Cipher Decryption

Cipher Decryption

substitution cypher

I like cats → high prob.
Y work job → low prob.

N Grams / Markov Model

sequence of N tokens in word

ex. "CAT"

$p(A|C) = C \to A$
$p(T|A) = A \to T$

Counting

$$p(x_t | x_{t-1}) = \frac{\text{count}(x_{t-1} \to x_t)}{\text{count}(x_{t-1})}$$

$$p(A|C) = \frac{\#\text{ times "CA" appears}}{\#\text{ times "C" appears}} \text{ in dataset}$$

V letters = V^2 bigram probabilities

$$\underbrace{p(AB)}_{\text{Joint}} = \underbrace{p(B|A)}_{\text{conditional bigram}} \underbrace{p(A)}_{\text{marginal unigram}}$$

Chain Rule:

$$p(ABC) = \boxed{p(C|AB)}\, p(B|A)\, p(A)$$
$$= \boxed{p(C|B)}\, p(B|A)\, p(A)$$

assume C only depends on prev. letter

- Words of any length

$$p(x_1, x_2 \dots x_T) = p(x_1) \underbrace{\prod_{t=2}^{T} p(x_t | x_{t-1})}_{\text{bigram prob. only}}$$

- Prob. of sentence

$$p(w_1, w_2, w_3 \dots) = \prod_{n=1}^{N} p(x_1^{(n)}) \prod_{t=2}^{T(n)} p(x_t^{(n)} | x_{t-1}^{(n)})$$

$$w_n = \{x_1^{(n)}, x_2^{(n)} \dots x_{T(n)}^{(n)}\}$$

- Only letters → so sent. shall not make sense to be valid
- Add One Smoothing (rare bigram will multiply everything by 0 if it appears in testing dataset)

$$p(x_t \mid x_{t-1}) = \frac{\text{count}(x_{t-1} \to x_t) + 1}{\text{count}(x_{t-1}) + V}$$

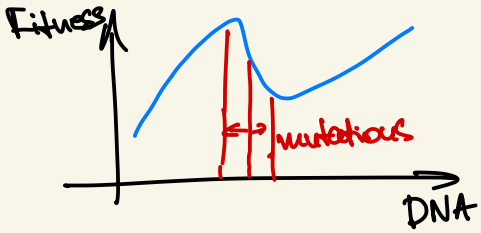
- Max likelyhood
  g gqwq lrpm → low prob.
  i like cats → high prob.
- Log likelyhood (to keep result above 0)

$$\log p(x_1, x_2, x_3 \ldots) = \log p(x_1) + \sum_{t=2}^{T} \log p(x_t \mid x_{t-1})$$

Generic Algorithms

- Brute Force: 26 letters for Cesear Cipher
  26! in mapping word2word
  ⇒ $\sim 4 \cdot 10^{26}$ tries

- Genetic Algos
  Copy DNA, mistakes can happen (not exact copy)
  Substitution/Deletion/Insertion in DNA string
  Time scale of evolution 1 M years
  Mutation can be good or bad → smaller chance of survival



- Numerical Optimization
  gradient descent? = derivative of $f(x)$
  $f(x)$ is log likelyhood of decrypted sentence
  Parameters = map from coded to plaintext
  $f(\text{Params})$ is not differentiable

Model Parameters

DNA model = 26 Letters without Repetition

f(DNA string) = fitness value

DNA string → map → decode(message) → log. likelyhood

- Only Mutation of DNA is allowed (1 swap from parent to child)

  get random DNA → mutate multiple times → if new DNA > old DNA
  keep new DNA

- Too easy to get stuck in local optimum

  DNA pool of size 20
  3 offspring per parent
  sort DNA strings by score
  select only top performers