Computer Science 188 – Artificial Intelligence

Probability

Ryan Waliany

Note: this is meant to be a simple reference sheet and help students under the derivations. If there's anything that seems "shaky" or incorrect don't hesitate to email me at rwaliany@gmail.com

Introduction to Probability

Random Variables

A random variable is an arbitrary measurement of the world which may have some degree uncertainty. For example, we can have the random variable H = 'will the next coin flip be heads or tails?', C = 'will the robot crash' generally denoted by capital letters where as a realized value are denoted by a lower-case letter.

Assume we are navigating a robot in the Urban DARPA Grand Challenge, robots are driven autonomously on real roads with traffic signals. We can then represent probabilities depending on what evidence we observe. For example, if we saw that $P(C = crash|ACTION = turn\ \ left)$, we could probably argue that $P(C = crash|ACTION = turn\ \ right)$ is greater since, left turns are often more complex and more difficult against traffic. If we also had evidence that there was a car in your way, we know that there's a higher likelihood of you crashing. Thus,

$$P(C = crash|ACTION = turn\ \ left, CAR = true) > P(C = crash|ACTION = turn\_left\ \ )$$

Normalization

$$\sum_x P(x) = 1$$

Marginalization

$$P(X_1) = \sum_{x_2} P(X_1, X_2 = x_2)$$

Conditional Probabilities

$$P(a|b) = \frac{P(a,b)}{P(b)}$$

Normalization Trick

$$P(x_1|x_2) = \frac{P(x_1, x_2)}{P(x_2)} = \frac{P(x_1, x_2)}{\sum_{x_1} P(x_1, x_2)}$$

Chain Rule

$$\mathrm{P(A \ldots Z)} = \mathrm{P(A \mid B \ldots Z)} * \mathrm{P(B \mid C \ldots Z) \ldots P(Y \mid Z)} * \mathrm{P(Z)}$$

Bayes' Rule

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)}$$

Independence

$$P(A, B) = P(A) * P(B)$$

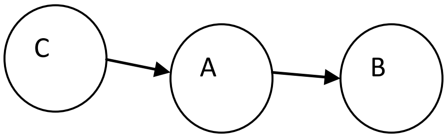
Directed Graphical Models

Representation

Node – Each node represents a random variable

Edges – Represent a relation of causality

A directed arrow from A to B represents that A causes B.



In this case, B is conditionally *independent of* C or $P(B|A, C) = P(B|A)$. More formally, we can represent the probability of a given node conditioned on the world.

$$P(B|\pi(B), \dots) = P(B|\pi(B))$$

$\pi(X)$ is a term that references the parents of X and $\pi(B)$ = A in our example. Thus, we can represent the joint distribution of any direct graphical model as $P(x_1, x_2, \dots) = \prod_{i=1}^{n} P(x_i \, |\pi(x_i))$

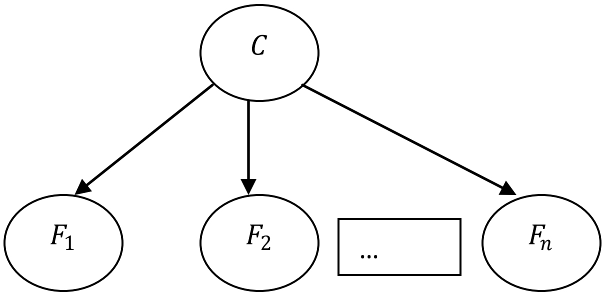
What is the directed graphical model representing N independent coin flips?

Probabilistic Part of Speech Tagging

What is tagging?

Tagging is the process of marking the words in a text corresponding to their part of speech. A tag can be a noun, verb, adjective, adverb—or an even more descriptive subtype such as a proper noun.

The naïve Bayes probabilistic model



We need to determine the probability that the class is a “noun” given a set of features describing the text, $P(C|F_1, \dots, F_n)$

$$P(C|F_1, \dots, F_n) = \frac{P(C, F_1, \dots, F_n)}{P(F_1 \dots, F_n)} = \frac{P(C)P(F_1, \dots, F_n|C)}{P(F_1, \dots, F_n)}$$

However, since we know that $P(F_1, \dots, F_n)$ is constant, we can drop the term when taking a max over classes.

$$\max_c P(C|F_1, \dots, F_n) = P(C)P(F_1, \dots, F_n|C)$$

Since $F_1, \dots, F_n$ are conditionally independent given C,

$$\max_c P(C|F_1, \dots, F_n) = P(C) \prod_{i=1}^{n} P(F_i|C)$$

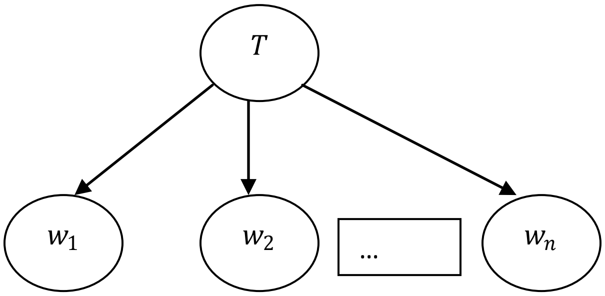
Because we don’t know the exact probabilities, we can estimate them.

$$\max_c \hat{P}(C|F_1, \dots, F_n) = \hat{P}(C) \prod_{i=1}^{n} \hat{P}(F_i|C)$$

$$\hat{P}(C) = \frac{C(C)}{\sum_c C(c)}$$

$$\hat{P}(F_i|C) = \frac{C(F_i, C)}{C(C)}$$

Where we define $\hat{P}$ as simply the frequency. In this example, we assume that C is the tag that we are trying to classify to a word. And the features that are contained in the word are self-determined by us. This can simply be a Boolean flag such that the first letter is capital. Or it can be simply the letters contained in the word, or pairs of letters (bigram), or triples of letters (trigram). Some drawbacks of using higher-order features is that it requires a substantial amount of more training data and it requires more memory to store examples.

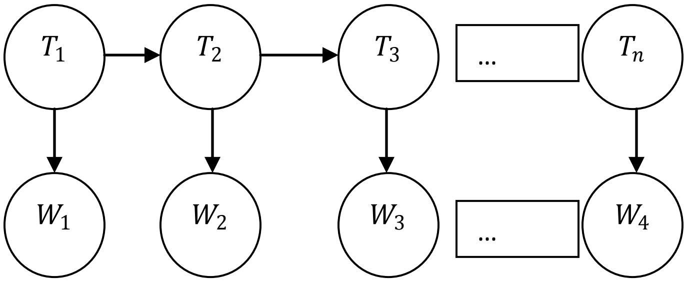


Hidden Markov-Models

A Markov model is distinguished by two properties.

1. Limited horizon: $P(X_{i+1} = x|X_1, \dots, X_n) = P(X_{i+1} = x|X_i)$
2. Time invariant: $P(X_{i+1} = x|X_i) = P(X_2 = x|X_1)$

A hidden Markov model is a form of a markov chain which contains a set of states X and some observable characteristic Y.



We wish to solve the equation:

$$\arg\max_{t_{1,n}} P\left(T_{1,n} \middle| W_{1,n}\right)$$

$$= \arg\max_{t_{1,n}} \frac{P\left(W_{1,n} \middle| T_{1,n}\right) P\left(T_{1,n}\right)}{P\left(W_{1,n}\right)}$$

Since, $P(W_{1,n})$ is constant for maximizing $T_{1,n}$, we can simply remove it from the maximizing equation

$$= \arg\max_{T} P\left(W_{1,n} \middle| T_{1,n}\right) P\left(T_{1,n}\right)$$

$$= \arg\max_{T} (\prod_{1}^{n} P(W_i|T_i))\, P\left(T_{1,n}\right)$$

$$= \arg\max_{T} \prod_{1}^{n} P(W_i|T_i) P(T_i|T_{i-1})$$

For simplicity sake, we assume that $P(T_1|T_0) = 1$, and we can simply calculate the maximum likelihood estimate (MLE)

$$\arg\max_{t_{1,n}} P\left(T_{1,n} \middle| W_{1,n}\right) = \arg\max_{T} \prod_{1}^{n} \hat{P}(W_i|T_i) \hat{P}(T_i|T_{i-1})$$

We could evaluate this equation for all possible tagging, however, that would make tagging of exponential length. One might look at a more efficient algorithm for HMM's such as Viterbi to perform this computation. Viterbi is a dynamic programming algorithm that is for finding the most likely sequence of hidden states.