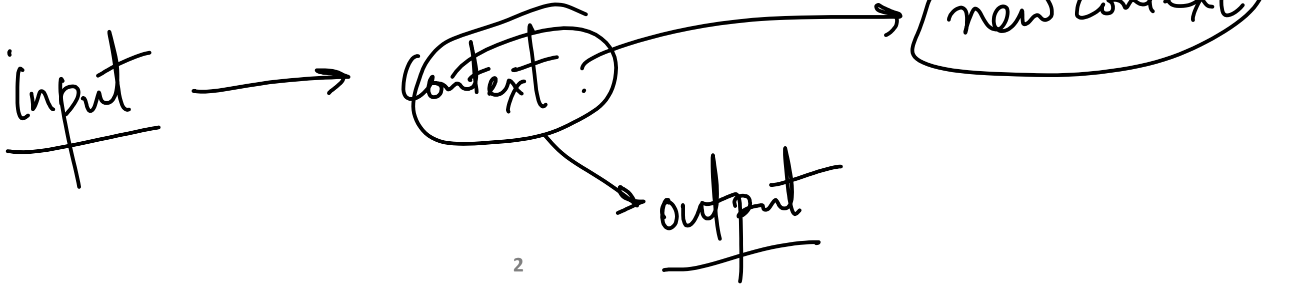
06 | Jan | 2023.

Hidden Markov Models

Markov Model

- Let's say there is a process which generates a sequence:
  - Sequence of written words text
  - Sequence of spoken words (speech)
  - DNA, RNA protein's amino acid sequence, etc.

- The observed sequence is governed by some context which itself keeps updating as the observations are generated.



$w_1 \; w_2 \; \cdots \; w_n$

$P(\text{obs} \mid \text{past history})$

- The probability of a sequence can be written as

$$P(w_{1,n}) = P(w_1)\,\overbrace{P(w_2 \mid w_1)}^{2}\,\overbrace{p(w_3 \mid w_{1,2})}^{3} \ldots \overbrace{P(w_n \mid \underbrace{w_{1,n-1}}_{\text{past history}})}^{n \text{ variable}}$$

$(w_1 \; w_2 \; w_3 \; \cdots \; w_{n-1} \; w_n)$ $\quad w_{1:n} \quad w_{1,n}$ entire sequence with indices 1 to n

- This is the chain rule of probability.
- Modeling the sequence of words $P(w_{1,n})$ requires learning each of the component terms.

$P(w_3 \mid w_1, w_2)$

w_1	w_2	$P(w_3)$
—	—	
—	—	

- We can do something smarter by modelling the context in a better way.

$\underbrace{P(w_{1,n})}$ We want to model. $P(w_1)$ $P(w_2 \mid w_1)$ -

CT CPT needs to be learned

$\boxed{P(w_1\ w_2 \ldots w_n)}$ parameterized (CPT)

$\underbrace{\text{Valid sentence}}_{\text{correct}}$ $\underbrace{w_1 \ldots w_n}$ Prob high.

$\downarrow$

$\underbrace{\text{has errors}}$ Prob $\downarrow$.

Language Model

$w_{1,n} \longrightarrow$ conforms to a language
well formed sentence

Context $\longrightarrow$ observation

- The context sequence can be
  - Observable
  - Hidden

HMMs

$P(w_k \mid \underline{\underline{w_{1;k-1}}})$ context

- An example of observable context model is the n-gram model of text.

Limiting the context

- n-Gram model: Only the previous n-1 words have any effect on the probabilities of the next word.

trigram $P(w_3 \mid w_{1,2})$

$P(w_5 \mid w_{3,4})$

$\frac{w_1}{} \Big| \underbrace{n-1}$ previous words

unigram $P(w$

Bigram $(P(w_2 \mid w_1)$

$P(w_1 \mid w_0)$

- For a trigram model:

$$p(w_n \mid \underbrace{w_1, \ldots, w_{n\ \ 1}}_{\text{past history}}) = p(w_n \mid \underbrace{w_{n\ \ 2}, w_{n\ \ 1}}_{\text{last two words.}})$$

|| Due to limited context (assumption)

- Applying the trigram model gives:

1 2 3 . . . . . 3 variables.

LHS

Joint distribution of $w_{1,n}$

$$\boxed{P(w_{1,n})} = P(w_1)P(w_2 \mid w_1)p(w_3 \mid w_{1,2})\ldots \underbrace{P(w_n \mid w_{n\ \ 2,n\ \ 1})}_{\text{CPTs}}$$

$$= P(w_1)P(w_2 \mid w_1) {}_{i=3}^{n}\ P(w_i \mid w_{i\ \ 2}, w_{i\ \ 1})$$

w_0 w_{-1}

$\prod_{i=1}^{n} P($

Corpus.

- To create such a trigram model, we require the probability of every possible valid trigram.

$$\rightarrow \frac{C(w_{i-2}\, w_{i-1}\, w_i)}{C(w_{i-2}\, w_{i-1})}$$

$$\left\{ \begin{array}{cc|c} w_1 & w_2 & P(w_3) \\ \text{—} & \text{—} & 0 \\ \text{—} & \text{—} & 0 \\ \text{—} & \text{—} & 0 \\ \text{—} & \text{—} & 0 \\ \vdots & \vdots & \vdots \end{array} \right\} \quad P(w_3 = \ ) \ldots$$

- The estimate of the conditional probability

Training dataset

$$P_e(w_i \mid \underline{w_{i-2}, w_{i-1}}) = \frac{C(w_{i-2,i})}{C(w_{i-2,i-1})}$$

e: estimate

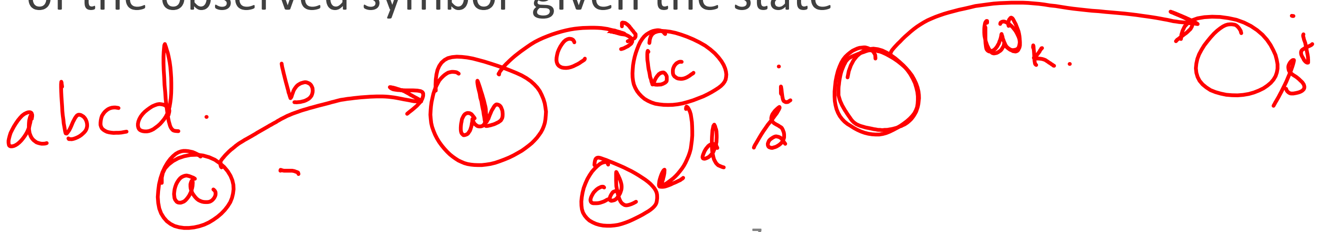
Count

- "To create such a model we simply go through some training text"

Corpus.

$$P_e(\text{'model'} \mid \text{'such'}, \text{'a'}) = \frac{C(\text{'such' 'a' 'model'})}{C(\text{'such' 'a'})} = \frac{1}{1}$$

- After all the hard work, we now have a model which can generate the probability of any string. $P(w_{1,n})$
- This model is called a Markov chain. $\rightarrow$ Context is observable.
- It can be represented as a graph of nodes and arcs
- Nodes are the states $\rightarrow$ context
- Arcs represent transitions from one state to another.
  - An arc is labelled with the observation (emitted symbol) and the probability of the observed symbol given the state



- Why is the state obvious here?

- Given a Markov chain, the probability of an observed sequence can be computed as the product of the probabilities of each transition in the path.

$P(w_{1,n})$

$a \; b \; c \; d$

$s^0, a \rightarrow s^1$

$P(s^1 \mid s^0, a) = 1$

$P(a \mid s^0)$

$$\begin{tikzcd}
s^0 \arrow[r, bend left, "P(a \mid s^0)", "a"'] & s^1 \arrow[r, bend right, "P(b \mid s^1)", "b"'] & s^2 \arrow[r, bend left, "c", "P(c \mid s^2)"'] & s^3 \arrow[r, "P(d \mid s^3)", "d"'] & s^4
\end{tikzcd}$$

$$P(abcd) = \cancel{P(s^0)}\, P(s^1 \mid s^0, a)\, P(s^2 \mid s^1, b)\, P(s^3 \mid s^2, c)\, P(s^4 \mid s^3, d)$$

$P(b \mid s^1)$

What if the states are hidden?

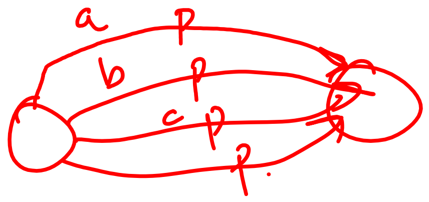
- Now consider the scenario when the states are non-obvious.
- That means, we are uncertain about the state sequence that generated the given observation sequence.
- That means,
  - there can be multiple state sequences that can lead to a given observation sequence.
  - a given state sequence can generate multiple observation sequences
  - for a given state, we can make transition to multiple possible states, even while generating the same output.

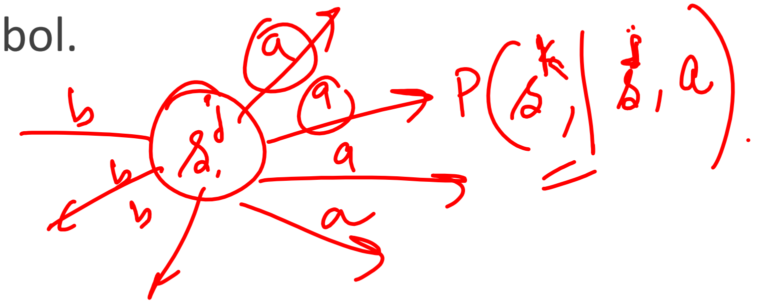
Hidden Markov Models

- So now, transition to multiple states is possible from a given state, even while emitting the same observation (symbol).

|| Difference from a Markov chain.

- HMMs are a generalisation of Markov Chains in which a given state may have several transitions out of it, all with the same $\underset{\text{arc}}{\text{label}}$ (symbol).
  - Recall that in a Markov chain, given a state, the next state is certain (fixed) once you observe a given symbol.





- Since more than one successor state may have the same output, in general there may be several paths through an HMM that produce the same output. sequence.

$$\begin{tikzcd} s \arrow[r, "a"'] \arrow[rr, bend right, "a"'] \arrow[rrr, bend right=40, "a"'] & 1 & 2 & 3 \end{tikzcd}$$

- In such cases, the probability of the output is the sum of the probabilities of all possible paths.

- The Markov chain cannot be used now.
    - What we need is a state transition diagram to represent an HMM.

Moving from a Markov Chain to an HMM

$P(w_3 | w_1, w_2)$ CPT

- Consider that we are learning a trigram model.
- If the training corpus is sparse, it may happen that there is a trigram in the next text that never appeared in the training corpus.
  - In that case, a 0 probability will be assigned to the trigram, i.e. probability of the third word, given the first two words.
- One solution to this problem is to smooth the probabilities by also using the bigram and unigram probabilities.

$\lambda_1 + \lambda_2 + \lambda_3 = 1$

$\lambda_3 > \lambda_2 > \lambda_1$

$$\underbrace{P(w_n | w_{n\ 2,n\ 1})}_{\text{trigram}} = \underbrace{\overset{\lambda_1}{{}_1P_e(w_n)}}_{\text{uni}} + \underbrace{\overset{\lambda_2}{{}_2P_e(w_n | w_{n\ 1})}}_{\text{Bi}} + \underbrace{\overset{\lambda_3}{{}_3P_e(w_n | w_{n\ 2,n\ 1})}}_{\text{Trigram}}$$

0.1   0.3   0.6.

0.6 0.3 0.1

Path 1   Path 2   Path 3

- If we have missing trigram then the bigram and unigram probabilities take over.
- If we have missing trigram and bigram, then we fall back on the unigram probability.
- The probability of a transition has become an addition of multiple components (arcs) between the two states.
    - An HMM will allow multiple paths between two states.

HMM state transition diagram for a trigram model

- Consider that we have seen the symbol ab.
- The next symbol in the sequence is a.
- So the next state is ba
- The trigram $P(a|ab)$ itself can be represented as an HMM

$$P(a|ab) = {}_1P_e(a) + {}_2P_e(a|b) + {}_3P_e(a|ab)$$

- An HMM is a 4 tuple

$$s^1, S, W, E$$

- S: the set of states
- s^1 S is the initial state of the model
- W : the set of output symbols generated/emitted/accepted
- E : the set of edges or transitions.

- For each of the sets S, W, E we assume a canonical ordering of the elements

  $S = s^1, s^2, \dots, s$

  $W = w^1, w^2, \dots, w$

  $E = e^1, e^2, \dots, e$

- The starting state of the HMM is the first element in the ordering of states.
- A transition is a 4-tuple s^i, s^j, w^k, p where s^i is the state from which the transition starts, s^j is the state where the transition ends, w^k is the output symbol generated by the model, p is the probability of taking the transition

- p is the probability of the transition $s^i \overset{w^k}{\quad} s^j$
- No two transitions can have the same starting and ending states as well as the same output value.
- We can leave out the zero probability paths from the graphical representation of the HMM.
- Note that a state s can be the starting state for several transitions that have the same output symbol but go to different ending states.

- It is not possible to know what state the machine has gone into simply by looking at the output.
- Thus, the state sequence followed by an HMM is not deductible from the input. It is hidden.

- There are n+1 states for n outputs

- The probability p associated with a transition $s^i \overset{w_k}{\quad\quad} s^j$

$$P(s^i \overset{w_k}{s} \quad {}^j) = P(S_{t+1} = s^j, W_t = w^k | S_t = s^i)$$

$$- \; P(s^j, w^k | s^i)$$

- When writing $P(s^j, w^k | s^i)$, it is understood that s^i is the prior state and s^j is the next state.

- Markov models assume that the only information affecting the probability of an output, or of the next state, is the prior state.
- As per this Markov Assumption

$$P(w_n, s_{n+1} \,|\, w_{1,n\quad 1}, s_{1,n}) = P(w_n, s_{n+1} \,|\, s_n) = P(s^i \quad \overset{w^k}{s} \quad {}^j)$$

- The probability of a sequence $w_{1,n}$ is the probability of all possible paths through the HMM that can produce this sequence.

In other words:

$$P(w_{1,n}) = \underset{s_{1,n+1}}{\qquad} P(w_{1,n}, s_{1,n+1})$$

The Markov Assumption helps

$$P(w_{1,n}) = \underset{s_{1,n+1}}{} \quad P(w_{1,n}, s_{1,n+1})$$

$$P(w_{1,n}) = \underset{s_{1,n+1}}{} \quad P(s_1)P(w_1, s_2 \,|\, s_1)P(w_2, s_3 \,|\, w_1, s_{1,2})\ldots P(w_n, s_{n+1} \,|\, w_{1,n\ 1}, s_{1,n})$$

- The Markov Assumption helps here

$$P(w_{1,n}) = \underset{s_{1,n+1}}{} \ \overset{n}{\underset{i=1}{}} \quad P(w_i, s_{i+1} \,|\, s_i)$$

Exercise: Graphically visualize the simplification offered by Markov Assumption

An application of HMM: Part of Speech Tagging

- English words can be in more than one PoS class.
- PoS tags can be assigned using HMM.
- But to use HMM we need to phrase the problem of assigning PoS tags to the words as one of assigning probabilities to the input text.
- We assume that the HMM generates outputs which are the words of the corpus.
- We assume that there is some connection between the states and the transitions of the tags.

- Earlier we had for Language Model (LM)

$$P(w_{1,n}) = \underset{s_{1,n+1}}{} P(w_{1,n}, s_{1,n+1})$$

- For PoS tagging, we assume correspondence between states and the tags.
- So,

$$P(w_{1,n}) = \underset{t_{1,n+1}}{} P(w_{1,n}, t_{1,n+1})$$

- Here, $t_{1,n+1}$ is a sequence of $n + 1$ parts of speech or tags.

- The last tag t_{n+1} is the tag achieved after emitting w_n.
- So t_{n+1} is a tag which corresponds to the non-existent word w_{n+1}
- We define the problem of PoS tagging as finding

$$\arg\max_{t_{1,n}} P(t_{1,n} | w_{1,n}) = \arg\max_{t_{1,n}} \frac{P(w_{1,n}, t_{1,n})}{P(w_{1,n})} = \arg\max_{t_{1,n}} P(w_{1,n}, t_{1,n})$$