Natural Language Models and Interfaces

BSc Artificial Intelligence

Lecturer: Wilker Aziz
Institute for Logic, Language, and Computation

2018, week 3, lecture a

Problems with n-gram LMs

Estimation

- number of parameters grows exponentially in n

$$O(v^n)$$

- Zipf's law tells us most words will be extremely rare
  n-grams are even sparser

What can we do beyond smoothing and interpolation?

NLMI

Parts of speech

Hidden Markov Models

Evaluation

Generalisations in language

We can organise words into classes

- semantic criteria: what does the word refer to?
  nouns often refer to ‘people’, ‘places’ or ‘things’
- formal criteria: what form does the word have?
  -ly makes an adverb out of an adjective
  -tion makes a noun out of a verb
- distributional criteria: in what contexts can the word occur?
  adjectives precede nouns

Adapted from S. Goldwater, S. Cohen, T. Deoskar

Criteria for classifying words

	Semantically	Formally	Distributionally
Nouns	refer to things, concepts	-ness, -tion, -ity, -ance	After determiners, possessives
Verbs	refer to actions, states	-ate, -ize	infinitives: to jump, to learn
Adjectives	properties of nouns	-al, -ble	appear before nouns
Adverbs	properties of actions	-ly	next to verbs, beginning of sentence

Importance of formal and distributional criteria

Often in text, we come across unknown words

And, as in uffish thought he stood,
The Jabberwock, with eyes of flame,
Came whiffling through the tulgey wood,
And burbled as it came!

Formal and distributional criteria help one recognise which class an unknown word belongs to:

Those zorls you splarded were malgy

Parts of Speech

- Open class words (or content words)
  - nouns, verbs, adjectives, adverbs
  - mostly content-bearing
    they refer to objects, actions, and features in the world
  - open class, since there is no limit to what these words are
    new ones are added all the time (email, website, selfie)
- Closed class words (or function words)
  - pronouns, determiners, prepositions, connectives, ...
  - there is a limited number of these
  - mostly functional: to tie the concepts of a sentence together

---

Adapted from T. Deoskar

But how many parts of speech

- Both linguistic and practical considerations
- Corpus annotators decide. Distinguish between
  - proper nouns (names) and common nouns ?
  - past and present tense verbs?
  - auxiliary and main verbs?

English POS tag sets

Brown corpus (87 tags)

- one of the earliest large corpora collected for computational linguistics (1960s)
- balanced corpus: different genres (fiction, news, academic, editorial, etc)

Penn Treebank corpus (45 tags)

- first large corpus annotated with POS and full syntactic trees (1992)
- possibly the most-used corpus in NLP
- originally, just text from the Wall Street Journal (WSJ)

Universal POS tags

- Simplify the set of tags to lowest common denominator across languages
- Map existing annotations onto universal tags

$$\text{VBD, VBN, VB, VBG, VBP} \rightarrow \text{VERB}$$

- Allows interoperability of systems across languages
- Promoted by Google and others

Universal POS tags

NOUN (nouns)
VERB (verbs)
ADJ (adjectives)
ADV (adverbs)
PRON (pronouns)
DET (determiners and articles)
ADP (prepositions and postpositions)
NUM (numerals)
CONJ (conjunctions)
PRT (particles)
?.? (punctuation marks)
X (anything else, such as abbreviations or foreign words)

Example of POS tagged data

The/DT grand/JJ jury/NN commented/VBD on/IN a/DT number/NN of/IN other/JJ topics/NNS ./.

There/EX was/VBD still/JJ lemonade/NN in/IN the/DT bottle/NN ./.

NLMI

Parts of speech

Hidden Markov Models

Evaluation

How does any of that help modelling language?

Linguistic generalisation abstracts away from surface form

- knowing X_i took on an adjective should increase the chance that X_{i+1} takes on a noun
  - regardless of the adjective and of the noun

Role of conditional independence

Suppose A and B take on values in $\{1, \ldots, n\}$ and $\{1, \ldots, m\}$

- how many parameters to represent P_{AB}?

Role of conditional independence

Suppose A and B take on values in $\{1, \ldots, n\}$ and $\{1, \ldots, m\}$

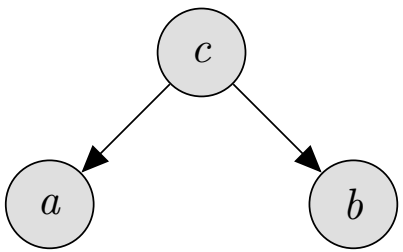
- how many parameters to represent P_{AB}? $O(n \times m)$

Role of conditional independence

Suppose A and B take on values in $\{1, \ldots, n\}$ and $\{1, \ldots, m\}$

- how many parameters to represent P_{AB}? $O(n \times m)$

We can make A and B conditionally independent given C



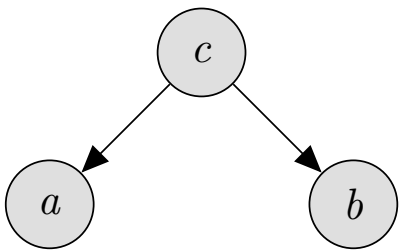
$$
\begin{aligned}
P_{AB}(a, b) &= \sum_{c=1}^{t} P_{ABC}(a, b, c) \\
&= \sum_{c=1}^{t} P_C(c) P_{AB|C}(a, b|c) \\
&= \sum_{c=1}^{t} P_C(c) P_{A|C}(a|c) P_{B|C}(b|c)
\end{aligned}
$$

Role of conditional independence

Suppose A and B take on values in $\{1, \ldots, n\}$ and $\{1, \ldots, m\}$

- how many parameters to represent P_{AB}? $O(n \times m)$

We can make A and B conditionally independent given C



$$
\begin{aligned}
P_{AB}(a, b) &= \sum_{c=1}^{t} P_{ABC}(a, b, c) \\
&= \sum_{c=1}^{t} P_C(c) P_{AB|C}(a, b|c) \\
&= \sum_{c=1}^{t} P_C(c) P_{A|C}(a|c) P_{B|C}(b|c)
\end{aligned}
$$

and still *marginally dependent*

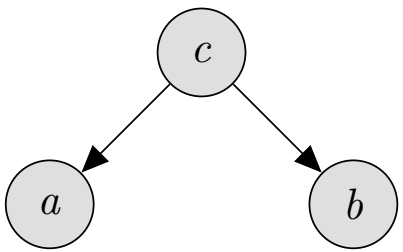
- with how many parameters?

Role of conditional independence

Suppose A and B take on values in $\{1, \ldots, n\}$ and $\{1, \ldots, m\}$

- how many parameters to represent P_{AB}? $O(n \times m)$

We can make A and B conditionally independent given C

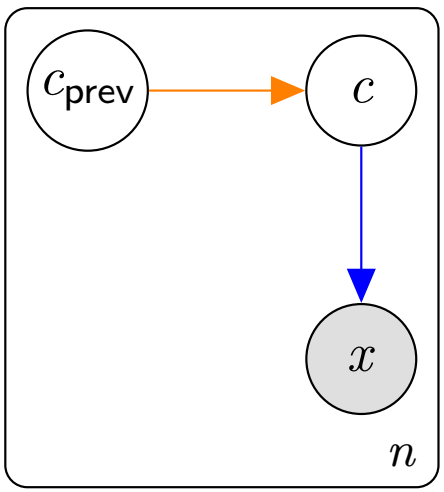


$$
\begin{aligned}
P_{AB}(a, b) &= \sum_{c=1}^{t} P_{ABC}(a, b, c) \\
&= \sum_{c=1}^{t} P_C(c) P_{AB|C}(a, b|c) \\
&= \sum_{c=1}^{t} P_C(c) P_{A|C}(a|c) P_{B|C}(b|c)
\end{aligned}
$$

and still *marginally dependent*

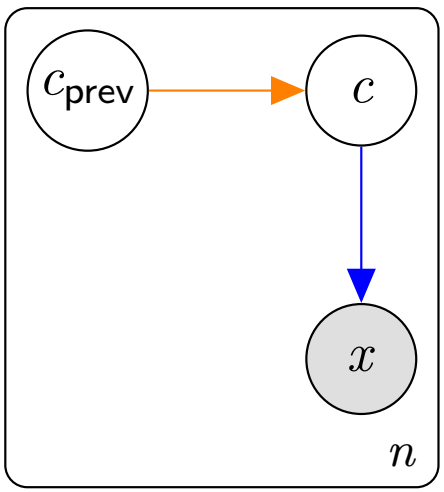
- with how many parameters? $O(t + t \times n + t \times m)$

Hidden Markov Model



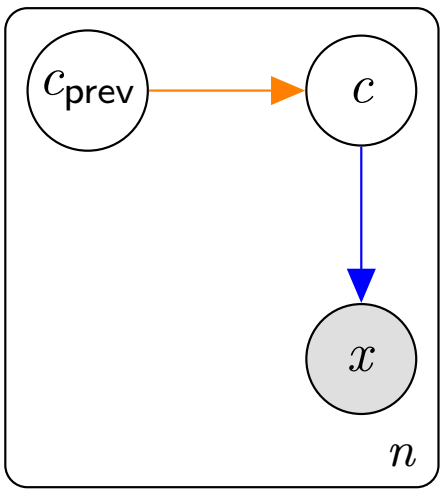
$$P_{CX|C_{\text{prev}}}(x, c|c_{\text{prev}}) = ?$$

Hidden Markov Model



$$P_{CX|C_{\text{prev}}}(x, c|c_{\text{prev}}) = P_{C|C_{\text{prev}}}(c|c_{\text{prev}})P_{X|C}(x|c)$$

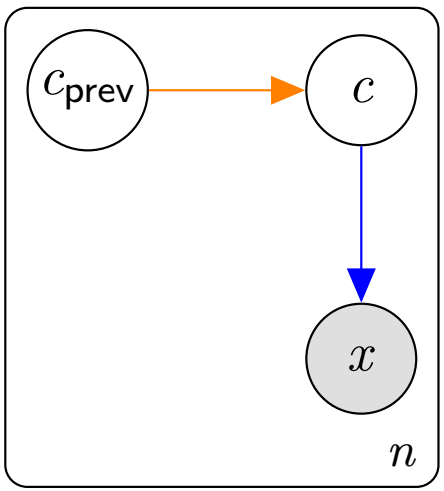
Hidden Markov Model



$$P_{CX|C_{\text{prev}}}(x, c|c_{\text{prev}}) = P_{C|C_{\text{prev}}}(c|c_{\text{prev}})P_{X|C}(x|c)$$

$$P_{X|C_{\text{prev}}}(x|c_{\text{prev}}) = ?$$

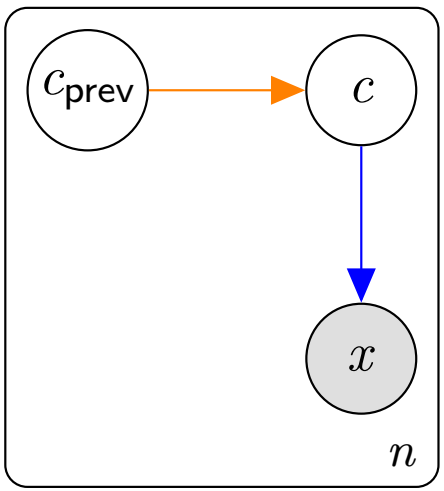
Hidden Markov Model



$$P_{CX|C_{\text{prev}}}(x, c|c_{\text{prev}}) = P_{C|C_{\text{prev}}}(c|c_{\text{prev}})P_{X|C}(x|c)$$

$$P_{X|C_{\text{prev}}}(x|c_{\text{prev}}) = \sum_{c=1}^{t} P_{CX|C_{\text{prev}}}(c, x|c_{\text{prev}})$$

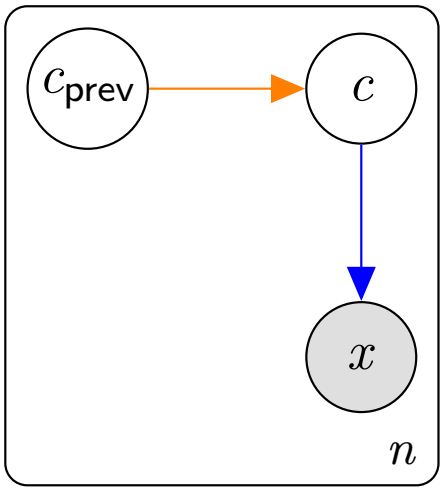
Hidden Markov Model



$$P_{CX|C_{\text{prev}}}(x, c|c_{\text{prev}}) = P_{C|C_{\text{prev}}}(c|c_{\text{prev}})P_{X|C}(x|c)$$

$$P_{X|C_{\text{prev}}}(x|c_{\text{prev}}) = \sum_{c=1}^{t} P_{C|C_{\text{prev}}}(c|c_{\text{prev}})P_{X|C}(x|c)$$

Hidden Markov Model

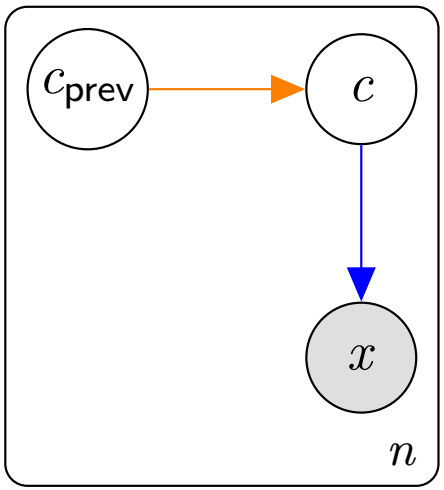


$$P_{CX|C_{\text{prev}}}(x, c|c_{\text{prev}}) = P_{C|C_{\text{prev}}}(c|c_{\text{prev}})P_{X|C}(x|c)$$

$$P_{X|C_{\text{prev}}}(x|c_{\text{prev}}) = \sum_{c=1}^{t} P_{C|C_{\text{prev}}}(c|c_{\text{prev}})P_{X|C}(x|c)$$

Now note that we have n independent terms

Hidden Markov Model



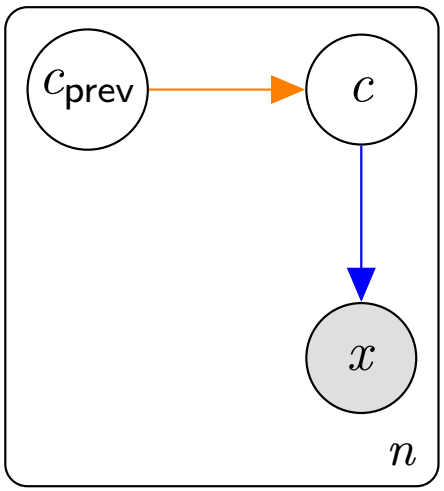
$$P_{CX|C_{\text{prev}}}(x, c|c_{\text{prev}}) = P_{C|C_{\text{prev}}}(c|c_{\text{prev}})P_{X|C}(x|c)$$

$$P_{X|C_{\text{prev}}}(x|c_{\text{prev}}) = \sum_{c=1}^{t} P_{C|C_{\text{prev}}}(c|c_{\text{prev}})P_{X|C}(x|c)$$

Now note that we have n independent terms

$$P_{X_1^n}(x_1^n) = ?$$

Hidden Markov Model



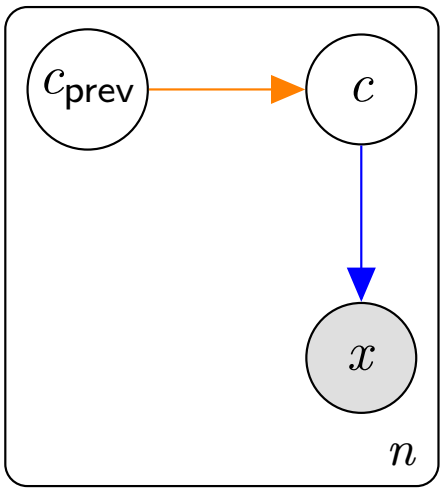
$$P_{CX|C_{\text{prev}}}(x, c|c_{\text{prev}}) = P_{C|C_{\text{prev}}}(c|c_{\text{prev}})P_{X|C}(x|c)$$

$$P_{X|C_{\text{prev}}}(x|c_{\text{prev}}) = \sum_{c=1}^{t} P_{C|C_{\text{prev}}}(c|c_{\text{prev}})P_{X|C}(x|c)$$

Now note that we have n independent terms

$$P_{X_1^n}(x_1^n) = \prod_{i=1}^{n} \sum_{c_{i-1}=1}^{t} P_{X|C_{\text{prev}}}(x_i|c_{i-1})$$

Hidden Markov Model



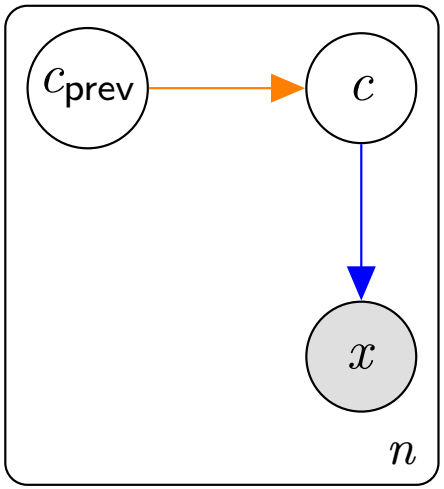
$$P_{CX|C_{\text{prev}}}(x, c|c_{\text{prev}}) = P_{C|C_{\text{prev}}}(c|c_{\text{prev}})P_{X|C}(x|c)$$

$$P_{X|C_{\text{prev}}}(x|c_{\text{prev}}) = \sum_{c=1}^{t} P_{C|C_{\text{prev}}}(c|c_{\text{prev}})P_{X|C}(x|c)$$

Now note that we have n independent terms

$$P_{X_1^n}(x_1^n) = \prod_{i=1}^{n} \sum_{c_{i-1}=1}^{t} \underbrace{\sum_{c_i=1}^{t} P_{C|C_{\text{prev}}}(c_i|c_{i-1})P_{X|C}(x_i|c_i)}_{P_{X|C_{\text{prev}}}(x_i|c_{i-1})}$$

Hidden Markov Model



$$P_{CX|C_{\text{prev}}}(x, c|c_{\text{prev}}) = P_{C|C_{\text{prev}}}(c|c_{\text{prev}})P_{X|C}(x|c)$$

$$P_{X|C_{\text{prev}}}(x|c_{\text{prev}}) = \sum_{c=1}^{t} P_{C|C_{\text{prev}}}(c|c_{\text{prev}})P_{X|C}(x|c)$$

Now note that we have n independent terms

$$P_{X_1^n}(x_1^n) = \prod_{i=1}^{n} \sum_{c_{i-1}=1}^{t} \underbrace{\sum_{c_i=1}^{t} P_{C|C_{\text{prev}}}(c_i|c_{i-1})P_{X|C}(x_i|c_i)}_{P_{X|C_{\text{prev}}}(x_i|c_{i-1})}$$

$$= \prod_{i=1}^{n} \underbrace{\sum_{c_i=1}^{t} P_{X|C}(x_i|c_i) \sum_{c_{i-1}=1}^{t} P_{C|C_{\text{prev}}}(c_i|c_{i-1})}_{P_X(x_i)}$$

Modelling POS-tagged data: illustration

Joint observations

the/*DET* book/*NOUN* is/*VERB* on/*ADP* the/*DET* table/*NOUN* ./*PUNC*

Generative story

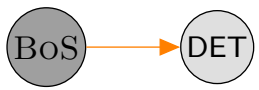
We use pad the tag sequence with a BoS symbol

Modelling POS-tagged data: illustration

Joint observations

the/*DET* book/*NOUN* is/*VERB* on/*ADP* the/*DET* table/*NOUN* ./*PUNC*

Generative story



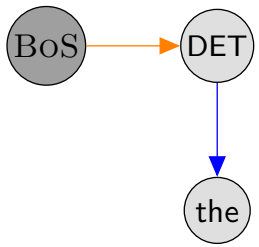
We use pad the tag sequence with a BoS symbol

Modelling POS-tagged data: illustration

Joint observations

the/*DET* book/*NOUN* is/*VERB* on/*ADP* the/*DET* table/*NOUN* ./*PUNC*

Generative story



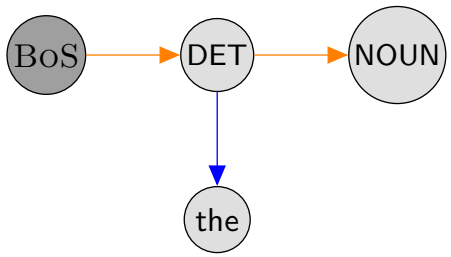
We use pad the tag sequence with a BoS symbol

Modelling POS-tagged data: illustration

Joint observations

the/*DET* book/*NOUN* is/*VERB* on/*ADP* the/*DET* table/*NOUN* ./*PUNC*

Generative story



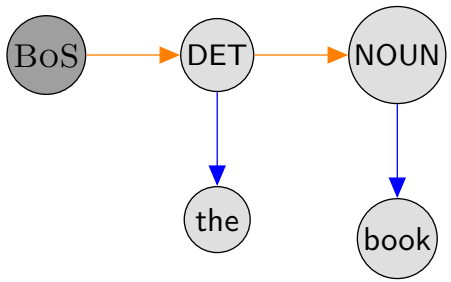
We use pad the tag sequence with a BoS symbol

Modelling POS-tagged data: illustration

Joint observations

the/*DET* book/*NOUN* is/*VERB* on/*ADP* the/*DET* table/*NOUN* ./*PUNC*

Generative story



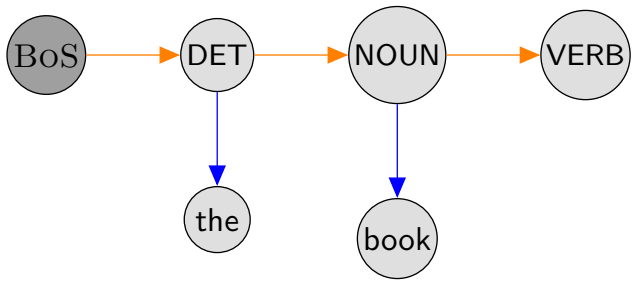
We use pad the tag sequence with a BoS symbol

Modelling POS-tagged data: illustration

Joint observations

the/*DET* book/*NOUN* is/*VERB* on/*ADP* the/*DET* table/*NOUN* ./*PUNC*

Generative story



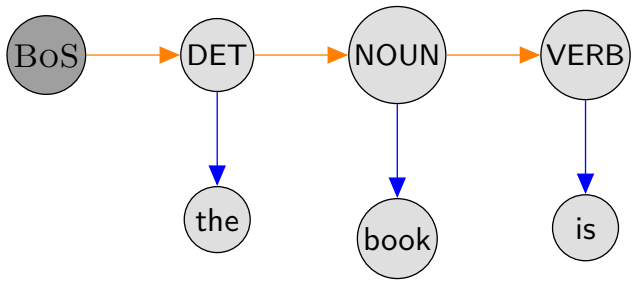
We use pad the tag sequence with a BoS symbol

Modelling POS-tagged data: illustration

Joint observations

the/*DET* book/*NOUN* is/*VERB* on/*ADP* the/*DET* table/*NOUN* ./*PUNC*

Generative story



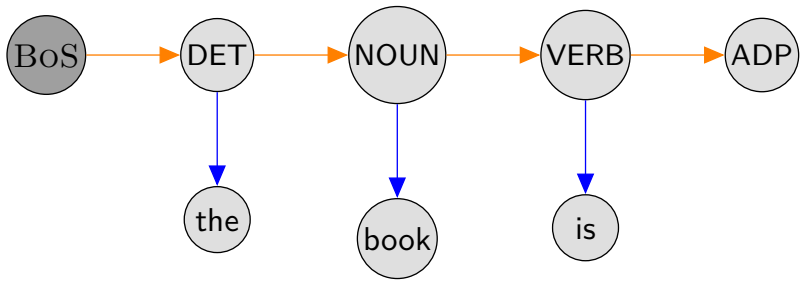
We use pad the tag sequence with a BoS symbol

Modelling POS-tagged data: illustration

Joint observations

the/*DET* book/*NOUN* is/*VERB* on/*ADP* the/*DET* table/*NOUN* ./*PUNC*

Generative story



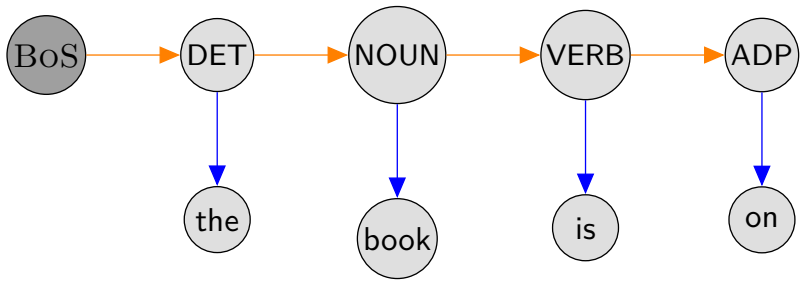
We use pad the tag sequence with a BoS symbol

Modelling POS-tagged data: illustration

Joint observations

the/*DET* book/*NOUN* is/*VERB* on/*ADP* the/*DET* table/*NOUN* ./*PUNC*

Generative story



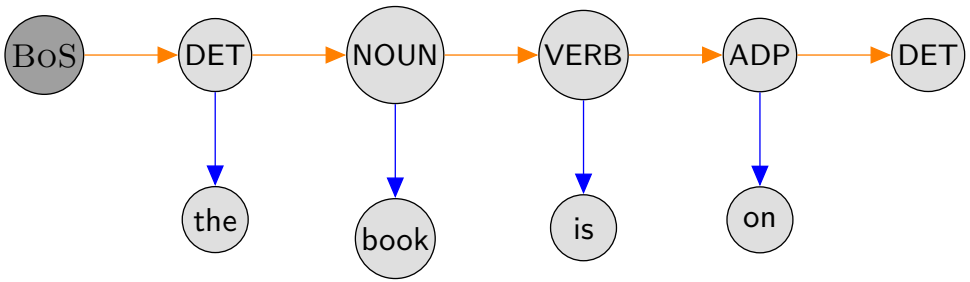
We use pad the tag sequence with a BoS symbol

Modelling POS-tagged data: illustration

Joint observations

the/*DET* book/*NOUN* is/*VERB* on/*ADP* the/*DET* table/*NOUN* ./*PUNC*

Generative story



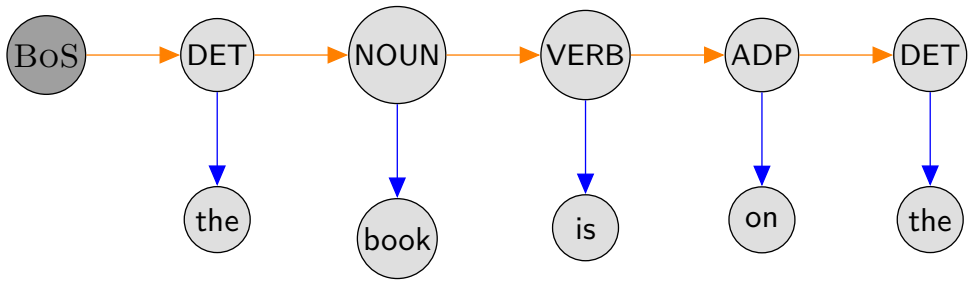
We use pad the tag sequence with a BoS symbol

Modelling POS-tagged data: illustration

Joint observations

the/*DET* book/*NOUN* is/*VERB* on/*ADP* the/*DET* table/*NOUN* ./*PUNC*

Generative story



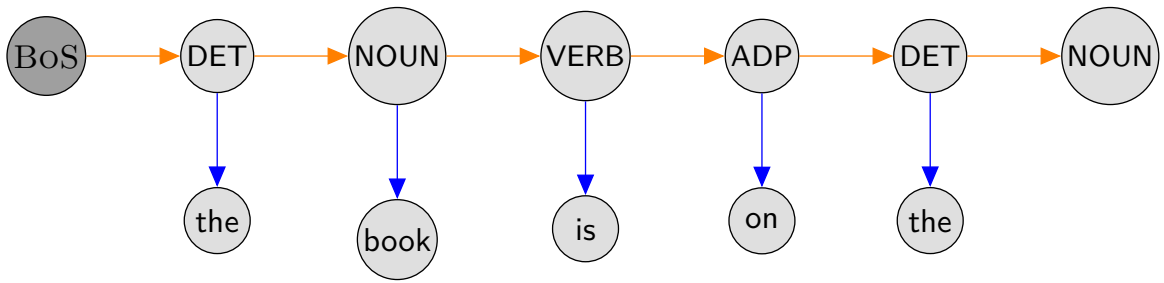
We use pad the tag sequence with a BoS symbol

Modelling POS-tagged data: illustration

Joint observations

the/*DET* book/*NOUN* is/*VERB* on/*ADP* the/*DET* table/*NOUN* ./*PUNC*

Generative story



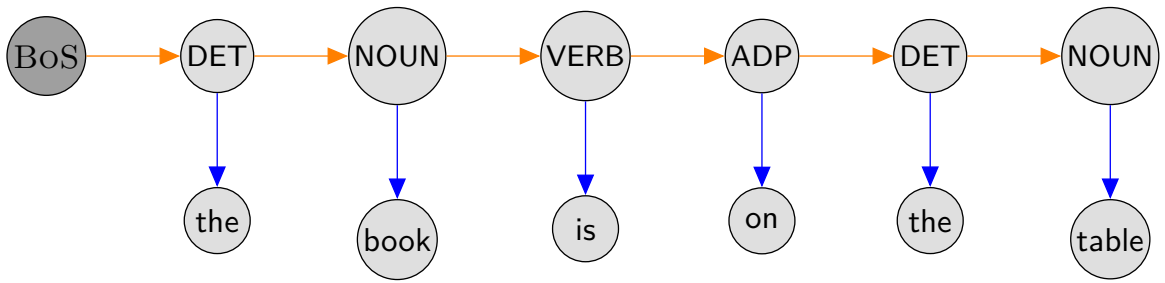
We use pad the tag sequence with a BoS symbol

Modelling POS-tagged data: illustration

Joint observations

the/*DET* book/*NOUN* is/*VERB* on/*ADP* the/*DET* table/*NOUN* ./*PUNC*

Generative story



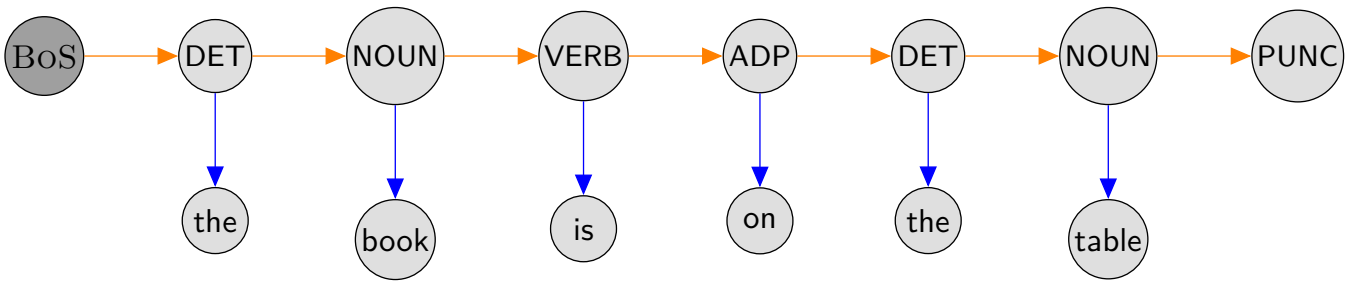
We use pad the tag sequence with a BoS symbol

Modelling POS-tagged data: illustration

Joint observations

the/*DET* book/*NOUN* is/*VERB* on/*ADP* the/*DET* table/*NOUN* ./*PUNC*

Generative story



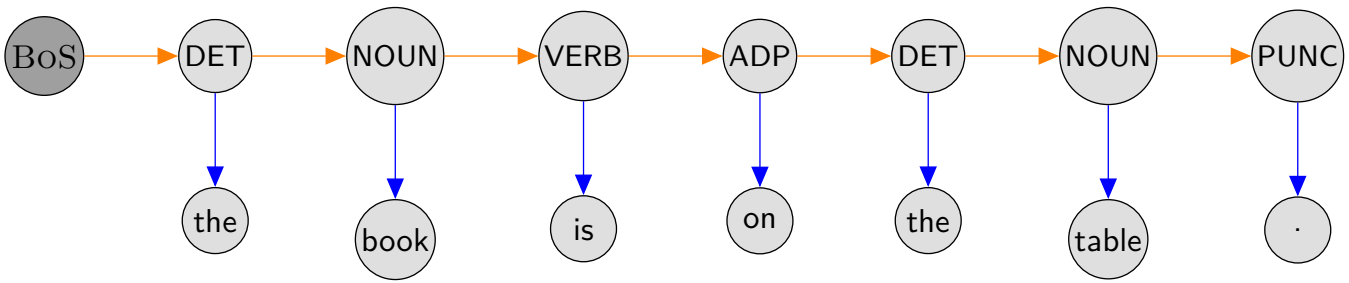
We use pad the tag sequence with a BoS symbol

Modelling POS-tagged data: illustration

Joint observations

the/*DET* book/*NOUN* is/*VERB* on/*ADP* the/*DET* table/*NOUN* ./*PUNC*

Generative story



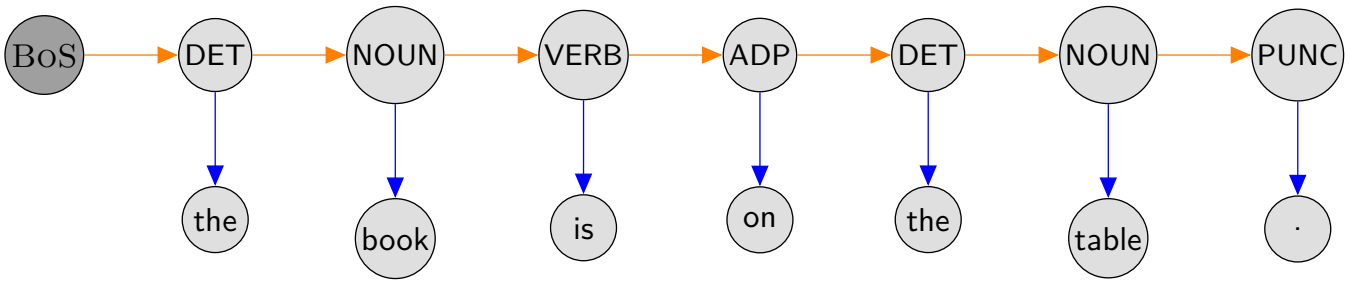
We use pad the tag sequence with a BoS symbol

Modelling POS-tagged data: illustration

Joint observations

the/*DET* book/*NOUN* is/*VERB* on/*ADP* the/*DET* table/*NOUN* ./*PUNC*

Generative story



Joint probability

$$
\begin{aligned}
& P_{C|C_{\text{prev}}}(\text{DET}|\text{BoS}) P_{X|C}(\text{the}|\text{DET}) \\
& \times P_{C|C_{\text{prev}}}(\text{NOUN}|\text{DET}) P_{X|C}(\text{book}|\text{NOUN}) \\
& \times \ldots \\
& \times P_{C|C_{\text{prev}}}(\text{PUNC}|\text{NOUN}) P_{X|C}(.|\text{PUNC})
\end{aligned}
$$

We use pad the tag sequence with a BoS symbol

Modelling POS-tagged data

Random variables

- X is a random word taking on values in $\mathcal{X} = \{1, \ldots, v\}$
- C is a random tag taking on values in $\mathcal{C} = \{1, \ldots, t\}$

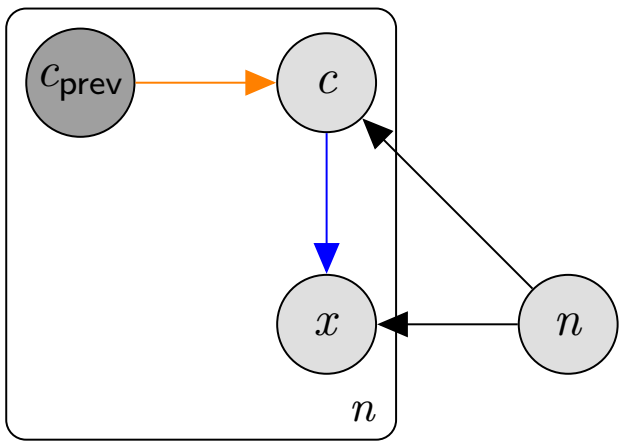
Modelling POS-tagged data

Random variables

- X is a random word taking on values in $\mathcal{X} = \{1, \ldots, v\}$
- C is a random tag taking on values in $\mathcal{C} = \{1, \ldots, t\}$

Generative story

1. $N \sim P_N$
2. For $i = 1, \ldots, n$
   - $C_i | c_{i-1} \sim P_{C|C_{\text{prev}}}$
   - $X_i | c_i \sim P_{X|C}$



Modelling POS-tagged data

Random variables

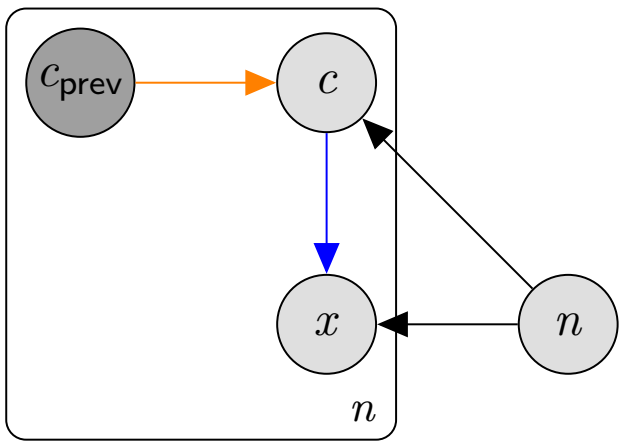
- X is a random word taking on values in $\mathcal{X} = \{1, \ldots, v\}$
- C is a random tag taking on values in $\mathcal{C} = \{1, \ldots, t\}$

Generative story

1. $N \sim P_N$
2. For $i = 1, \ldots, n$
   - $C_i | c_{i-1} \sim P_{C|C_{\text{prev}}}$
   - $X_i | c_i \sim P_{X|C}$

Parameterisation

- Transition distribution
  $C | C_{\text{prev}} = p \sim \text{Cat}(\lambda_1^{(p)}, \ldots, \lambda_t^{(p)})$
- Emission distribution
  $X | C = c \sim \text{Cat}(\theta_1^{(c)}, \ldots, \theta_v^{(c)})$



Modelling POS-tagged data

Random variables

- X is a random word taking on values in $\mathcal{X} = \{1, \ldots, v\}$
- C is a random tag taking on values in $\mathcal{C} = \{1, \ldots, t\}$

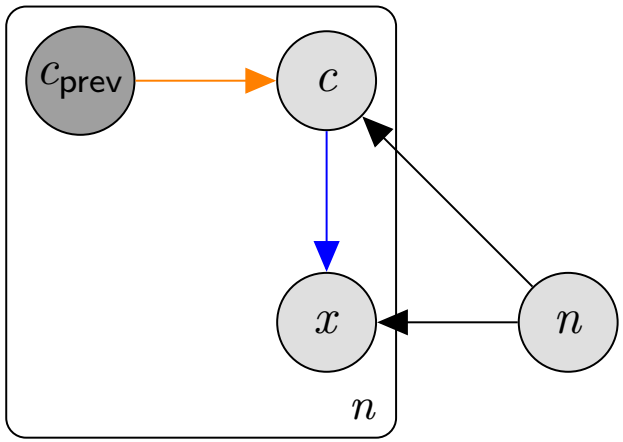
Generative story

1. $N \sim P_N$
2. For $i = 1, \ldots, n$
   - $C_i | c_{i-1} \sim P_{C|C_{\text{prev}}}$
   - $X_i | c_i \sim P_{X|C}$

Parameterisation

- Transition distribution
  $C | C_{\text{prev}} = p \sim \text{Cat}(\lambda_1^{(p)}, \ldots, \lambda_t^{(p)})$
- Emission distribution
  $X | C = c \sim \text{Cat}(\theta_1^{(c)}, \ldots, \theta_v^{(c)})$

How many parameters? $O(t^2 + tv)$



Modelling POS-tagged data

Random variables

- X is a random word taking on values in $\mathcal{X} = \{1, \ldots, v\}$
- C is a random tag taking on values in $\mathcal{C} = \{1, \ldots, t\}$

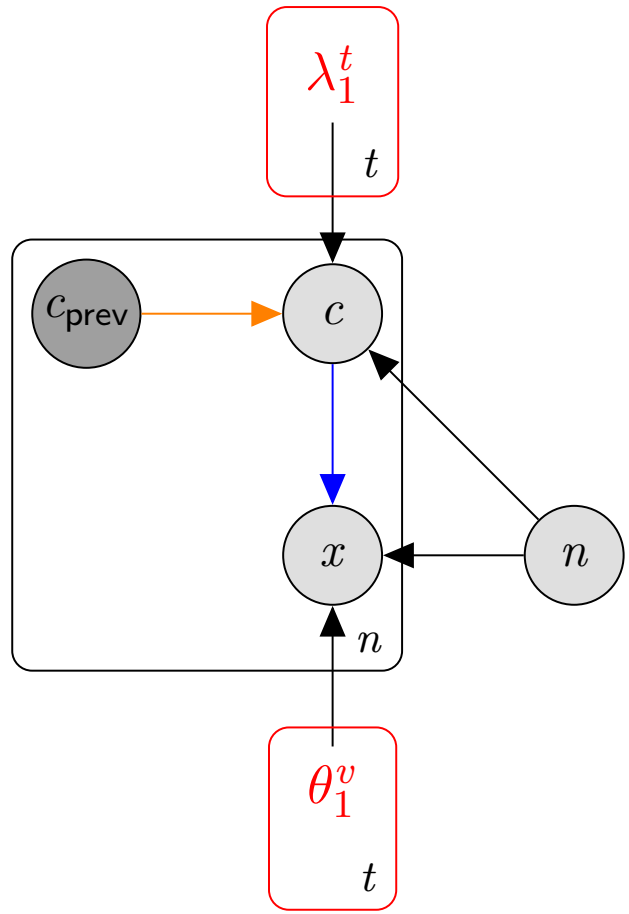
Generative story

1. $N \sim P_N$
2. For $i = 1, \ldots, n$
   - $C_i | c_{i-1} \sim P_{C|C_{\text{prev}}}$
   - $X_i | c_i \sim P_{X|C}$

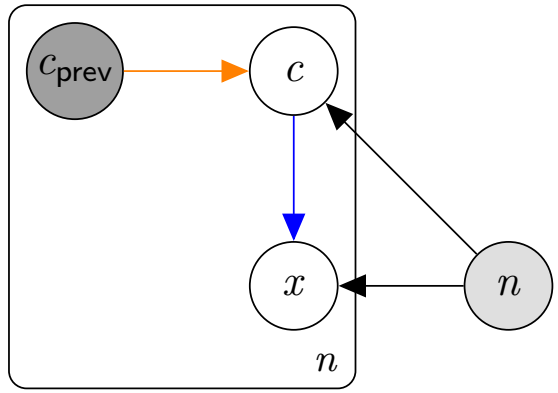
Parameterisation

- Transition distribution
  $C | C_{\text{prev}} = p \sim \text{Cat}(\lambda_1^{(p)}, \ldots, \lambda_t^{(p)})$
- Emission distribution
  $X | C = c \sim \text{Cat}(\theta_1^{(c)}, \ldots, \theta_v^{(c)})$

How many parameters? $O(t^2 + tv)$



Probability of a sentence



$$
\begin{aligned}
P_S(x_1^n) &= P_N(n) P_{X_1^n|N}(x_1^n|n) \\
&= P_N(n) \sum_{c_1=1}^{t} \cdots \sum_{c_n=1}^{t} P_{X_1^n C_1^n}(x_1^n, c_1^n|n) \\
&= P_N(n) \sum_{c_1=1}^{t} \cdots \sum_{c_n=1}^{t} \prod_{i=1}^{n} P_{XC|C_{\text{prev}}}(x_i, c_i|c_{i-1}) \\
&= P_N(n) \prod_{i=1}^{n} \sum_{c_{i-1}=1}^{t} \sum_{c_i=1}^{t} P_{XC|C_{\text{prev}}}(x_i, c_i|c_{i-1}) \\
&= P_N(n) \prod_{i=1}^{n} \sum_{c_{i-1}=1}^{t} \sum_{c_i=1}^{t} P_{C|C_{\text{prev}}}(c_i|c_{i-1}) P_{X|C}(x_i|c_i) \\
&= P_N(n) \prod_{i=1}^{n} \sum_{c_i=1}^{t} P_{X|C}(x_i|c_i) \sum_{c_{i-1}=1}^{t} P_{C|C_{\text{prev}}}(c_i|c_{i-1})
\end{aligned}
$$

Identities of summation

Maximum likelihood estimation for labelled data

Suppose a data set of m observations

$$\left(\langle x_1^{(k)}, \ldots, x_{n_k}^{(k)} \rangle, \langle c_1^{(k)}, \ldots, c_{n_k}^{(k)} \rangle\right)_{k=1}^{m}$$

MLE solution

- Transition distribution

$$\lambda_c^{(p)} = \frac{\sum_{k=1}^{m} \sum_{i=1}^{n_k} [p = c_{i-1}^{(k)} \wedge c = c_i^{(k)}]}{\sum_{k=1}^{m} \sum_{i=1}^{n_k} [p = c_{i-1}]} = \frac{\text{count}_{C_{\text{prev}}C}(p, c)}{\text{count}_{C_{\text{prev}}}(p)}$$

- Emission distribution

$$\theta_x^{(c)} = \frac{\sum_{k=1}^{m} \sum_{i=1}^{n_k} [c = c_i^{(k)} \wedge x = x_i^{(k)}]}{\sum_{k=1}^{m} \sum_{i=1}^{n_k} [c = c_i]} = \frac{\text{count}_{CX}(c, x)}{\text{count}_{C}(c)}$$

NLMI

Parts of speech

Hidden Markov Models

Evaluation

Evaluate our HMM language model

Intrinsically *no need for POS tag sequences*

- test set perplexity
- perplexity requires computing $P_{S|n}(x_1^n|n)$ by marginalising over tag sequences
- what's the complexity?

Evaluate our HMM language model

Intrinsically

no need for POS tag sequences

- test set perplexity
- perplexity requires computing $P_{S|n}(x_1^n|n)$ by marginalising over tag sequences
- what's the complexity?

$$P_S(x_1^n) = P_N(n) \underbrace{\prod_{i=1}^{n} \underbrace{\sum_{c_i=1}^{t} P_{X|C}(x_i|c_i) \sum_{c_{i-1}=1}^{t} P_{C|C_{\text{prev}}}(c_i|c_{i-1})}}$$

Evaluate our HMM language model

Intrinsically *no need for POS tag sequences*

- test set perplexity
- perplexity requires computing $P_{S|n}(x_1^n|n)$ by marginalising over tag sequences
- what's the complexity?

$$P_S(x_1^n) = P_N(n) \underbrace{\prod_{i=1}^{n} \underbrace{\sum_{c_i=1}^{t} P_{X|C}(x_i|c_i) \sum_{c_{i-1}=1}^{t} P_{C|C_{\text{prev}}}(c_i|c_{i-1})}_{O(t^2)}}$$

Evaluate our HMM language model

Intrinsically *no need for POS tag sequences*

- test set perplexity
- perplexity requires computing $P_{S|n}(x_1^n|n)$ by marginalising over tag sequences
- what's the complexity?

$$P_S(x_1^n) = P_N(n) \underbrace{\prod_{i=1}^{n} \underbrace{\sum_{c_i=1}^{t} P_{X|C}(x_i|c_i) \sum_{c_{i-1}=1}^{t} P_{C|C_{\text{prev}}}(c_i|c_{i-1})}_{O(t^2)}}_{O(nt^2)}$$

Evaluate our HMM language model

Extrinsically *given labelled test set*

- compare best possible tag sequence to tagged test set
- accuracy of tag prediction

Best tag sequence

Given a sentence, we want the most likely tag sequence

$$\begin{aligned}
&\operatorname*{argmax}_{c_1^n} P(c_1^n|x_1^n) && \text{posterior}\\
&= \operatorname*{argmax}_{c_1^n} \frac{P(x_1^n, c_1^n)}{P(x_1^n)} && \text{Bayes rule}\\
&= \operatorname*{argmax}_{c_1^n} P(x_1^n, c_1^n) && \text{proportionality}\\
&= \operatorname*{argmax}_{c_1^n} \prod_{i=1}^{n} P_{C|C_{\text{prev}}}(c_i|c_{i-1}) P_{X|C}(x_i|c_i)\\
&= \operatorname*{argmax}_{c_1^n} \prod_{i=1}^{n} \lambda_{c_i}^{(c_{i-1})} \theta_{x_i}^{(c_i)} && \text{Categorical pmf}\\
&= \operatorname*{argmax}_{c_1^n} \sum_{i=1}^{n} \log \lambda_{c_i}^{(c_{i-1})} + \log \theta_{x_i}^{(c_i)} && \text{monotonicity}
\end{aligned}$$

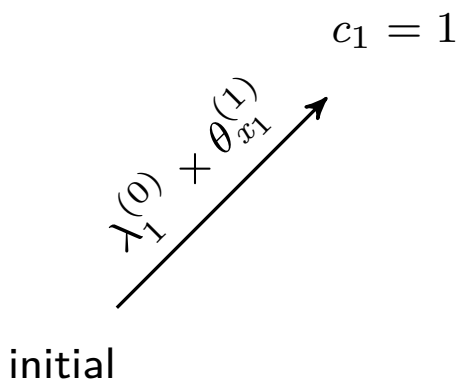
Space of analyses

Example: observation x_1^3 tagset $\{1, 2\}$

initial

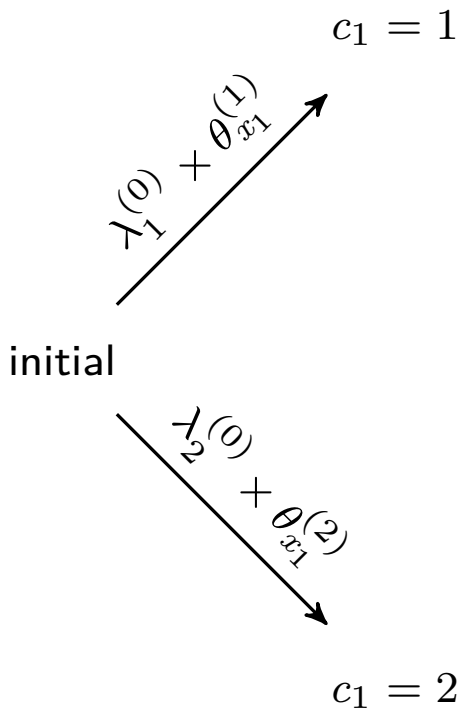
Space of analyses

Example: observation x_1^3 tagset $\{1, 2\}$



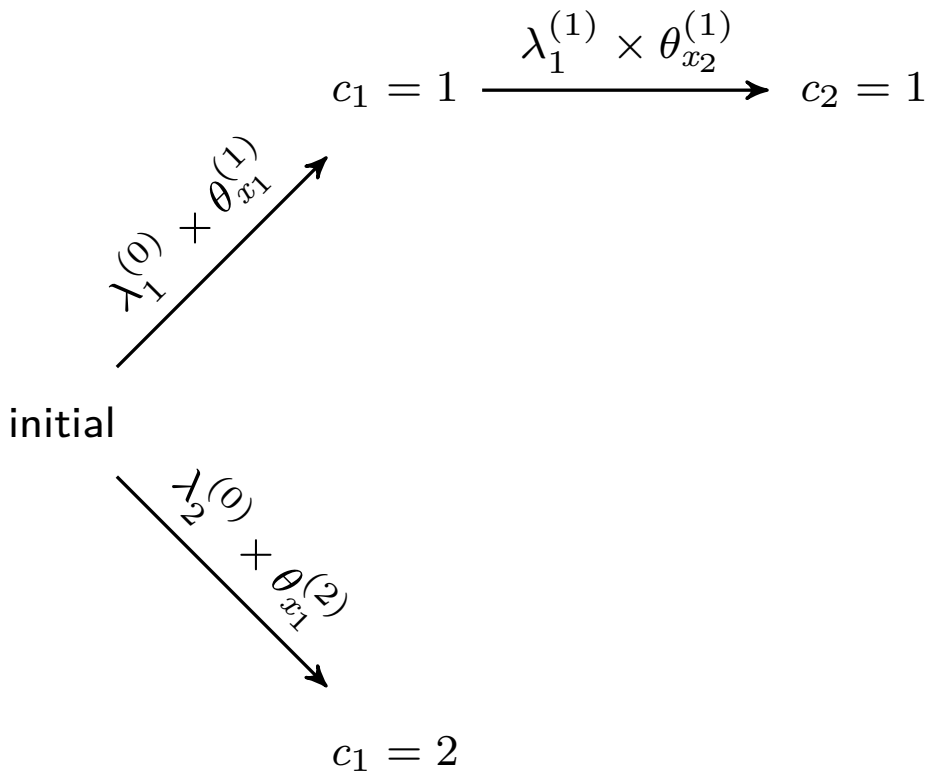
Space of analyses

Example: observation x_1^3 tagset $\{1, 2\}$



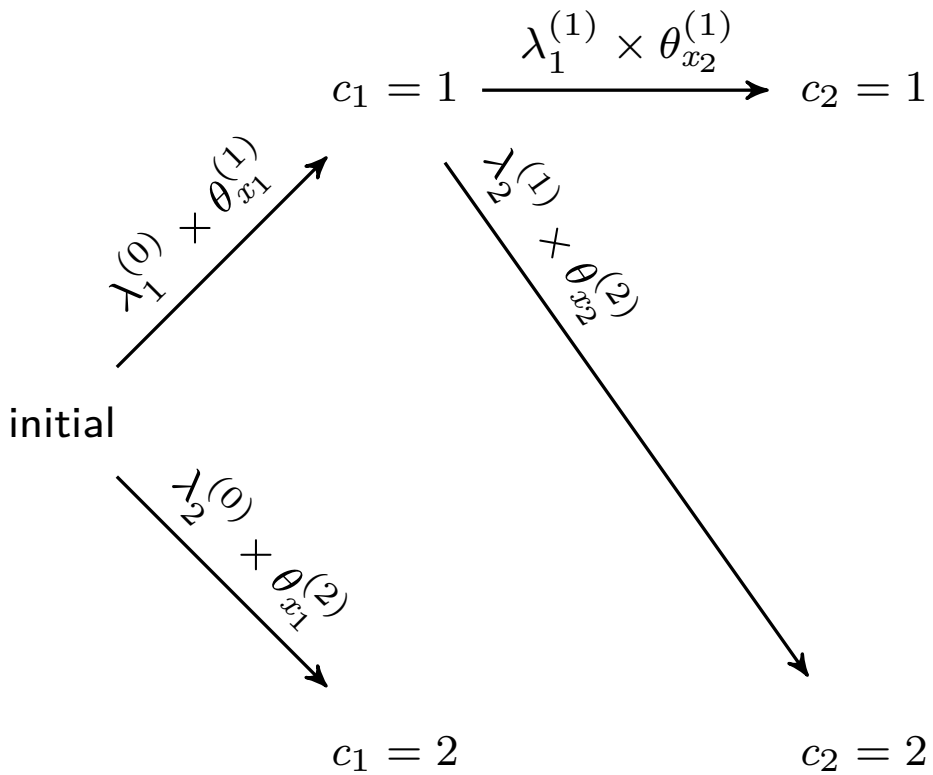
Space of analyses

Example: observation x_1^3 tagset $\{1, 2\}$



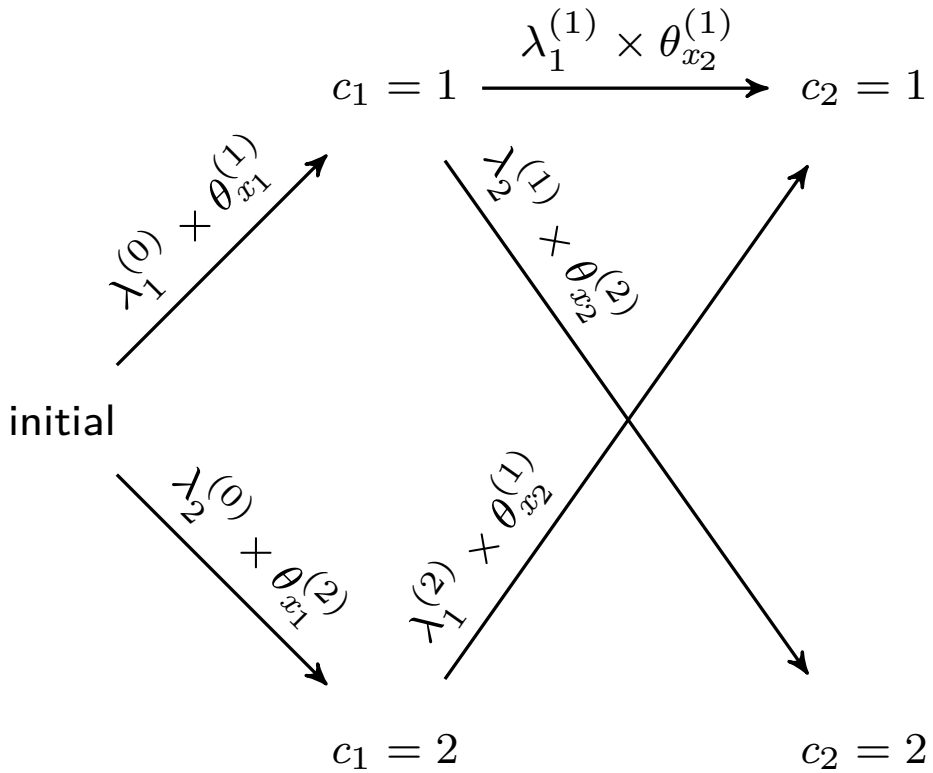
Space of analyses

Example: observation x_1^3 tagset $\{1, 2\}$



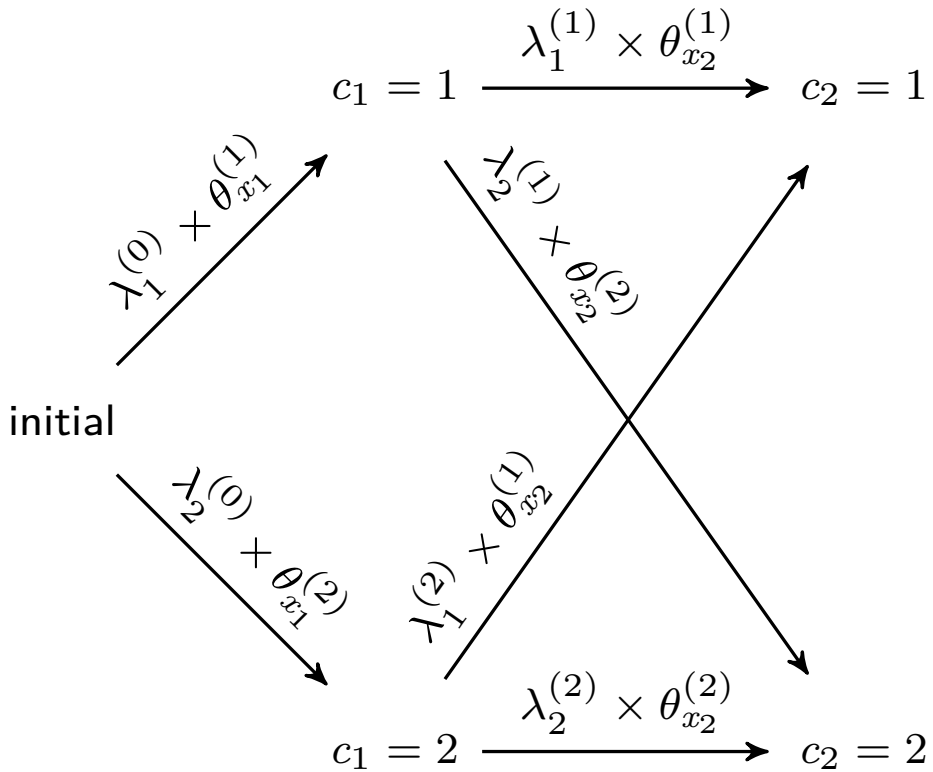
Space of analyses

Example: observation x_1^3 tagset $\{1, 2\}$



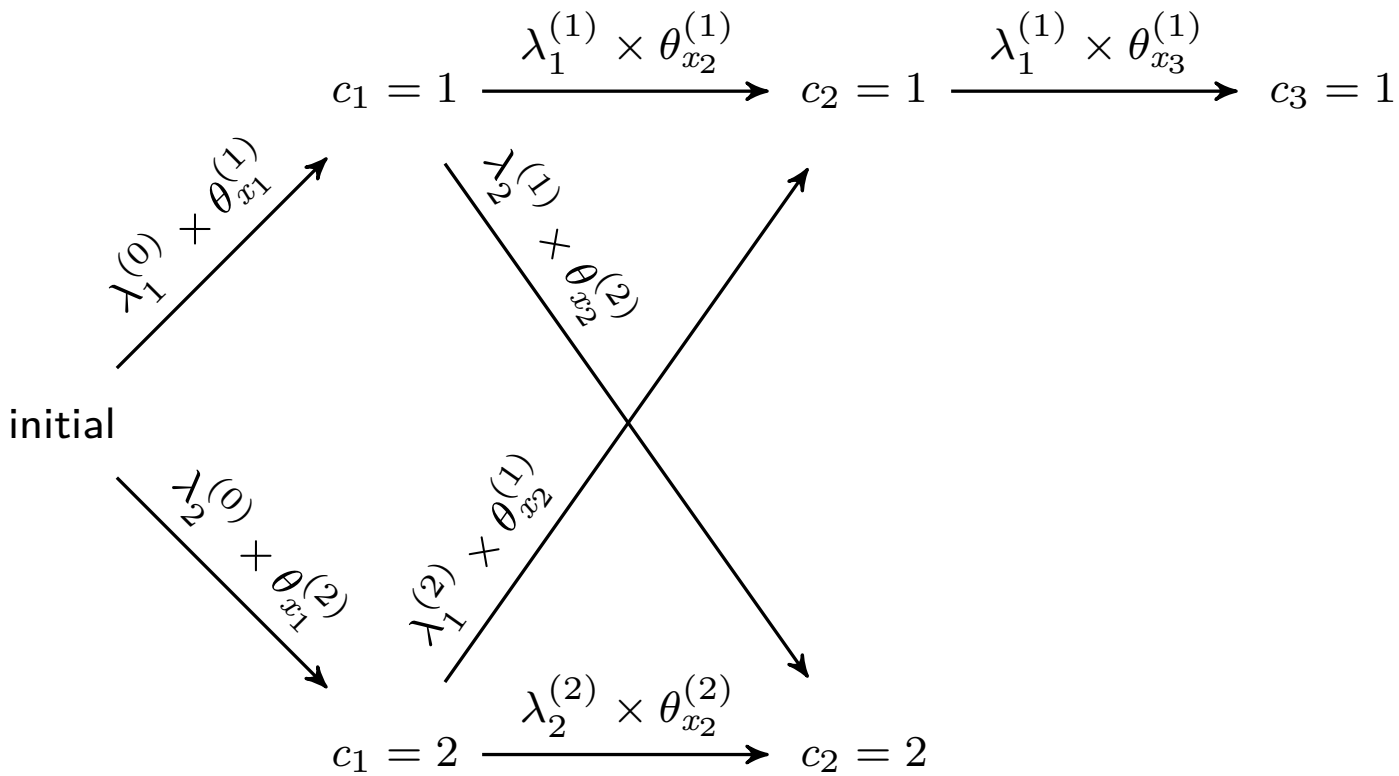
Space of analyses

Example: observation x_1^3 tagset $\{1, 2\}$



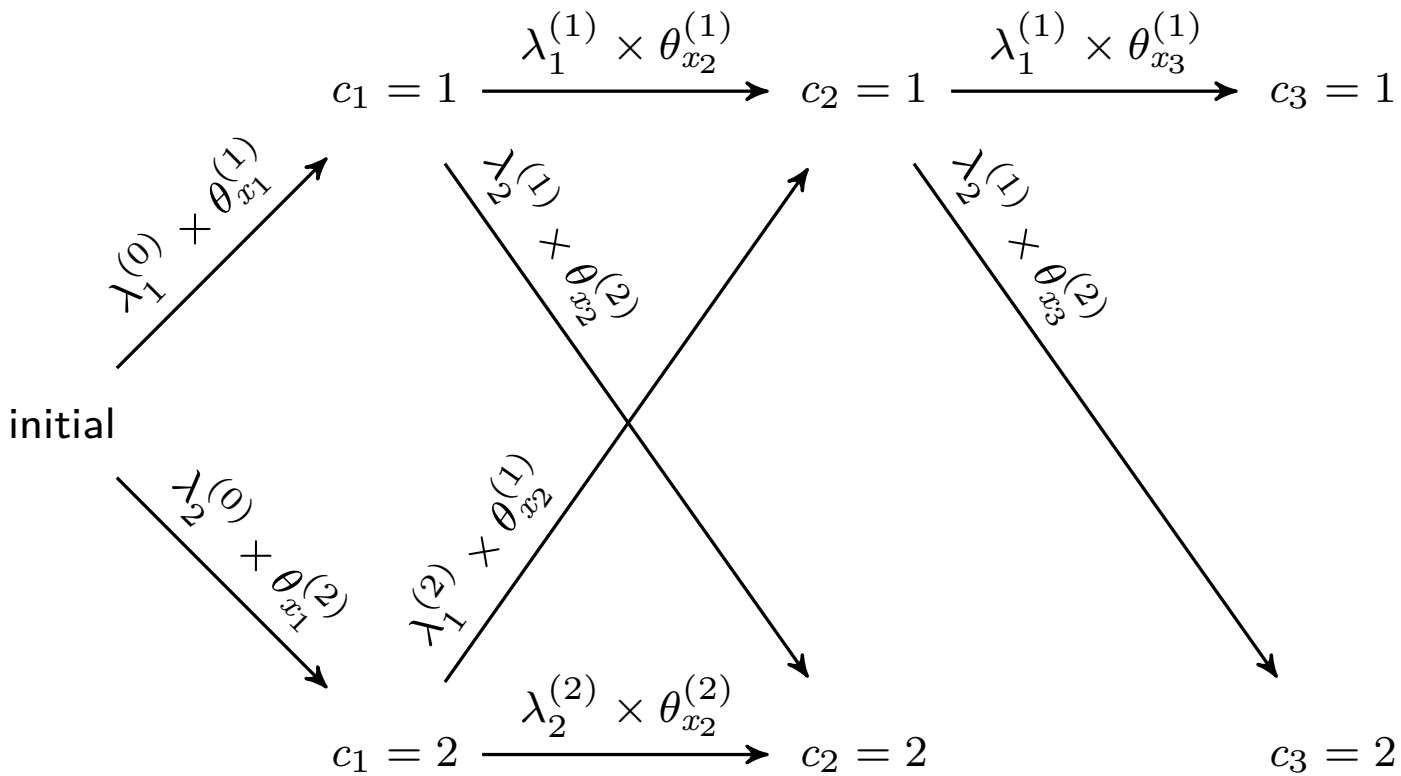
Space of analyses

Example: observation x_1^3 tagset $\{1, 2\}$



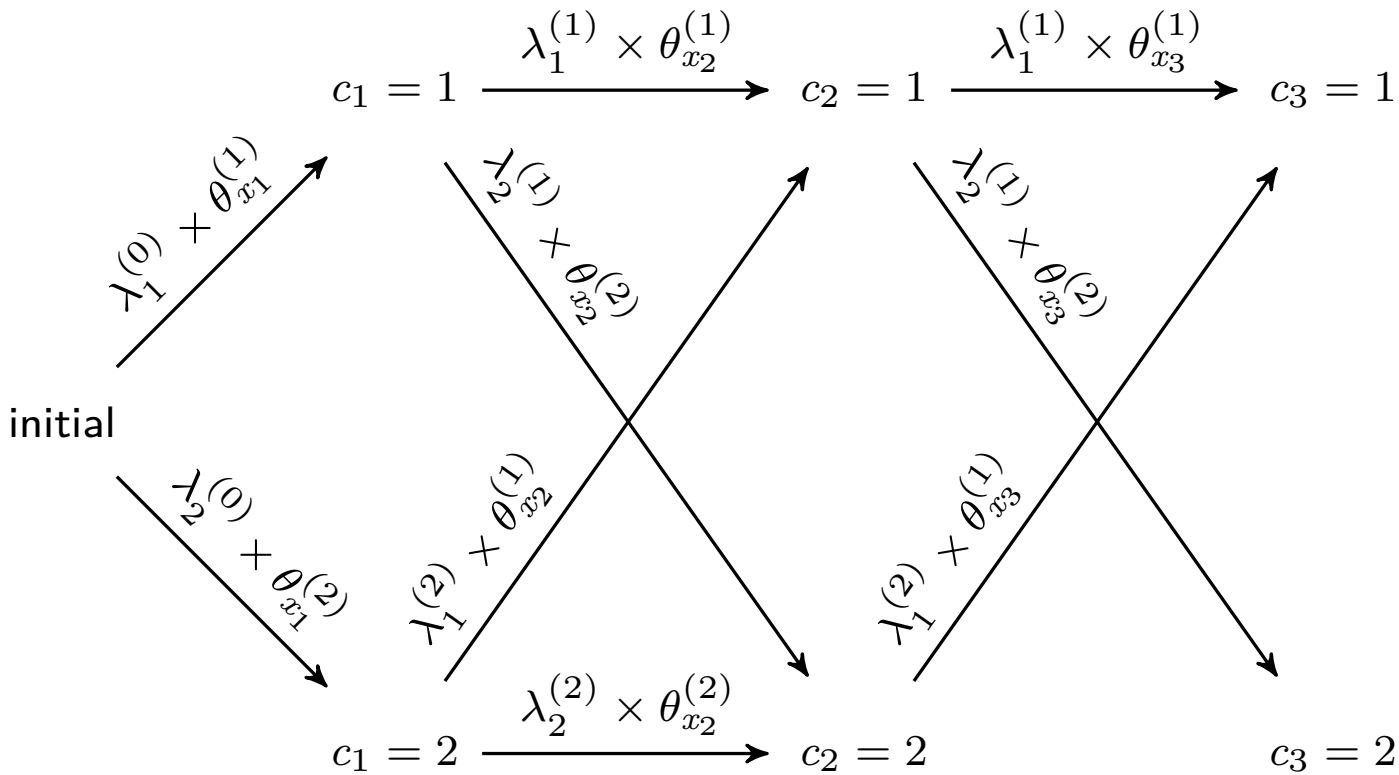
Space of analyses

Example: observation x_1^3 tagset $\{1, 2\}$



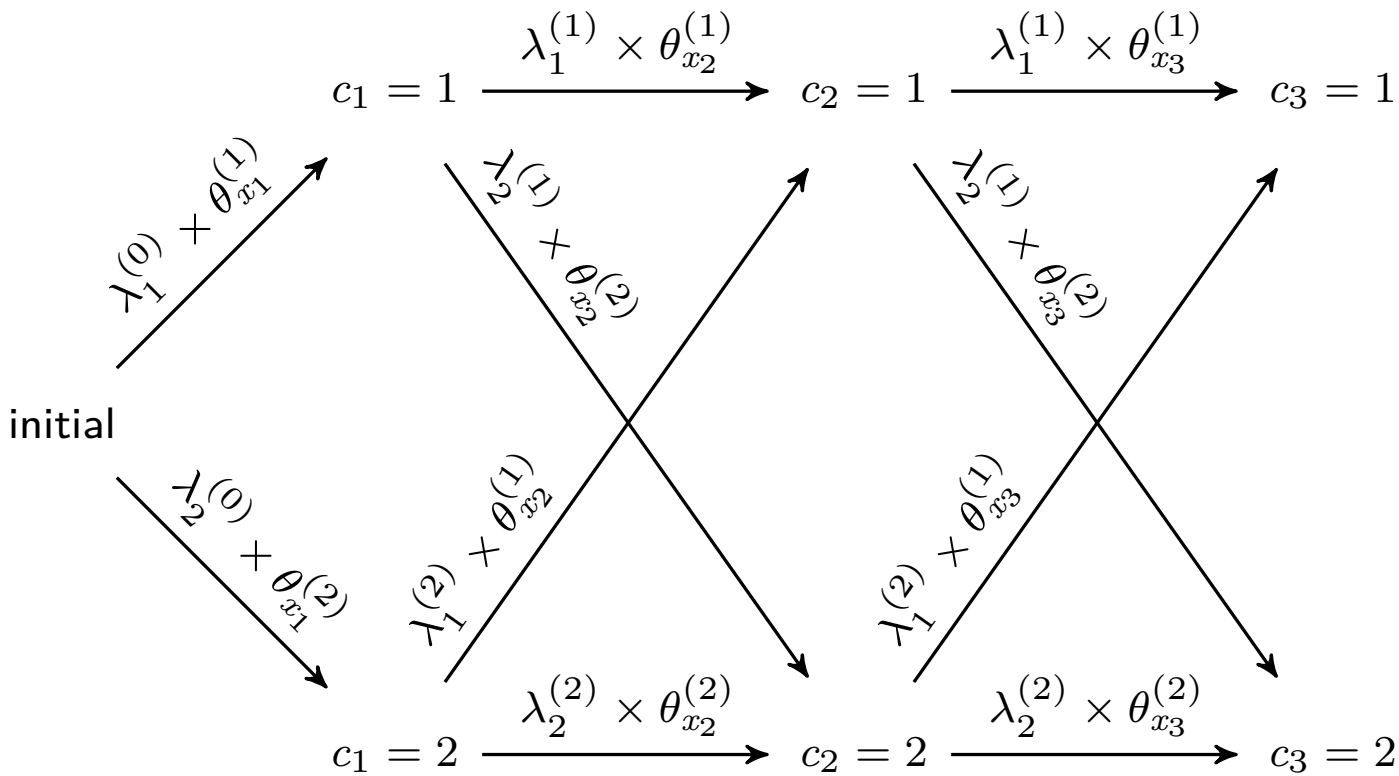
Space of analyses

Example: observation x_1^3 tagset $\{1, 2\}$



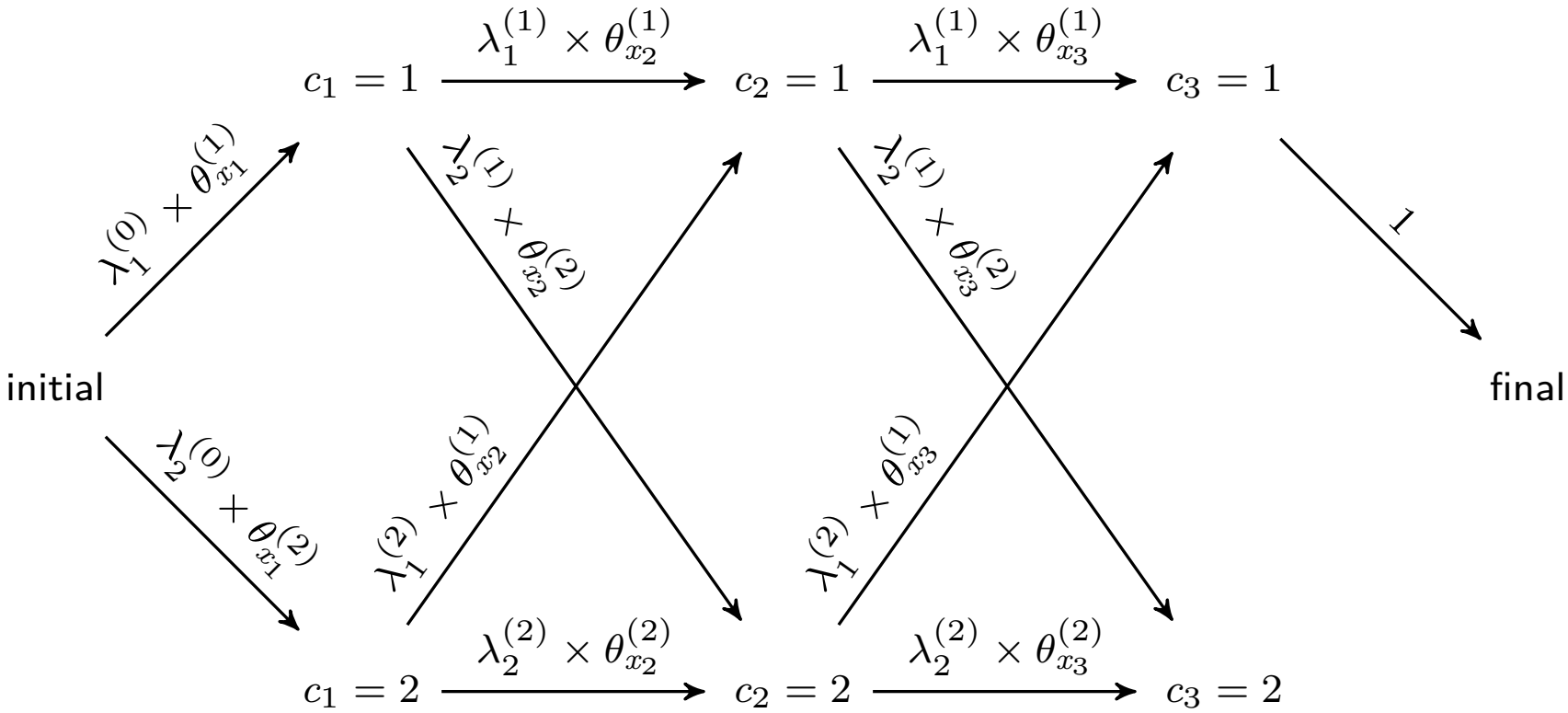
Space of analyses

Example: observation x_1^3 tagset $\{1, 2\}$



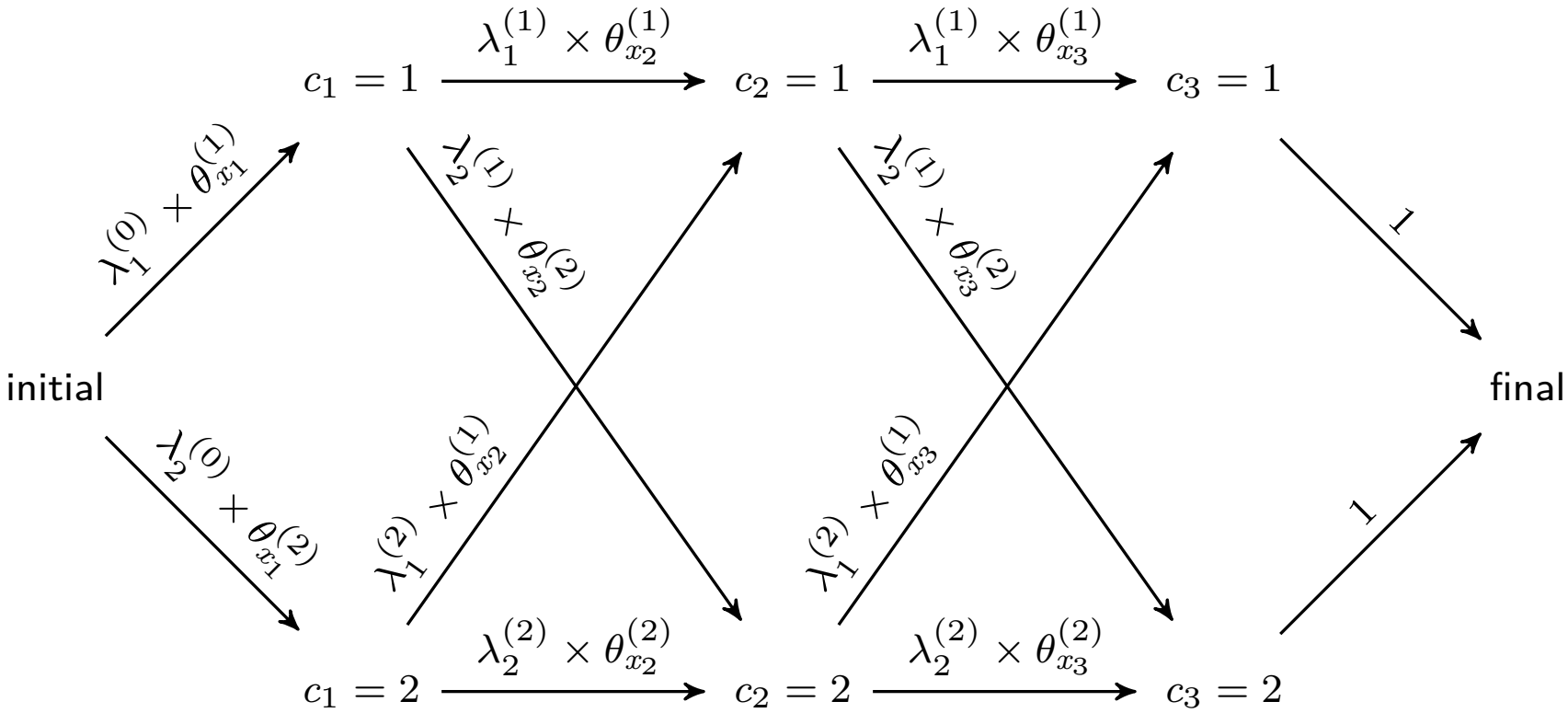
Space of analyses

Example: observation x_1^3 tagset $\{1, 2\}$



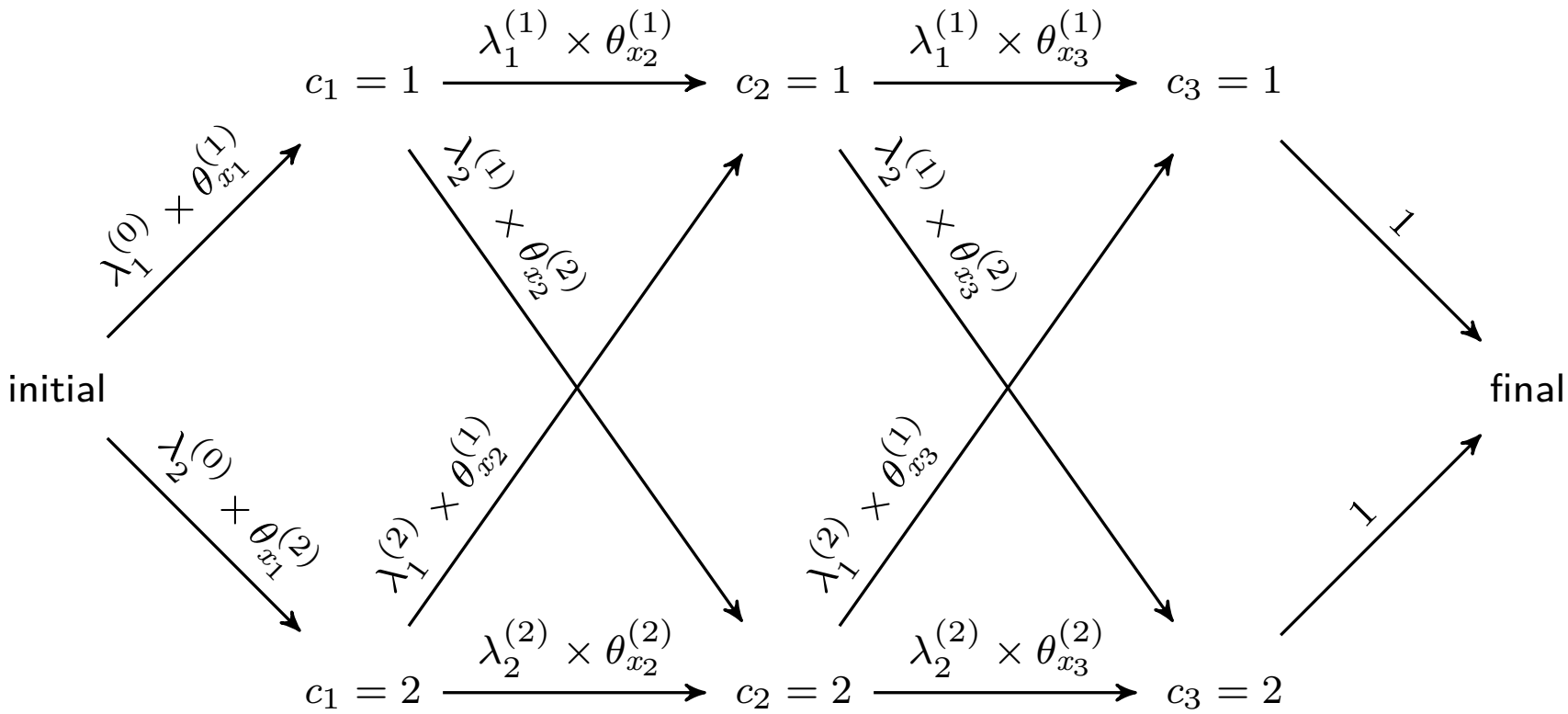
Space of analyses

Example: observation x_1^3 tagset $\{1, 2\}$



Space of analyses

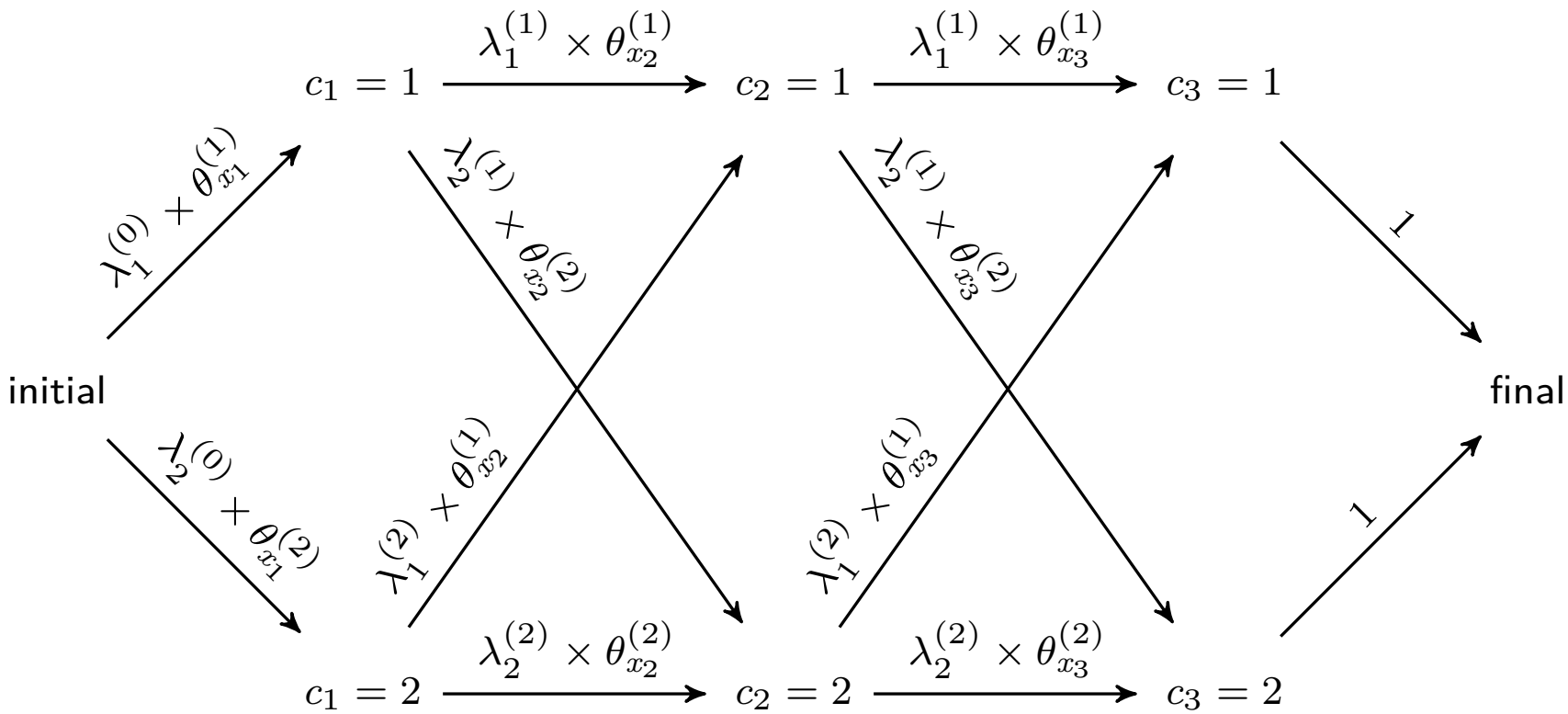
Example: observation x_1^3 tagset $\{1, 2\}$



Compact representation: $O(n \times t)$ nodes and $O(n \times t^2)$ edges

Space of analyses

Example: observation x_1^3 tagset $\{1, 2\}$



Best sequence: path with highest probability

Viterbi algorithm

Enumeration is intractable:

Viterbi algorithm

Enumeration is intractable: $O(t^n)$ paths

Viterbi algorithm

Enumeration is intractable: $O(t^n)$ paths

- but the scoring function factorises

Viterbi algorithm

Enumeration is intractable: $O(t^n)$ paths

- but the scoring function factorises

Dynamic programming

- identify optimal substructure and overlapping subproblems
- the ith decision only affects the score of the $(i+1)$th decision

Dynamic programming | Recursion

Viterbi algorithm

Enumeration is intractable: $O(t^n)$ paths

- but the scoring function factorises

Dynamic programming

- identify optimal substructure and overlapping subproblems
- the ith decision only affects the score of the $(i+1)$th decision

Viterbi recursion

$$\alpha(i,j) = \begin{cases} 1 & \text{if } i = 0 \\ \max_{p \in \{1,\ldots,t\}} \alpha(i-1,p)\lambda_j^{(p)}\theta_{x_i}^{(j)} & \text{otherwise} \end{cases}$$

can also be computed in $\log$-domain

Viterbi implementation

Viterbi recursion

$$\alpha(i,j) = \begin{cases} 1 & \text{if } i = 0 \\ \max\limits_{p \in \{1,\ldots,t\}} \alpha(i-1,p)\lambda_j^{(p)}\theta_{x_i}^{(j)} & \text{otherwise} \end{cases}$$

Implementation:

- for $i = 1, \ldots, n$
  - for $j = 1, \ldots, t$
    - solve $\alpha(i,j)$ and store its value in cell $\mathtt{V}[i,j]$

Viterbi implementation

Viterbi recursion

$$\alpha(i,j) = \begin{cases} 1 & \text{if } i = 0 \\ \max_{p \in \{1,\ldots,t\}} \alpha(i-1,p)\lambda_j^{(p)}\theta_{x_i}^{(j)} & \text{otherwise} \end{cases}$$

Implementation:

- for $i = 1, \ldots, n$
  - for $j = 1, \ldots, t$
    - solve $\alpha(i,j)$ and store its value in cell $\mathtt{V}[i,j]$

Complexity

- space:

Viterbi implementation

Viterbi recursion

$$\alpha(i,j) = \begin{cases} 1 & \text{if } i = 0 \\ \max_{p \in \{1,\ldots,t\}} \alpha(i-1,p)\lambda_j^{(p)}\theta_{x_i}^{(j)} & \text{otherwise} \end{cases}$$

Implementation:

- for $i = 1, \ldots, n$
  - for $j = 1, \ldots, t$
    - solve $\alpha(i,j)$ and store its value in cell $\mathtt{V}[i,j]$

Complexity

- space: $O(n \times t)$ cells in $\mathtt{V}$

Viterbi implementation

Viterbi recursion

$$\alpha(i, j) = \begin{cases} 1 & \text{if } i = 0 \\ \max_{p \in \{1,\ldots,t\}} \alpha(i-1, p) \lambda_j^{(p)} \theta_{x_i}^{(j)} & \text{otherwise} \end{cases}$$

Implementation:

- for $i = 1, \ldots, n$
  - for $j = 1, \ldots, t$
    - solve $\alpha(i, j)$ and store its value in cell $\mathtt{V}[i, j]$

Complexity

- space: $O(n \times t)$ cells in $\mathtt{V}$
- time:

Viterbi implementation

Viterbi recursion

$$\alpha(i, j) = \begin{cases} 1 & \text{if } i = 0 \\ \max_{p \in \{1,\ldots,t\}} \alpha(i-1, p) \lambda_j^{(p)} \theta_{x_i}^{(j)} & \text{otherwise} \end{cases}$$

Implementation:

- for $i = 1, \ldots, n$
  - for $j = 1, \ldots, t$
    - solve $\alpha(i, j)$ and store its value in cell $\mathtt{V}[i, j]$

Complexity

- space: $O(n \times t)$ cells in $\mathtt{V}$
- time: there are $O(n \times t)$ calls to $\alpha(i, j)$
  each requires solving a $\max$ over t values, thus $O(n \times t^2)$

References I