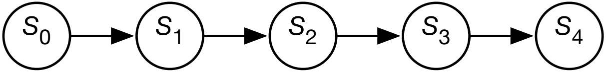
Markov chains

- A Markov chain is a special sort of belief network:



What probabilities need to be specified?

- $P(S_0)$ specifies initial conditions
- $P(S_{i+1} \mid S_i)$ specifies the dynamics

What independence assumptions are made?

- $P(S_{i+1} \mid S_0, \ldots, S_i) = P(S_{i+1} \mid S_i)$.
- Often S_t represents the state at time t.
  The state encodes all of the information about the past that can affect the future.
- "The future is independent of the past given the state."

Stationary Markov chain

- A stationary Markov chain is when for all $i > 0$, $i' > 0$, $P(S_{i+1} \mid S_i) = P(S_{i'+1} \mid S_{i'})$.
- We specify $P(S_0)$ and $P(S_{i+1} \mid S_i)$. Same parameters for each i.
  - Simple model, easy to specify
  - Often the natural model
  - The network can extend indefinitely
- A stationary distribution is a distribution over states such that for ever state s, $P(S_{i+1}{=}s) = P(S_i{=}s)$.
- Under reasonable assumptions, $P(S_k)$ will approach the stationary distribution as $k \to \infty$.

Pagerank

Consider the Markov chain:

- Domain of S_i is the set of all web pages
- $P(S_0)$ is uniform; $P(S_0 = p_j) = 1/N$

$$P(S_{i+1} = p_j \mid S_i = p_k)$$
$$= (1-d)/N + d * \begin{cases} 1/n_k & \text{if } p_k \text{ links to } p_j \\ 1/N & \text{if } p_k \text{ has no links} \\ 0 & \text{otherwise} \end{cases}$$

where there are N web pages and n_k links from page p_k

- $d \approx 0.85$ is the probability someone keeps surfing web
- This Markov chain converges to a stationary distribution over web pages (original $P(S_i)$ for $i = 52$ for 24 million pages and 322 million links):
  Pagerank - basis for Google's initial search engine

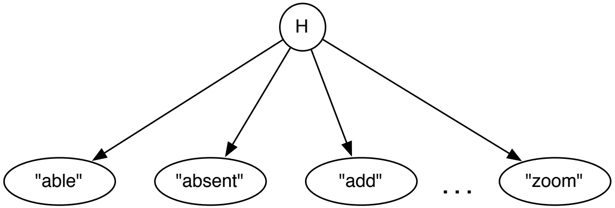
Simple Language Models: set-of-words

Sentence: $w_1, w_2, w_3, \ldots$.

Set-of-words model:

- Each variable is Boolean: *true* when word is in the text and *false* otherwise.
- What probabilities are provided?
  - $P("a")$, $P("aardvark")$, ..., $P("zzz")$
- How do we condition on the question "how can I phone my phone"?

Naive Bayes Classifier: User's request for help



Which of the following probabilities are not required?

A $P(h_i)$ for each help page h_i.

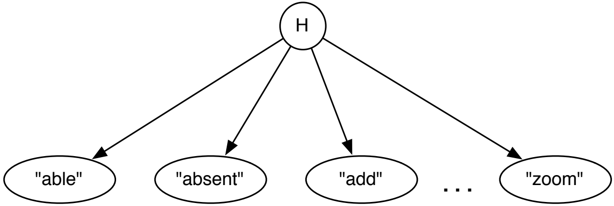
B $P(w_j \mid h_i)$ for each word w_j and help page h_i.

C $P(w_j)$ for each word w_j.

D All of the above are required

E None of the above are required

Naive Bayes Classifier: User's request for help



What is the independence assumption embedded in this model?

A The help pages are independent of each other

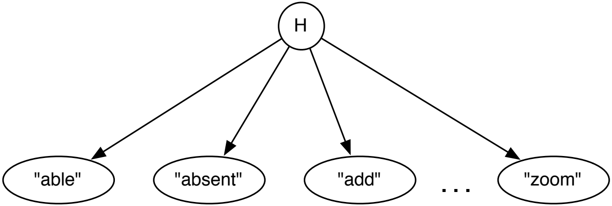
B The help pages are independent of the words.

C The words are independent of each other given the help page.

D The words are independent of each other given no information

E There are no independencies

Naive Bayes Classifier: User's request for help



H is the help page the user is interested in.
What probabilities are required?

- $P(h_i)$ for each help page h_i. The user is interested in one best web page, so $\sum_i P(h_i) = 1$.
- $P(w_j \mid h_i)$ for each word w_j given page h_i. There can be multiple words used in a query.
- Given a help query: condition on the query: words in the query are true and the other are false.
  Display the most likely help page.

Simple Language Models: bag-of-words

Sentence: $w_1, w_2, w_3, \ldots, w_n$.

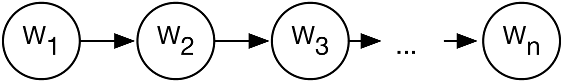
Bag-of-words or unigram:

- Domain of each variable is the set of all words.
- What probabilities are provided?
  - $P(w_i)$ is a distribution over words for each position
- How do we condition on the question "how can I phone my phone"?

Simple Language Models: bigram

Sentence: $w_1, w_2, w_3, \ldots, w_n$.

bigram:

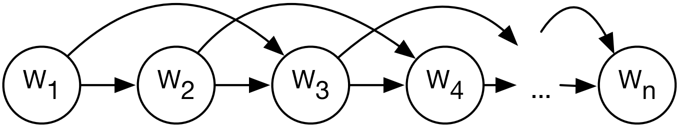


- Domain of each variable is the set of all words.
- What probabilities are provided?
  - $P(w_i \mid w_{i-1})$ is a distribution over words for each position given the previous word
- How do we condition on the question "how can I phone my phone"?

Simple Language Models: trigram

Sentence: $w_1, w_2, w_3, \ldots, w_n$.

trigram:



Domain of each variable is the set of all words.

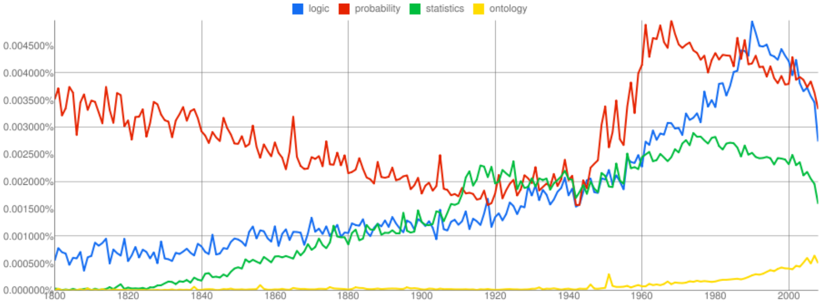
What probabilities are provided?

- $P(w_i \mid w_{i-1}, w_{i-2})$

N-gram

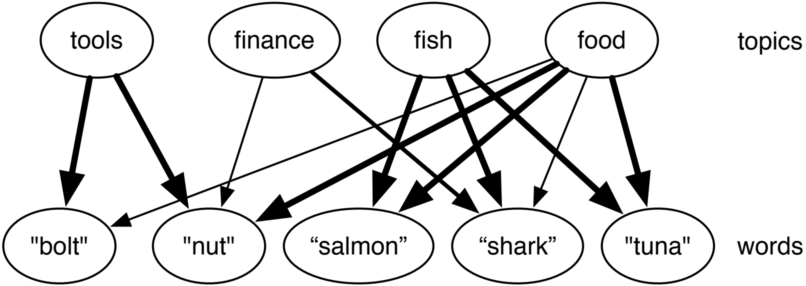
- $P(w_i \mid w_{i-1}, \ldots w_{i-n+1})$ is a distribution over words given the previous $n-1$ words
- ChatGPT (GPT-3) is a 2048-gram, with the conditional probabilities represented using neural-networks (transformers)

Logic, Probability, Statistics, Ontology over time

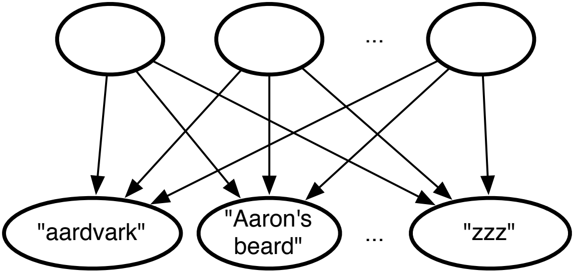


From: Google Books Ngram Viewer
(`https://books.google.com/ngrams`)

Topic Model


tools
finance
fish
food
topics
"bolt"
"nut"
“salmon”
“shark”
"tuna"
words

Google's rephil

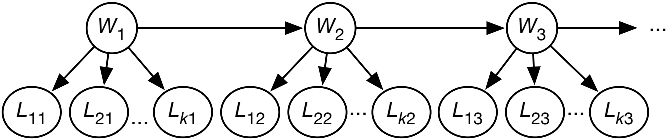


900,000 topics

350,000,000 links

12,000,000 words

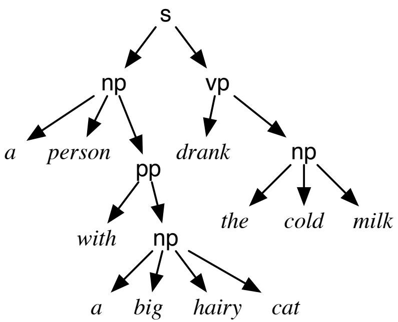
Predictive Typing and Error Correction



$$domain(W_i) = \{"a", "aarvark", \ldots, "zzz", "\bot", "?"\}$$
$$domain(L_{ji}) = \{"a", "b", "c", \ldots, "z", "1", "2", \ldots\}$$

Beyond N-grams

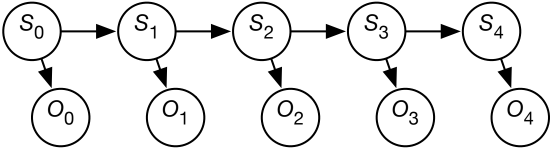
- *A person with a big hairy cat drank the cold milk.*
- Who or what drank the milk?



- Explicitly build a parse tree
- Use a generative model (e.g., a neural network with transformers) to represent $P(word \mid context)$ for a large context (e.g. 2048 tokens for ChatGPT/GPT-3).

Hidden Markov Model

- A Hidden Markov Model (HMM) is a belief network:



The probabilities that need to be specified:

- $P(S_0)$ specifies initial conditions
- $P(S_{i+1} \mid S_i)$ specifies the dynamics
- $P(O_i \mid S_i)$ specifies the sensor model

Filtering

Filtering:

$$P(S_i \mid o_0, \ldots, o_i)$$

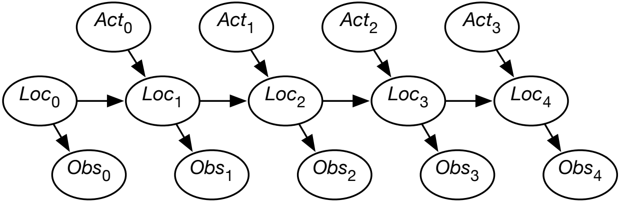
What is the current belief state based on the observation history?

- Observe O_0, query S_0. $P(S_0 \mid o_0)$
- then observe O_1, query S_1. $P(S_1 \mid o_0, o_1)$
- then observe O_2, query S_2. $P(S_2 \mid o_0, o_1, o_2)$
- ...

$$\begin{aligned} P(S_i \mid o_0, \ldots, o_i) &\propto P(o_i \mid S_i o_0, \ldots, o_{i-1}) P(S_i \mid o_0, \ldots, o_{i-1}) \\ &= P(o_i \mid S_i) \sum_{S_{i-1}} P(S_i \mid S_{i-1}) P(s_{i-1} \mid o_0, \ldots, o_{i-1}) \end{aligned}$$

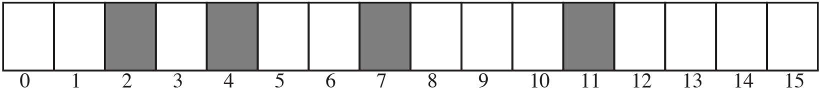
Example: localization

- Suppose a robot wants to determine its location based on its actions and its sensor readings: Localization
- This can be represented by the augmented HMM:



Example localization domain

- Circular corridor, with 16 locations:



- Doors at positions: 2, 4, 7, 11.
- Noisy Sensors
- Stochastic Dynamics
- Robot starts at an unknown location and must determine where it is.

See `probLocalization.py` in AIPython.org

Example Sensor Model

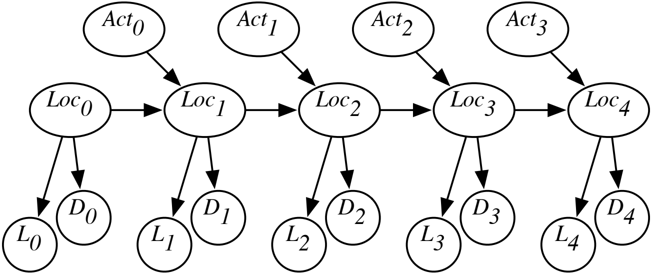
- $P(Observe\ Door \mid At\ Door) = 0.8$
- $P(Observe\ Door \mid Not\ At\ Door) = 0.1$

Example Dynamics Model

- $P(loc_{t+1} = L \mid action_t = goRight \wedge loc_t = L) = 0.1$
- $P(loc_{t+1} = L + 1 \mid action_t = goRight \wedge loc_t = L) = 0.8$
- $P(loc_{t+1} = L + 2 \mid action_t = goRight \wedge loc_t = L) = 0.074$
- $P(loc_{t+1} = L' \mid action_t = goRight \wedge loc_t = L) = 0.002$ for any other location L'.
  - All location arithmetic is modulo 16.
  - The action *goLeft* works the same but to the left.

Combining sensor information

- Example: we can combine information from a light sensor and the door sensor Sensor Fusion

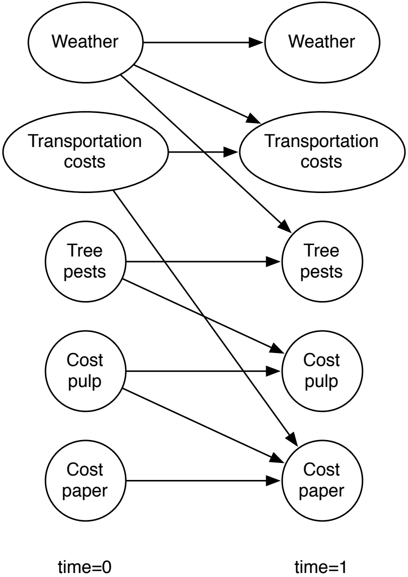


S_t robot location at time t
D_t door sensor value at time t
L_t light sensor value at time t

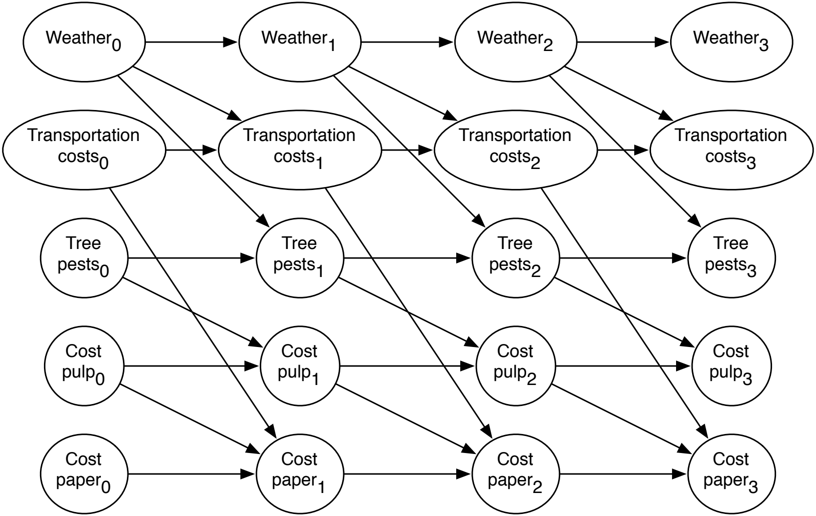
Dynamic Belief Networks

- State is factored into features.
- F_t as the random variable that represented the value of variable F at time t
- The set of features is the same at each time.
- For any time $t > 0$, the parents of variable F_t are variables at time t or time $t - 1$, such that the graph for any time is acyclic. $t = 0$ is a special case.
- stationary model: conditional probability distribution of how each variable depends on its parents is the same for every time $t > 0$.

Two-stage Dynamic Belief Networks



Expanded Dynamic Belief Networks



Time Granularity

- What happens when the time granularity changes from daily to hourly?
- What happens when the time granularity changes from event-based (time advances when an event happens) to hourly?
- A continuous time dynamic belief network contains:
  - a distribution of how long the variable is expected to keep its value
  - what value it will transition to when its value changes.
- This is enough information to compute the transition for any discretization.
  If time step is small enough, ignoring multiple value transitions in each time step will result only in small errors.