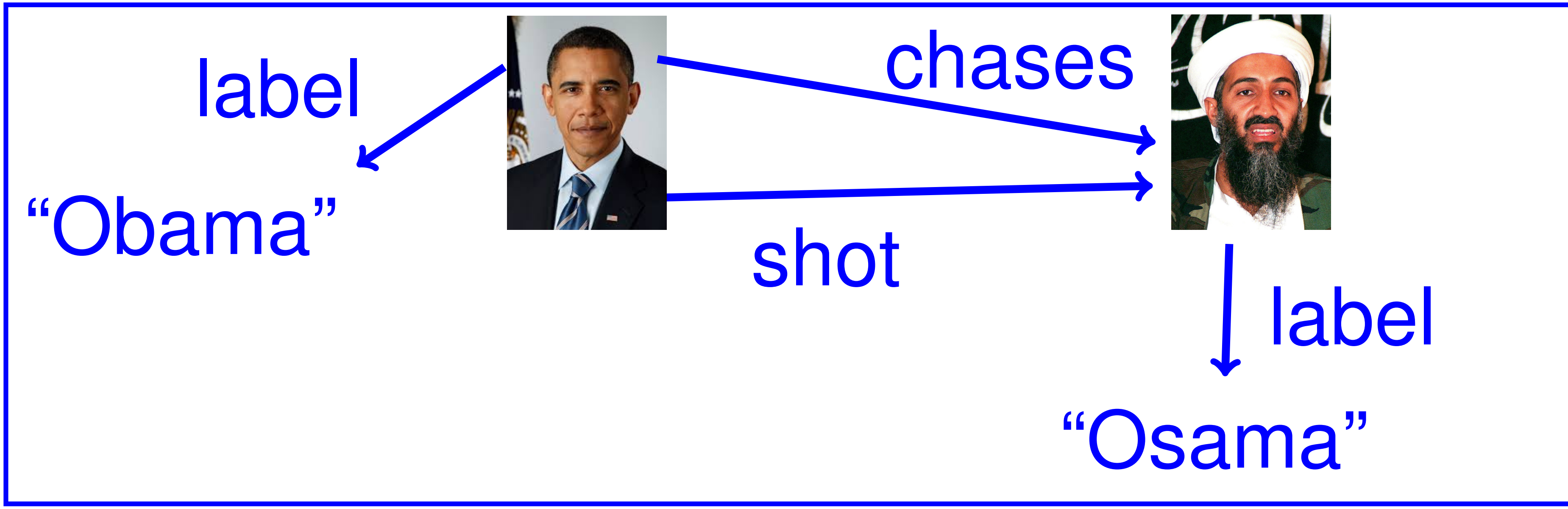
Corpus

+ Obama hat Osama gejagt.

=> “X hat Y” is a pattern
for chases(X,Y)?

Multiple links are a problem

KB

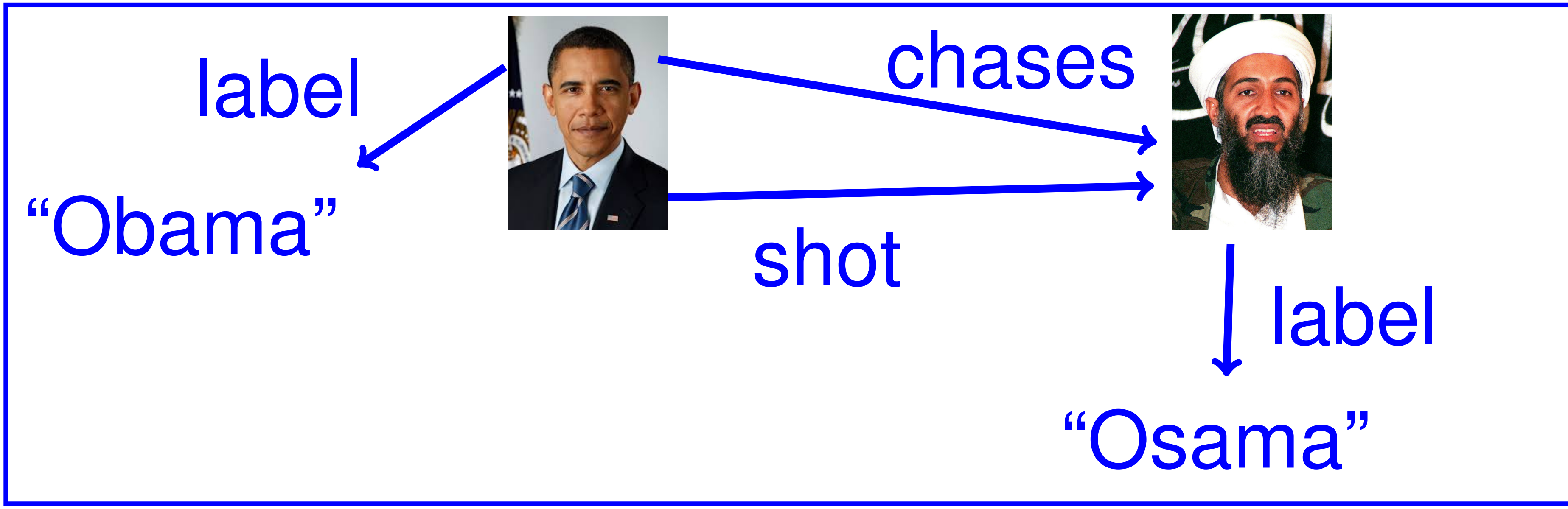


Corpus

\+ Obama chases Osama.

Multiple links are a problem

KB



Corpus

\+ Obama chases Osama.

=> "X chases Y" is a pattern

for chases(X,Y) or for shot(X,Y)?

Confidence of a pattern

The confidence of an extraction pattern is the number of matches that produce known facts divided by the total number of matches.

Pattern produces mostly new facts

=> risky

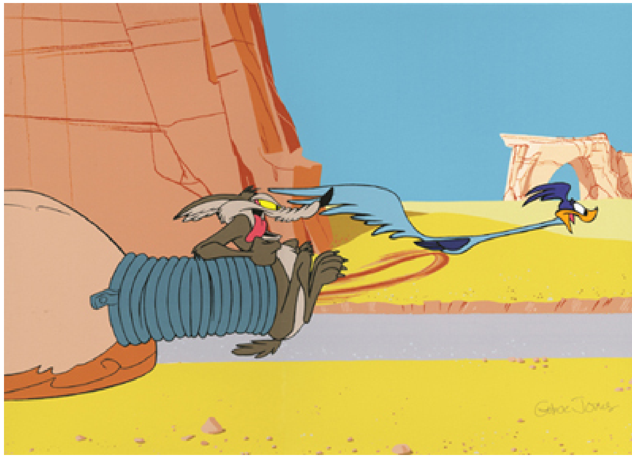
Pattern produces mostly known facts

=> safe

Simple word match is not enough

Coyote invents

a wonderful machine.



+

“X invents a Y”

=

invents(Coyote,wonderful)

Patterns may be too specific

Coyote invents

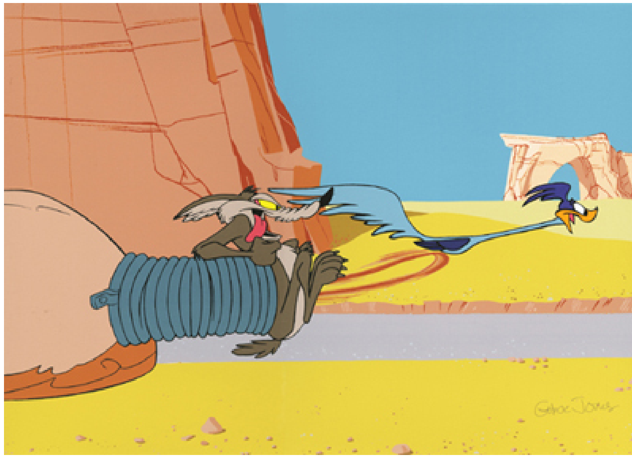
a wonderful machine.

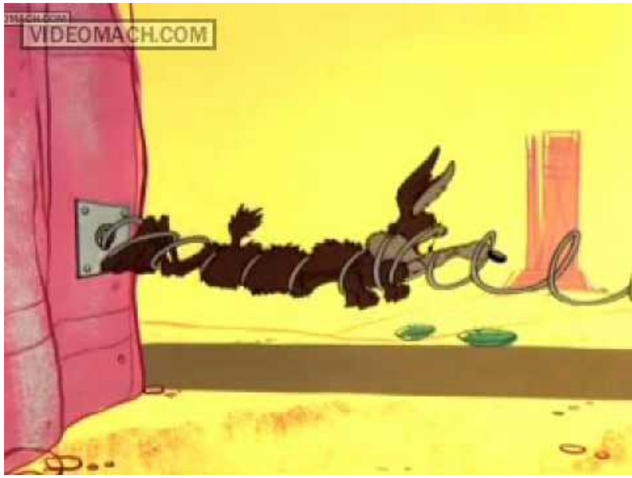
+

"X invents a great Y"

=

~~invents(Coyote,machine)~~





Overview

- Fact extraction from text
  - Difficulties
  - POS Tagging
  - Dependency grammars

Def: POS

A Part-of-Speech (also: POS, POS-tag, word class, lexical class, lexical category) is a set of words with the same grammatical role.

- Proper nouns (PN): Wile, Elvis, Obama...
- Nouns (N): desert, sand, ...
- Adjectives (ADJ): fast, funny, ...
- Adverbs (ADV): fast, well, ...
- Verbs (V): run, jump, dance, ...

Def: POS (2)

- Pronouns (PRON): he, she, it, this, ...
  (what can replace a noun)
- Determiners (DET): the, a, these, your,...
  (what goes before a noun)
- Prepositions (PREP): in, with, on, ...
  (what goes before determiner + noun)
- Subordinators (SUB): who, whose, that, which, because...
  (what introduces a sub-sentence)

Part of Speech

"Elvis wrote a really great song by himself"

Proper Name

Verb

Determiner

Adverb

Adjective

Noun

Preposition

Pronoun

A Part-of-Speech (also: POS, POS-tag, word class, lexical class, lexical category) is a set of words with the same grammatical role.

Def: POS-Tagging

POS-Tagging is the process of assigning to each word in a sentence its POS.

Wile tries a new machine.

PN V DET ADJ N

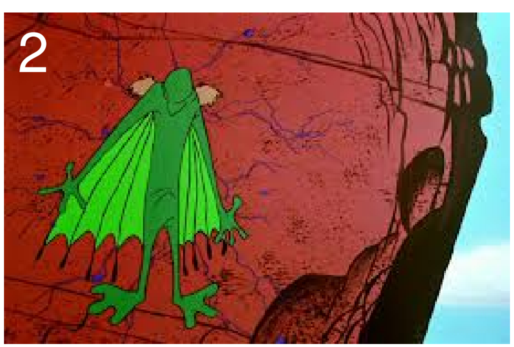
task>

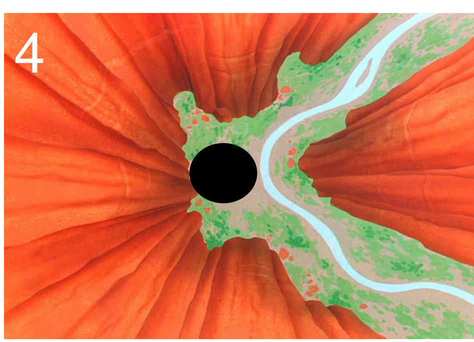
Task: POS-Tagging

POS-Tag the following sentence:

The coyote falls into a canyon.

- N, V, PN, ADJ, ADV, PRON
- Determiners (DET): the, a, these, your,...
  (what goes before a noun)
- Prepositions (PREP): in, with, on, ...
  (what goes before determiner + noun)
- Subordinators (SUB): who, whose, that,...
  (what introduces a sub-sentence)





Probabilistic POS-Tagging

Introduce random variables for words and for POS-tags:

	(visible)		(hidden)		
World	W1	W2	T1	T2	Probability
ω_1	Elvis	sings	PN	Verb	$P(\omega_1) = 0.2$
ω_2	Elvis	sings	Adj	Verb	$P(\omega_2) = 0.1$
ω_3	Elvis	runs	Prep	PN	$P(\omega_3) = 0.1$
	...		...		

Probabilistic POS-Tagging

Given a sentence $w_1, \ldots, w_n$

we want to find $argmax_{t_1,\ldots,t_n} \; P(t_1, \ldots, t_n, w_1, \ldots, w_n)$

World	W1	W2	T1	T2	Probability
ω_1	Elvis	sings	PN	Verb	$P(\omega_1) = 0.2$
ω_2	Elvis	sings	Adj	Verb	$P(\omega_2) = 0.1$
ω_3	Elvis	runs	Prep	PN	$P(\omega_3) = 0.1$
	...		...		

Markov Assumption 1

Every tag depends just on its predecessor

$$P(T_i|T_1, ..., T_{i-1}) = P(T_i|T_{i-1})$$

Markov Assumption 1

Every tag depends just on its predecessor

$$P(T_i|T_1, ..., T_{i-1}) = P(T_i|T_{i-1})$$

The probability that PN, V, D is followed by a noun is the same as the probability that D is followed by a noun:

$$P(N \mid PN, V, D) = P(N \mid D)$$

Markov Assumption 1

Every tag depends just on its predecessor

$$P(T_i|T_1, ..., T_{i-1}) = P(T_i|T_{i-1})$$

The probability that PN, V, D is followed by a noun is the same as the probability that D is followed by a noun:

$$P(N \,|\, PN, V, D) = P(N \,|\, D)$$

Elvis	sings	a	song
PN	Verb	Det	?

Markov Assumption 1

Every tag depends just on its predecessor

$$P(T_i|T_1,...,T_{i-1}) = P(T_i|T_{i-1})$$

The probability that PN, V, D is followed by a noun is the same as the probability that D is followed by a noun:

$$P(N\,|\,PN, V, D) = P(N\,|\,D)$$

Elvis	sings	a	song
PN	Verb	Det	?

Markov Assumption 2

Every word depends just on its tag:

$$P(W_i|W_1, ..., W_{i-1}, T_1, ..., T_i) = P(W_i|T_i)$$

Markov Assumption 2

Every word depends just on its tag:

$$P(W_i | W_1, ..., W_{i-1}, T_1, ..., T_i) = P(W_i | T_i)$$

The probability that the 4th word is “song” depends just on the tag of that word:

$$P(song \,|\, Elvis,\ sings,\ a,\ PN,\ V,\ D,\ N) = P(song \,|\, N)$$

Markov Assumption 2

Every word depends just on its tag:

$$P(W_i|W_1, ..., W_{i-1}, T_1, ..., T_i) = P(W_i|T_i)$$

The probability that the 4th word is “song” depends just on the tag of that word:

$$P(song \,|\, Elvis,\ sings,\ a,\ PN,\ V,\ D,\ N) = P(song \,|\, N)$$

Elvis	sings	a	?
PN	Verb	Det	Noun

Markov Assumption 2

Every word depends just on its tag:

$$P(W_i|W_1, ..., W_{i-1}, T_1, ..., T_i) = P(W_i|T_i)$$

The probability that the 4th word is "song" depends just on the tag of that word:

$$P(song\,|\,Elvis,\ sings,\ a,\ PN,\ V,\ D,\ N) = P(song\,|\,N)$$

Elvis	sings	a	?
PN	Verb	Det	Noun

Homogeneity Assumption 1

The tag probabilities are the same at all positions

$$P(T_i|T_{i-1}) = P(T_k|T_{k-1}) \; \forall \, i, k$$

Homogeneity Assumption 1

The tag probabilities are the same at all positions

$$P(T_i|T_{i-1}) = P(T_k|T_{k-1}) \; \forall \, i, k$$

The probability that a Det is followed by a Noun is the same at position 7 and 2:

$$P(T_7 = Noun \,|\, T_6 = Det) = P(T_2 = Noun \,|\, T_1 = Det)$$

Homogeneity Assumption 1

The tag probabilities are the same at all positions

$$P(T_i|T_{i-1}) = P(T_k|T_{k-1}) \; \forall \, i, k$$

The probability that a Det is followed by a Noun is the same at position 7 and 2:

$$P(T_7 = Noun \,|\, T_6 = Det) = P(T_2 = Noun \,|\, T_1 = Det)$$

Let's denote this probability by

$P(Noun|Det)$ "Transition probability"

$$P(s|t) := P(T_i = s|T_{i-1} = t) \;\; (for\, any\, i)$$

Homogeneity Assumption 2

The word probabilities are the same at all positions

$$P(W_i|T_i) = P(W_k|T_k)\ \forall\, i, k$$

Homogeneity Assumption 2

The word probabilities are the same at all positions

$$P(W_i|T_i) = P(W_k|T_k)\ \forall\, i, k$$

The probability that a PN is "Elvis"

is the same at position 7 and 2:

$$P(W_7 = Elvis \,|\, T_7 = PN) = P(W_2 = Elvis \,|\, T_2 = PN) =$$

Homogeneity Assumption 2

The word probabilities are the same at all positions

$$P(W_i|T_i) = P(W_k|T_k) \; \forall \, i, k$$

The probability that a PN is "Elvis"

is the same at position 7 and 2:

$$P(W_7 = Elvis \,|\, T_7 = PN) = P(W_2 = Elvis \,|\, T_2 = PN) =$$

Let's denote this probability by

$$P(Elvis|PN)$$ "Emission probability"

$$P(w|t) := P(W_i = w|T_i = t) \;\; (for\, any\, i)$$

Def: HMM

A (homogeneous) Hidden Markov Model (also: HMM) is a sequence of random variables, such that

$$P(\underbrace{w_1, \ldots, w_n}_{\text{Words of the sentence}}, \underbrace{t_1, \ldots, t_n}_{\text{POS-tags}}) = \prod_i P(w_i|t_i) \times P(t_i|t_{i-1})$$

Emission probabilities

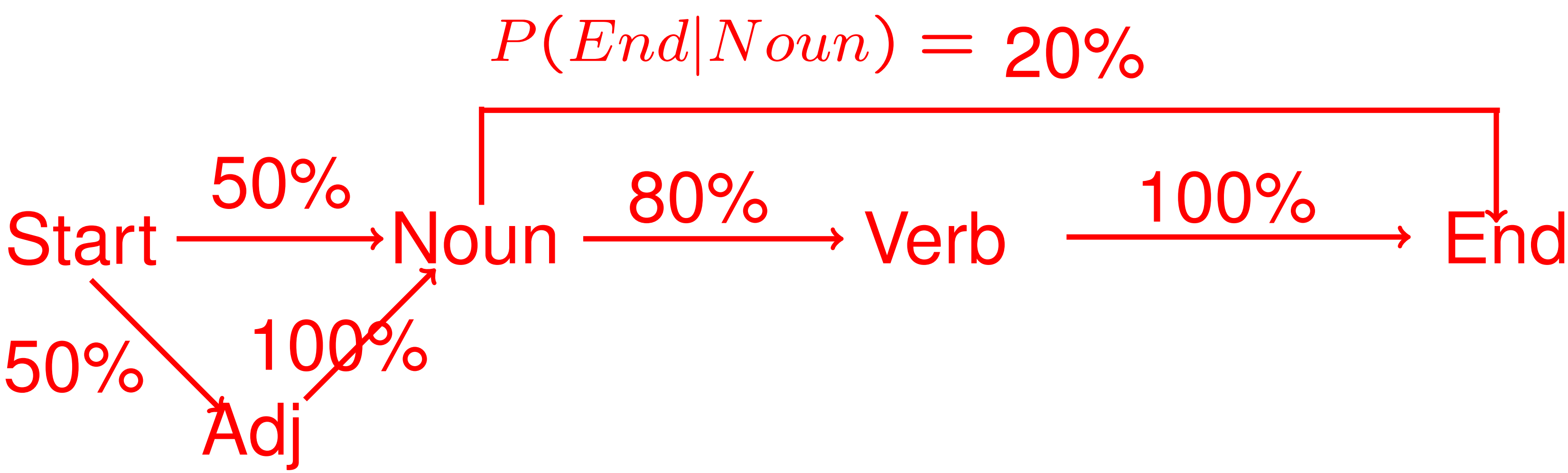
Transition probabilities

... with $t_0 = Start$

HMMs as graphs

$$P(w_1, ..., w_n, t_1, ..., t_n) = \prod_i P(w_i|t_i) \times P(t_i|t_{i-1})$$

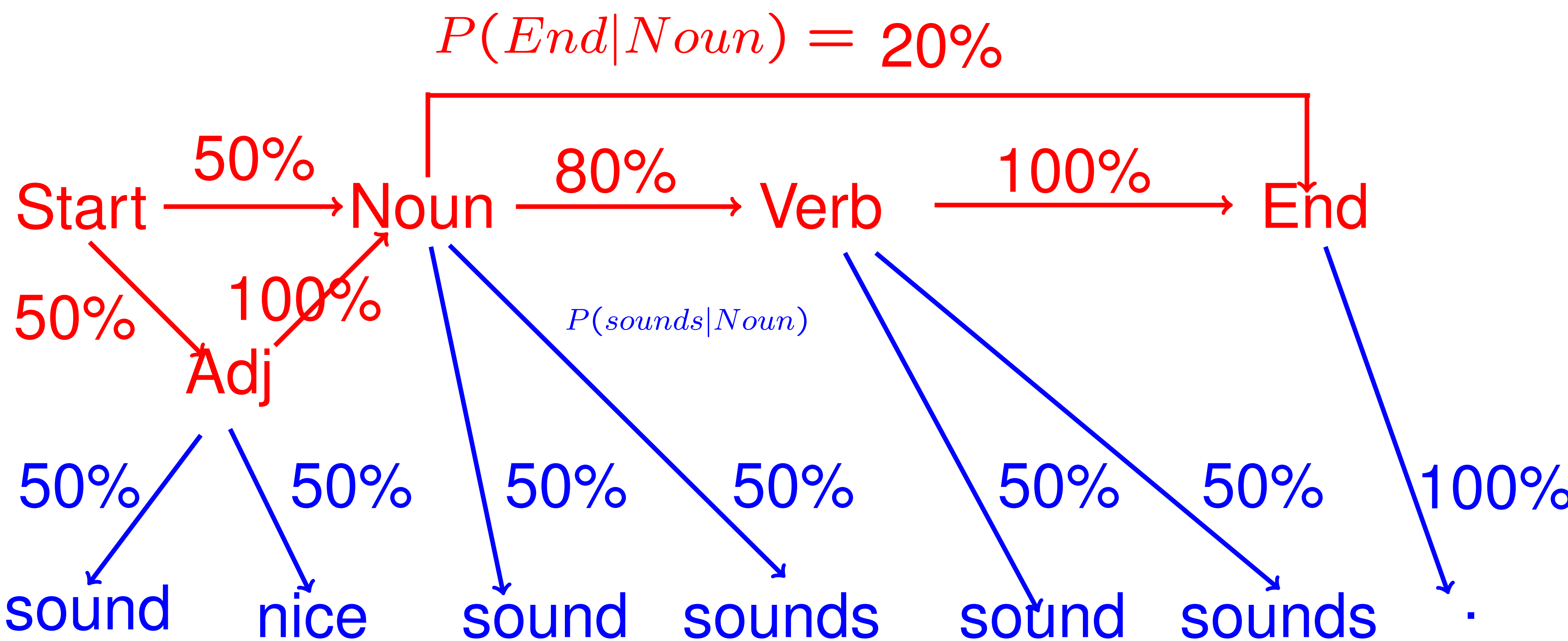
Transition probabilities



HMMs as graphs

$$P(w_1, ..., w_n, t_1, ..., t_n) = \prod_i P(w_i|t_i) \times P(t_i|t_{i-1})$$

Emission probabilities

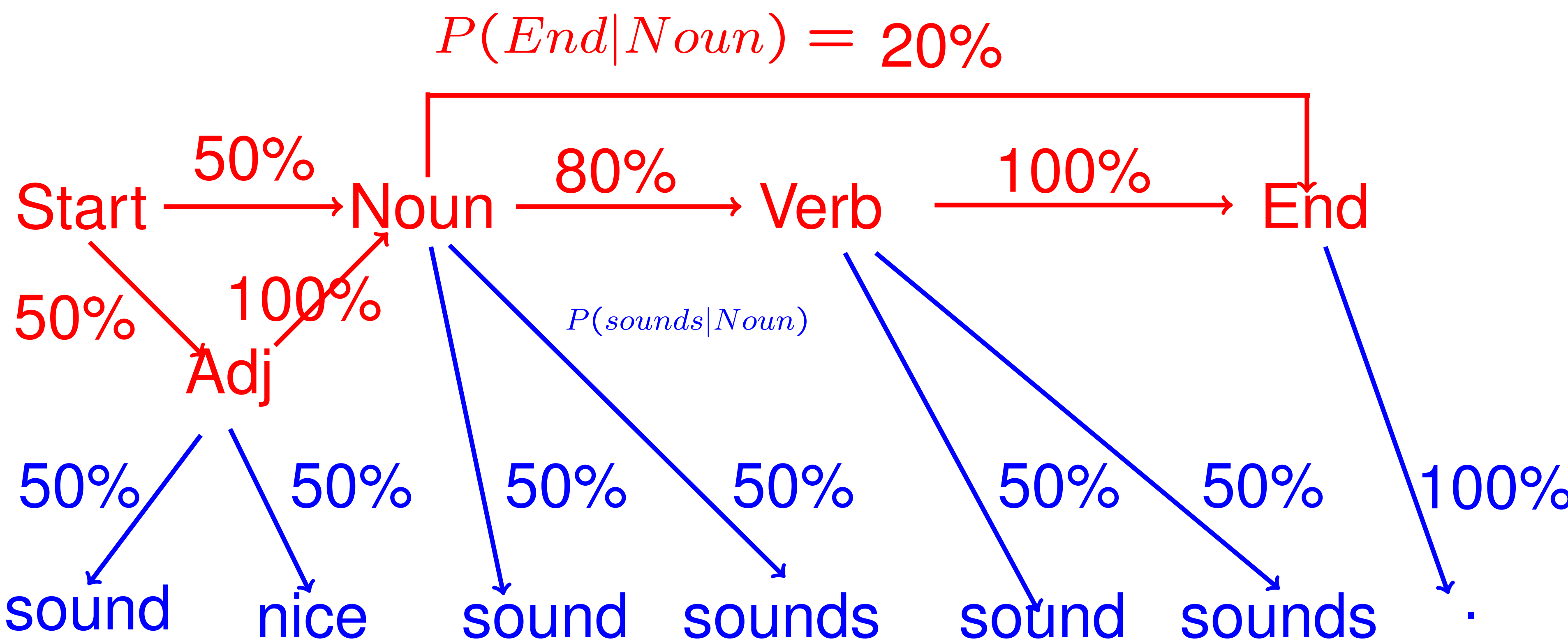


HMMs as graphs

$$P(w_1, ..., w_n, t_1, ..., t_n) = \prod_i P(w_i|t_i) \times P(t_i|t_{i-1})$$

P(nice, sounds, ., Adj, Noun, End) =

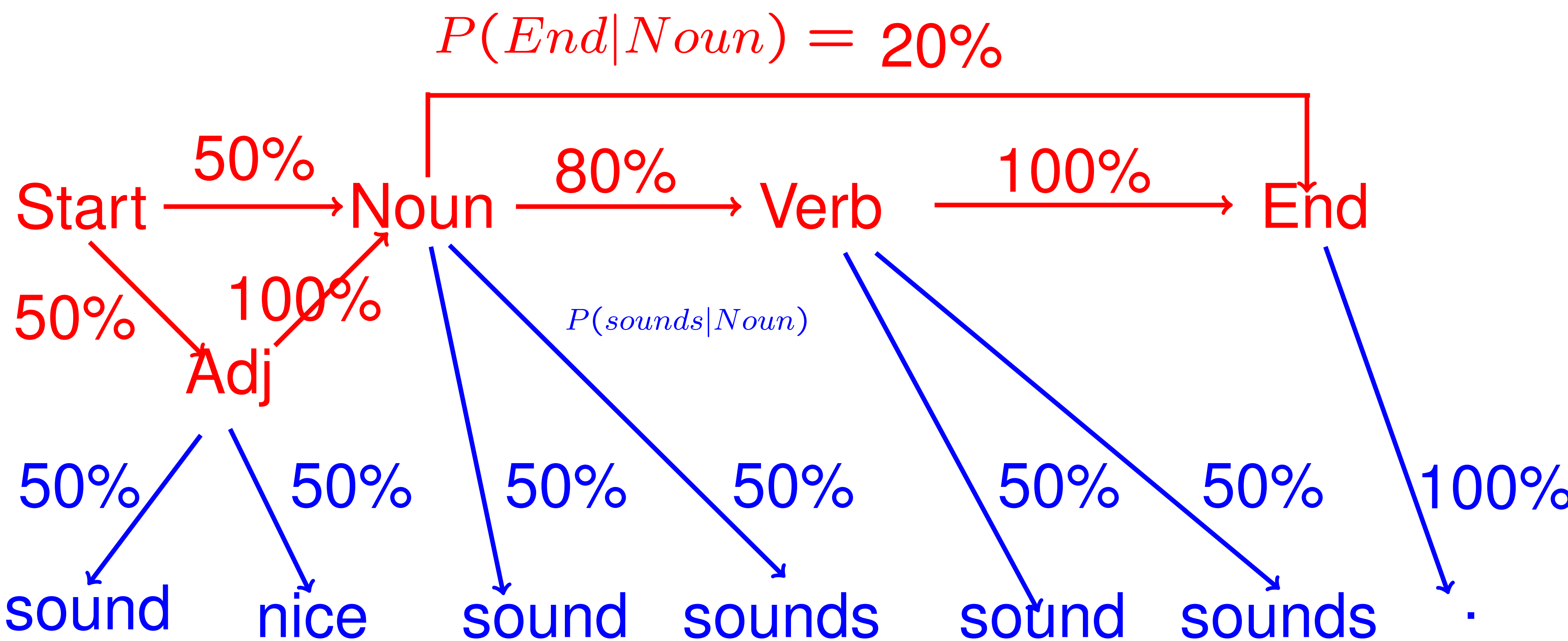
50% * 50% * 100% * 50% * 20% * 100% = 2.5%



HMM Question

What is the most likely tagging for

“sound sounds .” ?



POS tagging with HMMs

What is the most likely tagging for

“sound sounds .” ?

Adj + Noun: 50%*50%*100%*50%*20%*100% = 2.5%

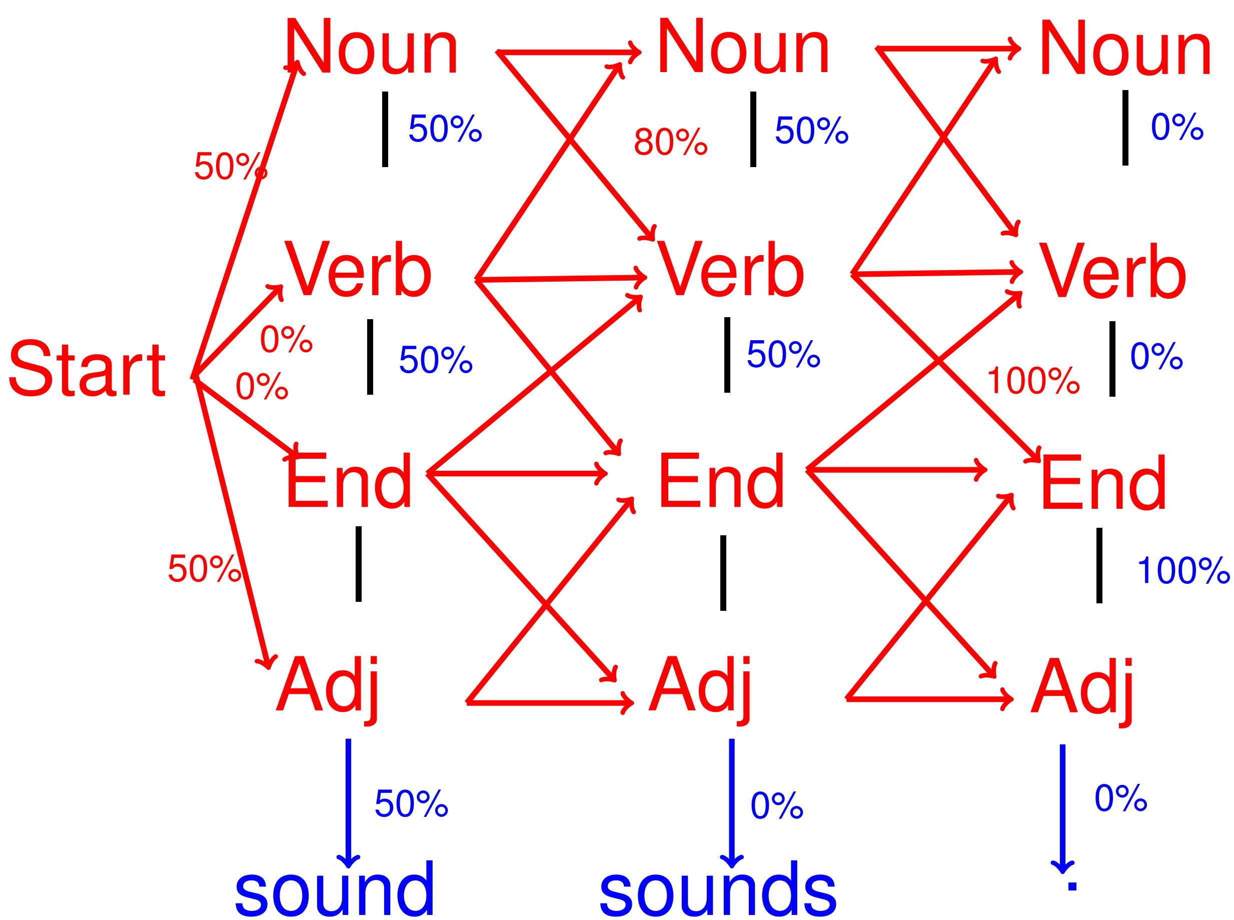
Noun + Verb: 50%*50%*80%*50%*100% = 10%

Finding the most likely sequence of tags

that generated a sentence is POS tagging (hooray!).

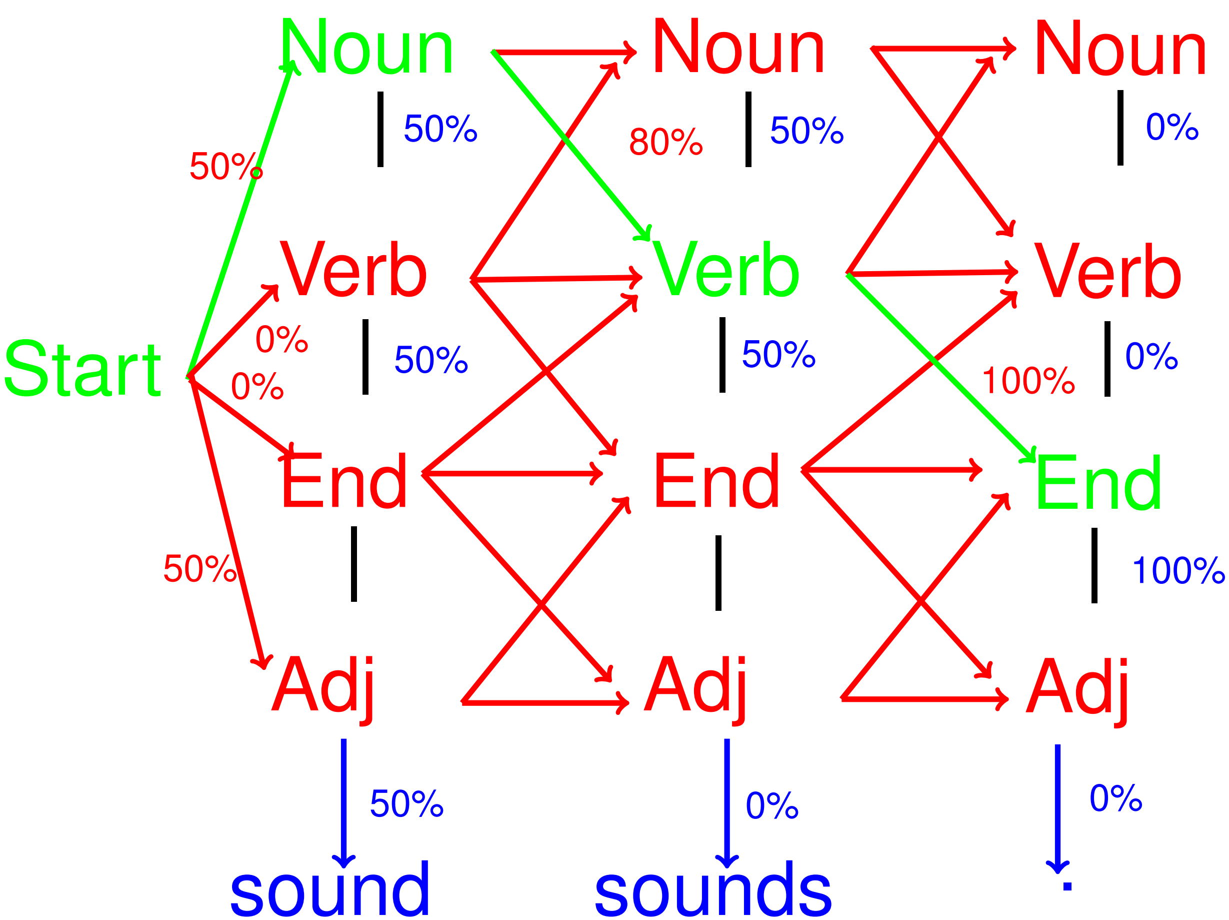
The HMM as a Trellis

Task: compute the probability of every possible path from left to right, find maximum.



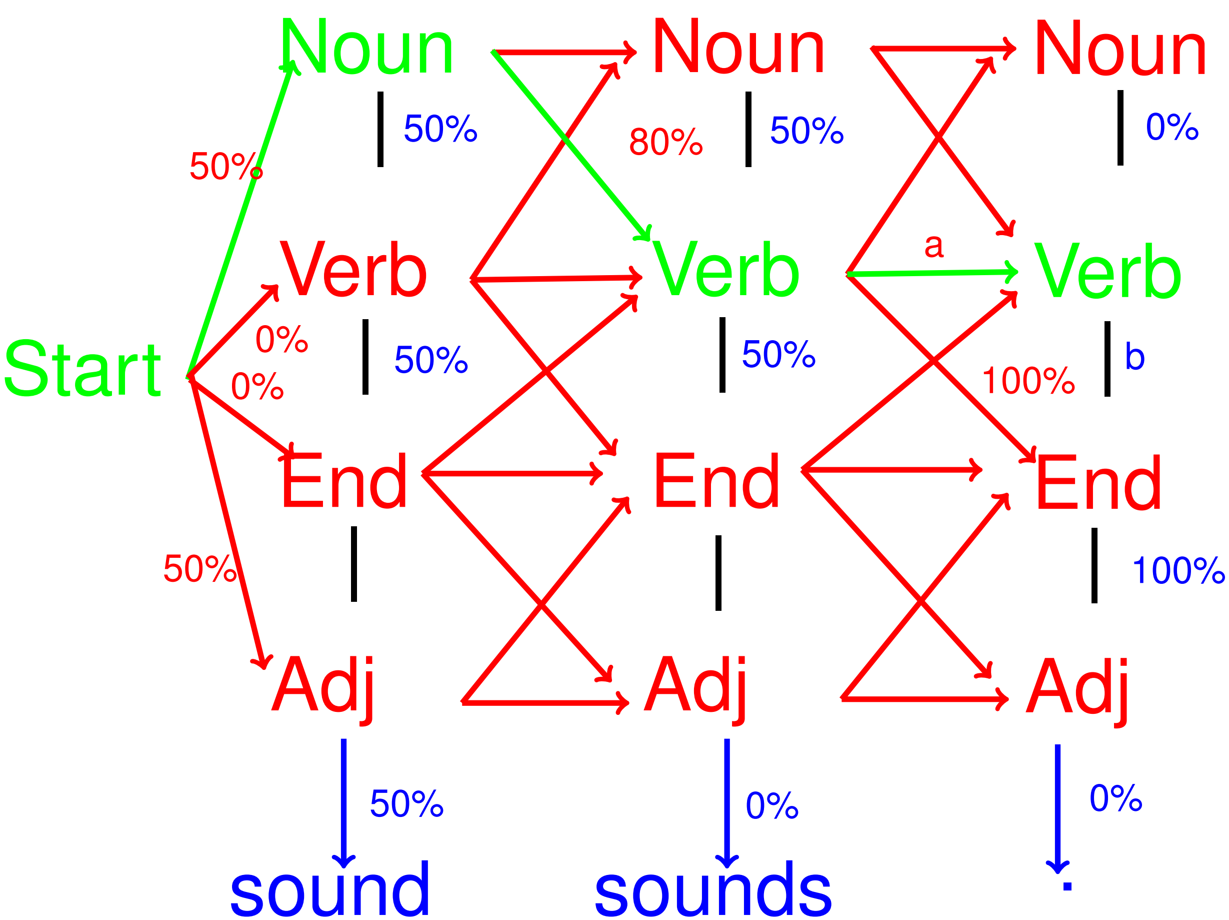
The HMM as a Trellis

50%*50%* 80%*50%* 100%*100% = 10%



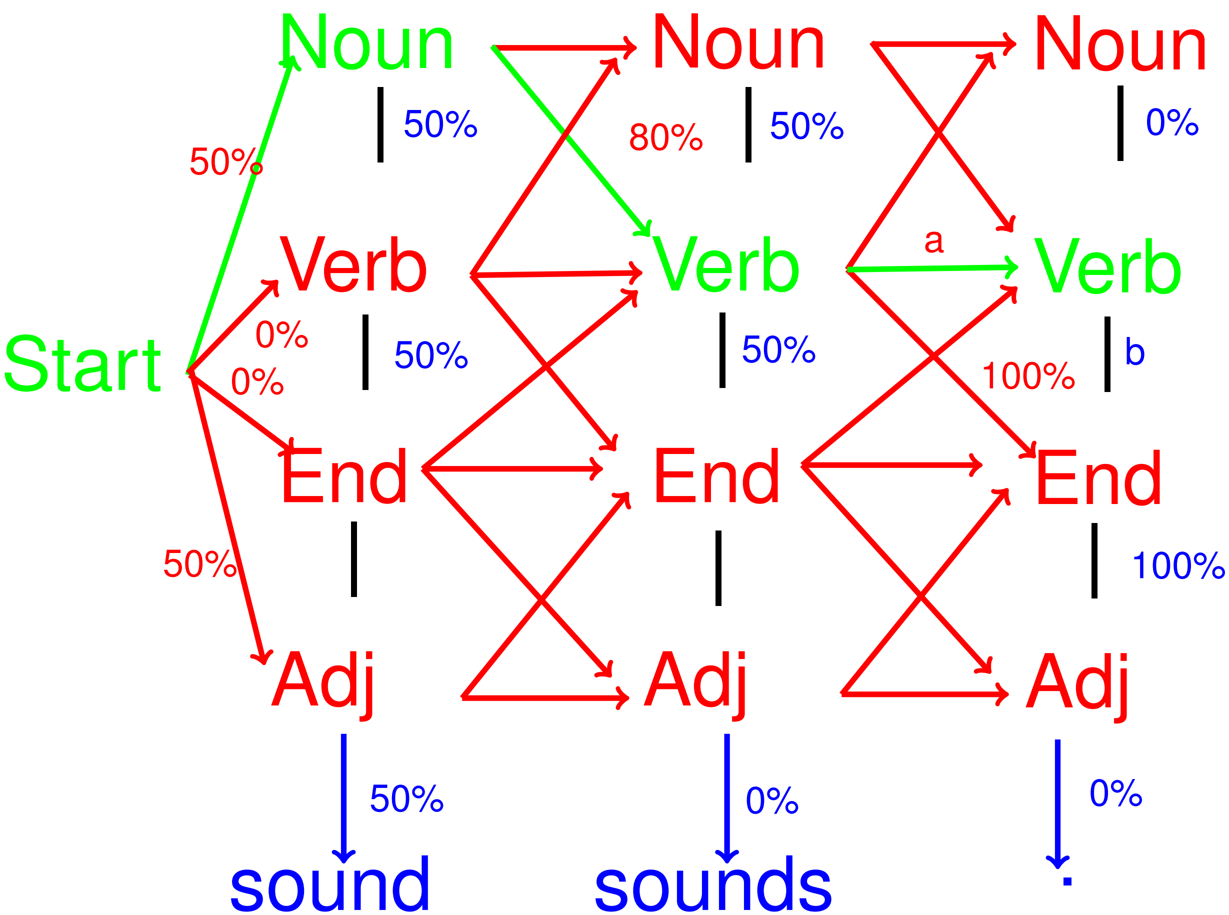
The HMM as a Trellis

50%*50%* 80%*50%* a*b



The HMM as a Trellis

50%*50%* 80%*50%* a*b



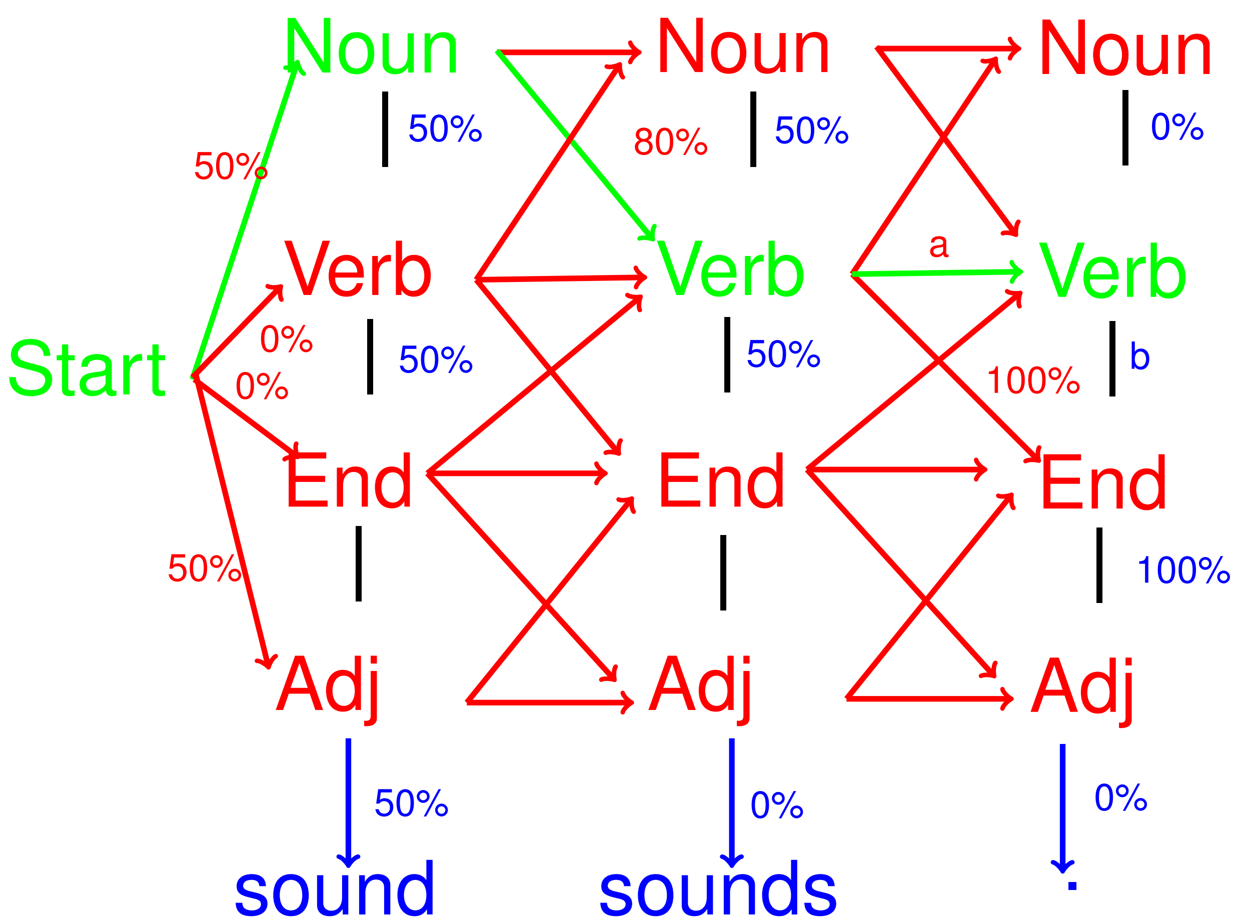
Complexity:

$O(T^S)$

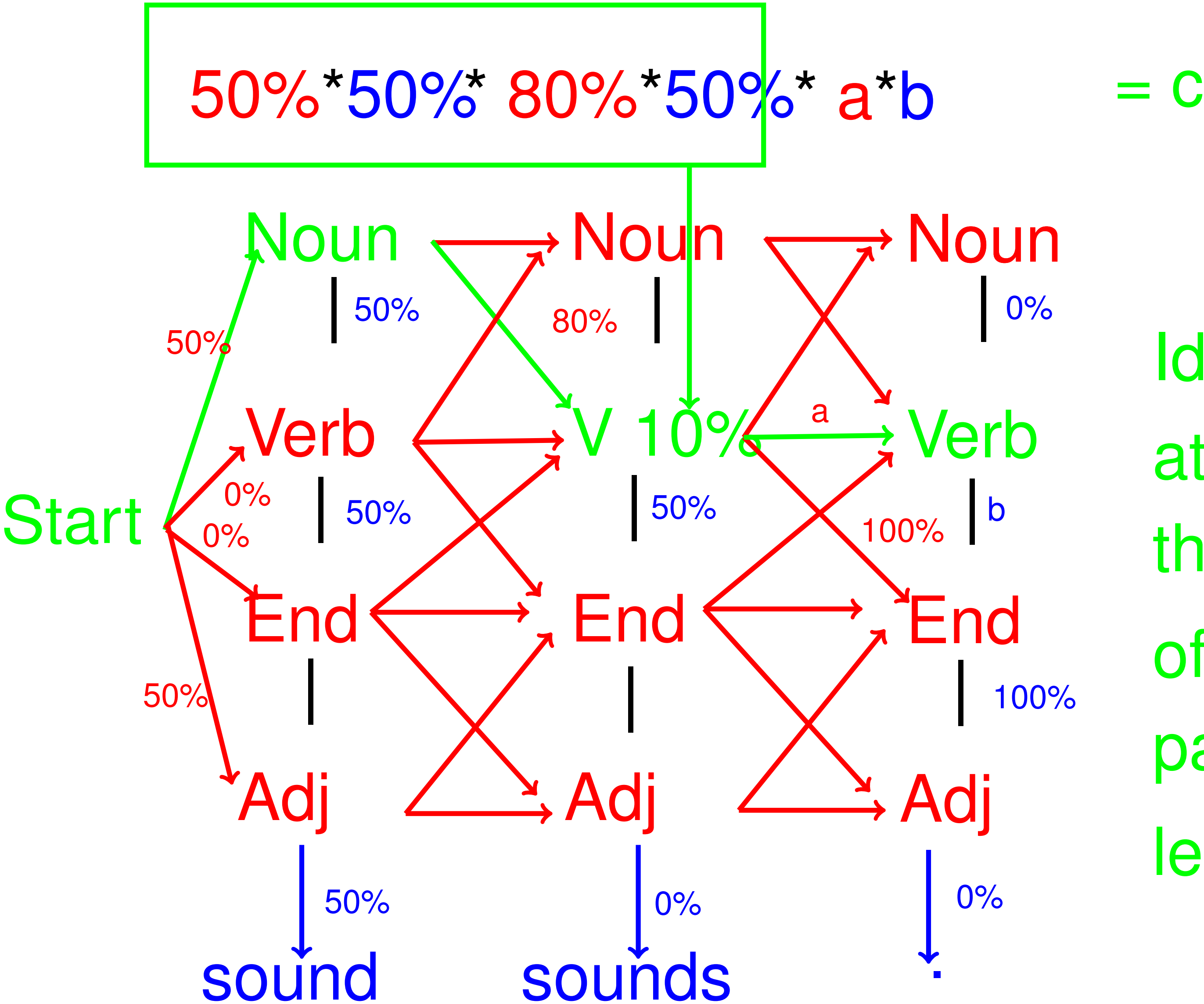
The HMM as a Trellis

same as before!

$\boxed{50\%*50\%*80\%*50\%}*a*b \quad = c$

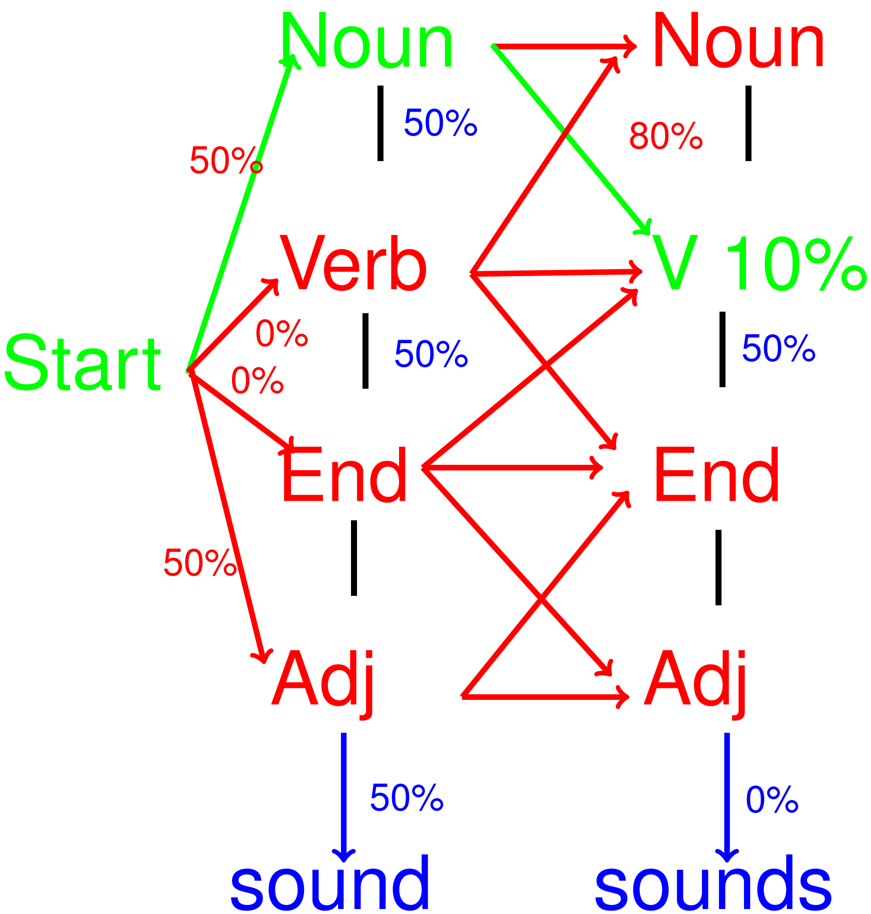


Viterbi Idea



Idea: Store at each node the probability of the maximal path that leads there.

Viterbi Algorithm

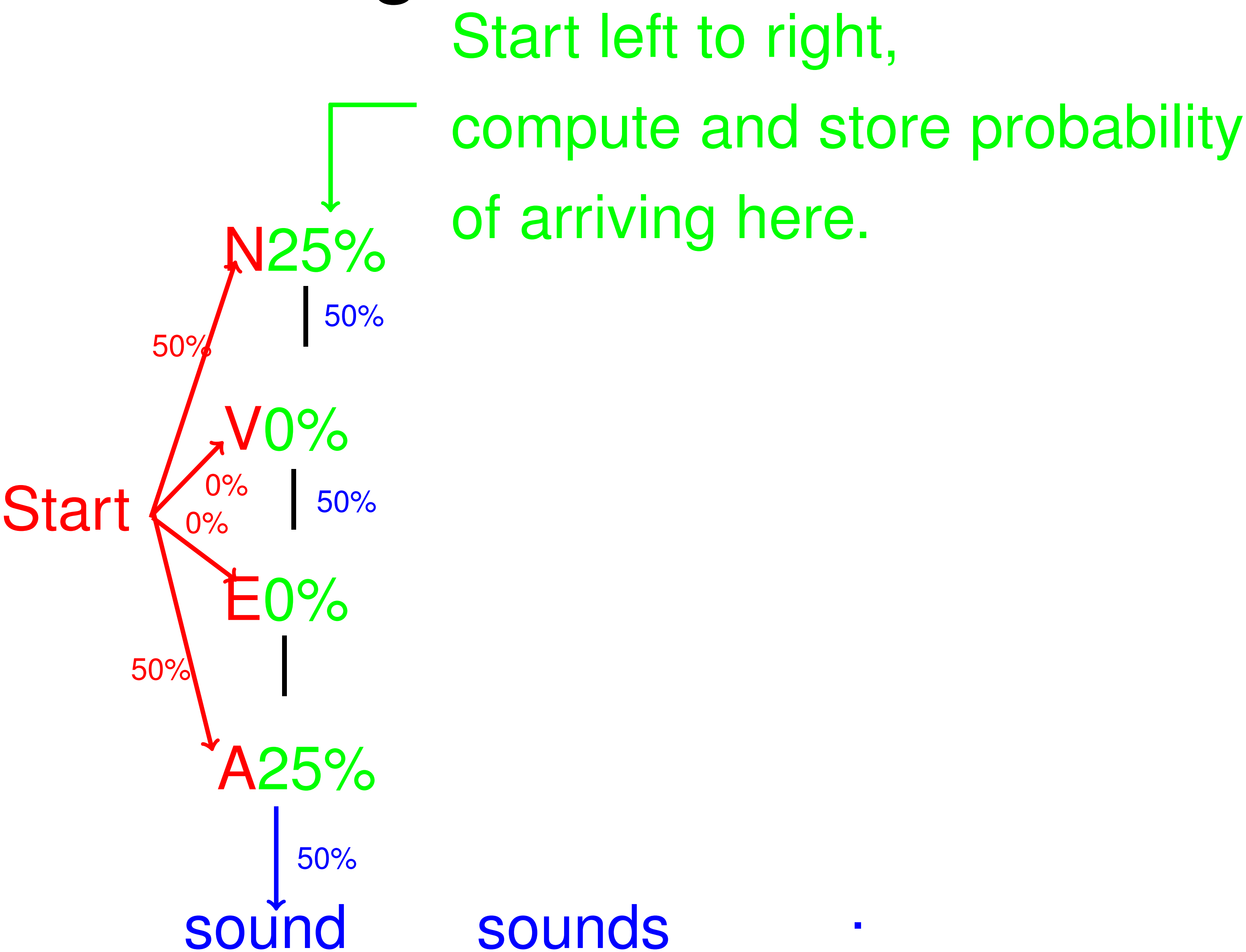


- For each word w
  - for each tag t
    - for each preceding tag t'
      - compute

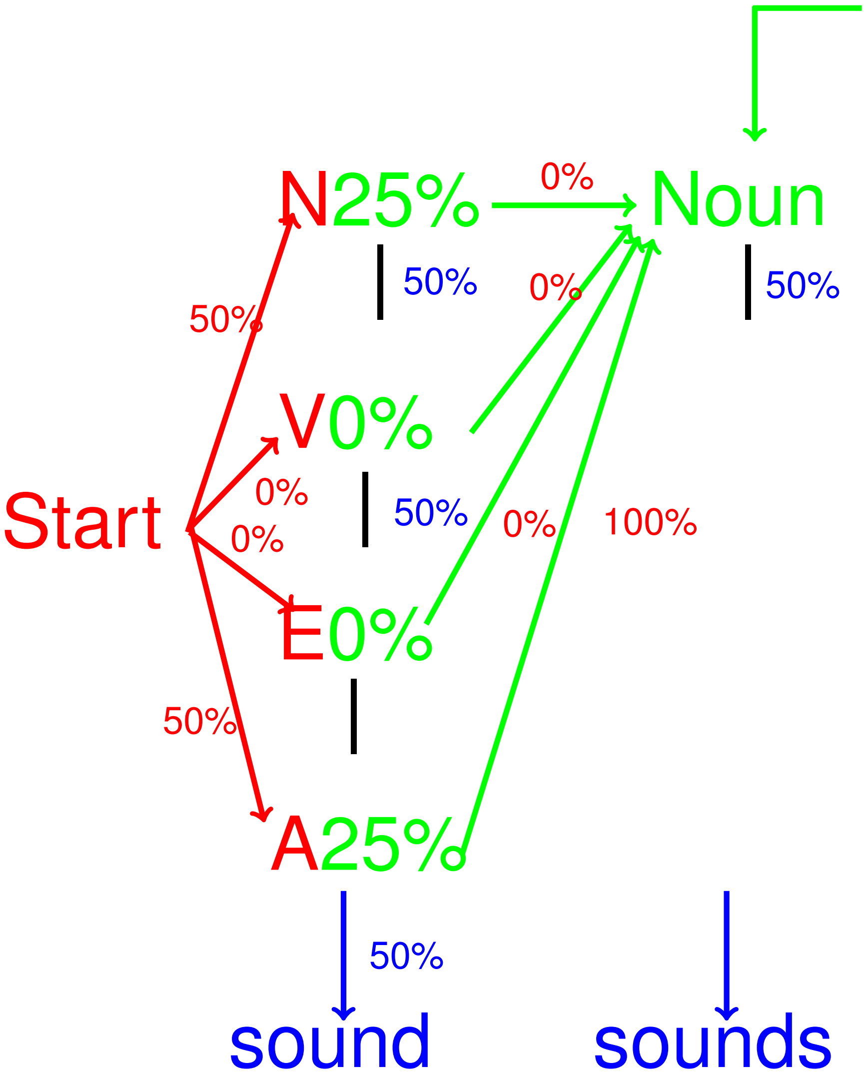
        $$P(t') \times P(t|t') \times P(w|t)$$

    - store the maximal probability at t, w

Viterbi Algorithm



Viterbi Algorithm



Find prev that maximizes

$P(prev) \times P(t|prev) \times P(w|t)$

$prev = Noun$ 25%*0%*50%

$prev = Verb$ 0%*0%*50%

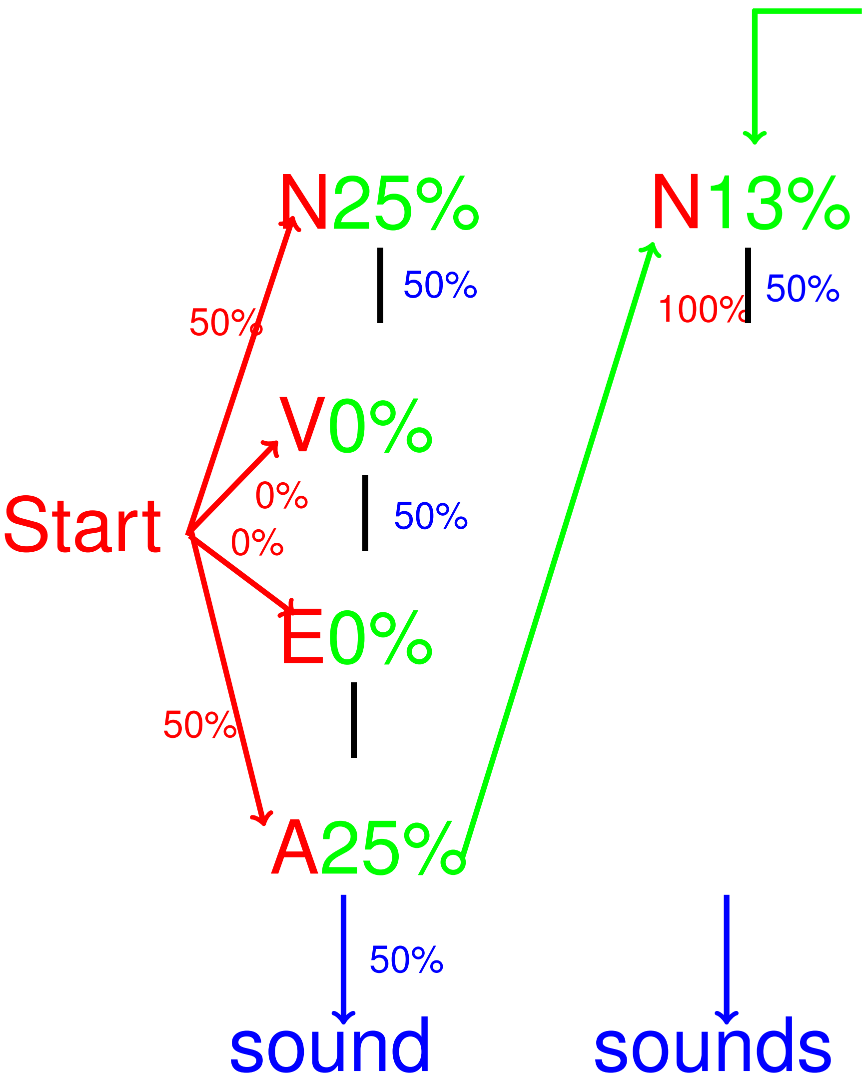
$prev = End$ 0%*0%*50%

$prev = Adj$ 25%*100%*50%

Viterbi Algorithm

Find prev that maximizes

$$P(prev) \times P(t|prev) \times P(w|t)$$



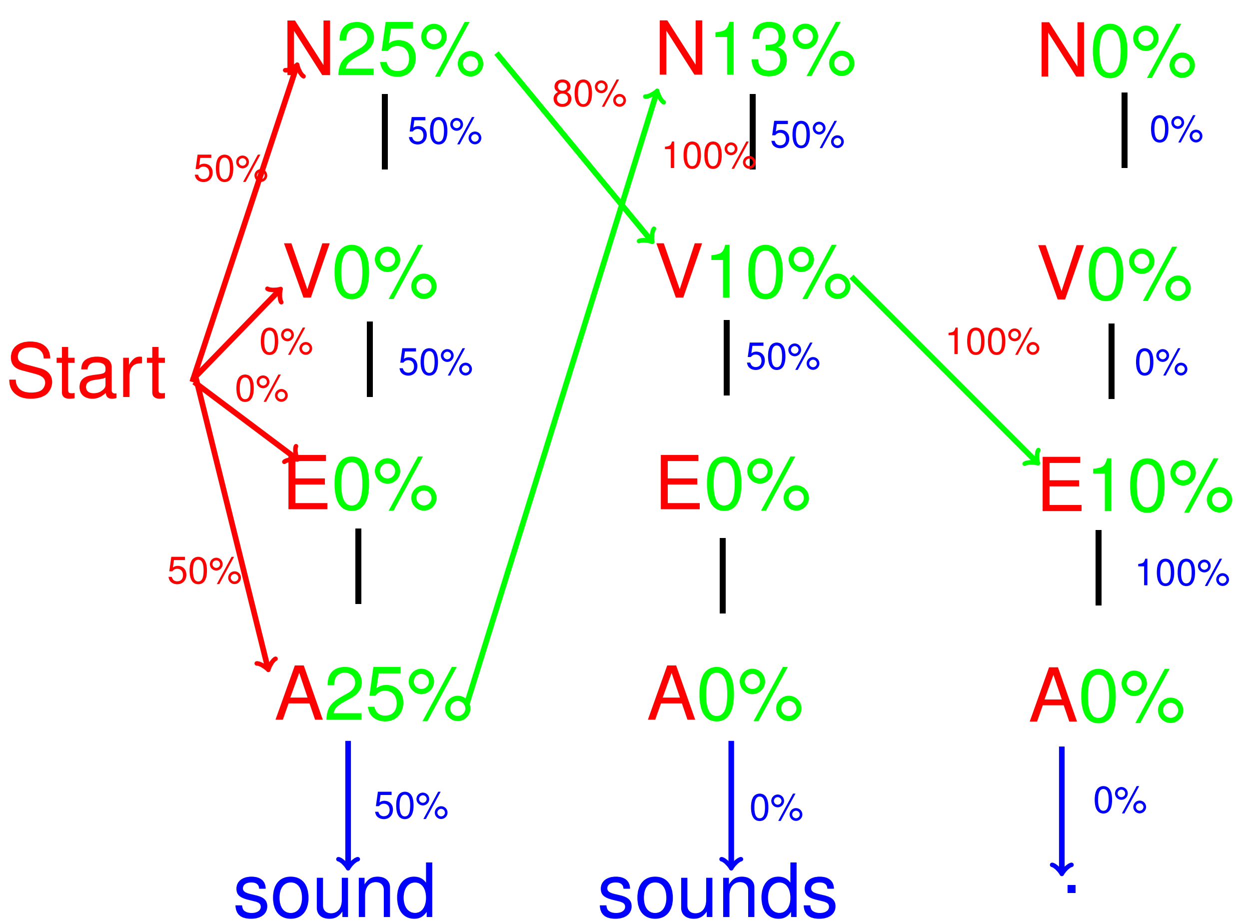
$prev = Noun$ 25%*0%*50%

$prev = Verb$ 0%*0%*50%

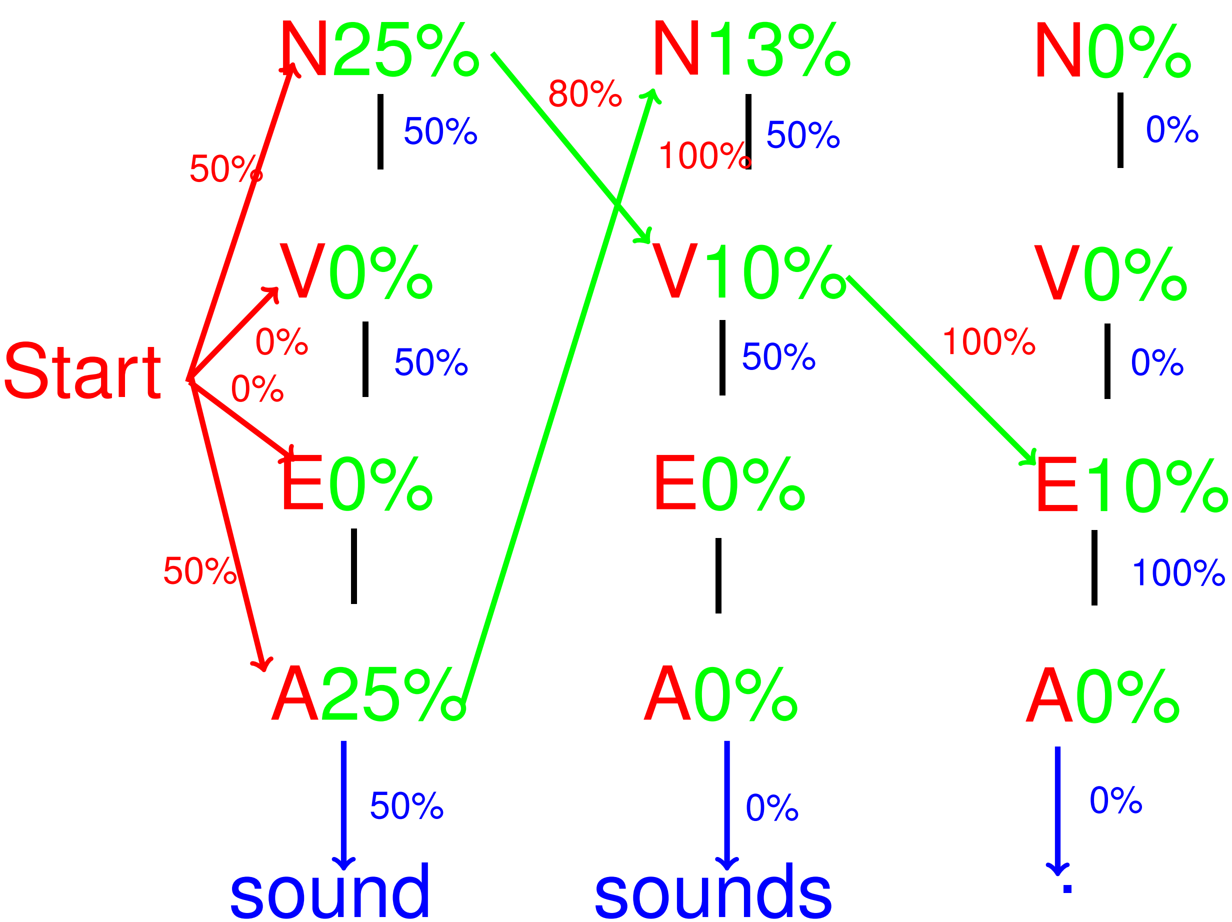
$prev = End$ 0%*0%*50%

$prev = Adj$ 25%*100%*50%

Viterbi Algorithm



Viterbi Algorithm



Read best path by following arrows backwards:

Start N V E

$$O(S \times T^2)$$

Where do we get the HMM?

Estimate probabilities from manually annotated corpus:

Elvis/PN sings/Verb

Elvis/PN ./End

Priscilla/PN laughs/Verb

$$P(Verb|PN) = \frac{2}{3} \quad P(End|PN) = \frac{1}{3} \quad \cdots$$

$$P(Elvis|PN) = \frac{2}{3} \quad P(sings|Verb) = \frac{1}{2} \cdots$$

Def: Probabilistic POS Tagging

Given a sentence and transition and emission probabilities, Probabilistic POS Tagging computes the sequence of tags that has maximal probability (in an HMM).

$\vec{X} =$ Elvis sings.

$P(Elvis, sings, PN, N) = 0.01$

$P(Elvis, sings, V, N) = 0.01$

winner

$P(Elvis, sings, PN, V) = 0.1$

...

Probabilistic POS Tagging

- Probabilistic POS tagging uses
  Hidden Markov Models
- General performance very good (>95% acc.)
- Several POS taggers are available
    - Stanford POS tagger
    - MBT: Memory-based Tagger
    - TreeTagger
    - ACOPOST
    - YamCha
    - ...

(HMMs and the Viterbi algorithm serve a wide variety of other tasks,
in particular NLP at all levels of granularity, e.g., in speech processing)

Research Questions

How can we deal with

- evil cases?
  - the word “blue” has 4 letters.
  - pre- and post-secondary
  - look it up
  - The Duchess was entertaining last night.

[Wikipedia/POS tagging]

- unknown words?
- new languages?

POS Tagging helps pattern IE

We can choose to match the place-holders with only nouns or proper nouns:

"X invents a Y"

Coyote invents a catapult

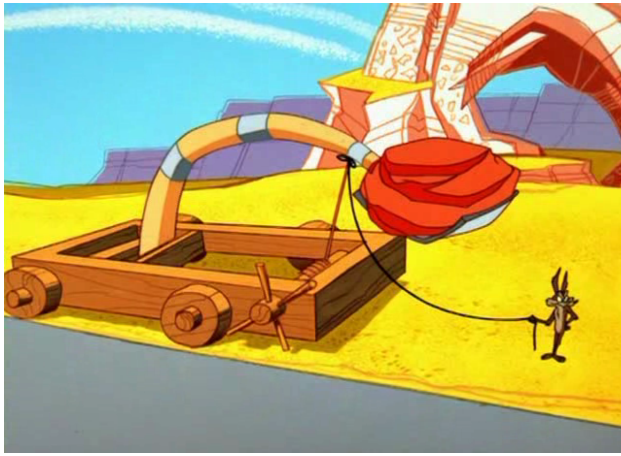
invents(Coyote,catapult)

Coyote invents a really cool thing.

~~invents(Coyote,really)~~

POS-tags can generalize patterns

"X invents a ADJ X"



Coyote invents a cool catapult — match

Coyote invents a great catapult — match

Coyote invents a super-duper catapult. — match

Phrase structure is a problem

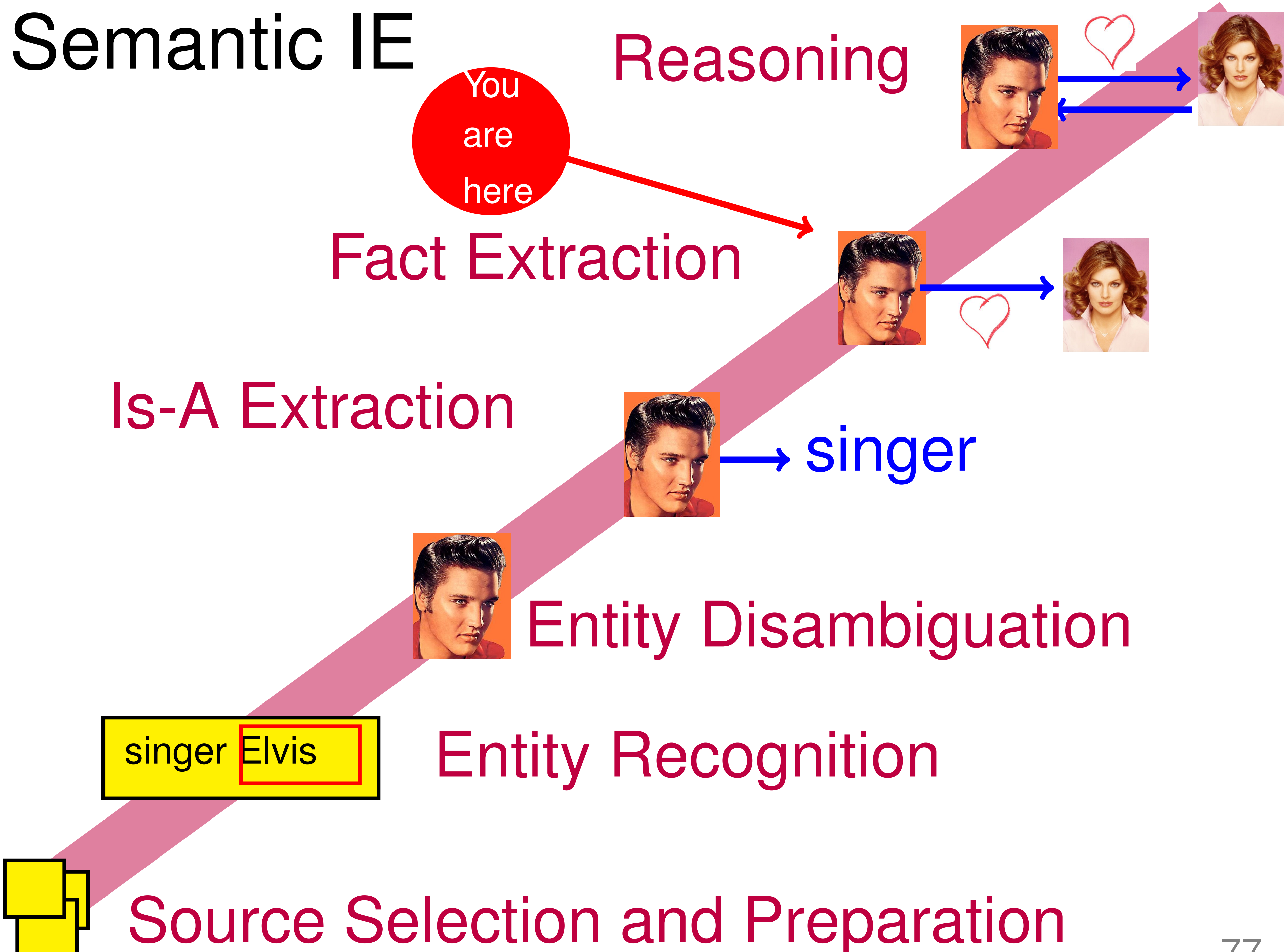
“X invents a ADJ X”

Coyote invents a very great catapult. — no match

Coyote, who is very hungry, invents a great catapult. — no match

We will see next time how to solve it.

Semantic IE


Reasoning
You are here
Fact Extraction
Is-A Extraction
singer
Entity Disambiguation
singer Elvis
Entity Recognition
Source Selection and Preparation

References

Brin: Extracting Patterns and Relations from the WWW

Agichtein: Snowball

Ramage: HMM Fundamentals

Web data mining class