Statistical Methods in Natural Language Processing

4. Language model smoothing

Pavel Pecina, Jindřich Helcl

4 November, 2025

Charles University
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics

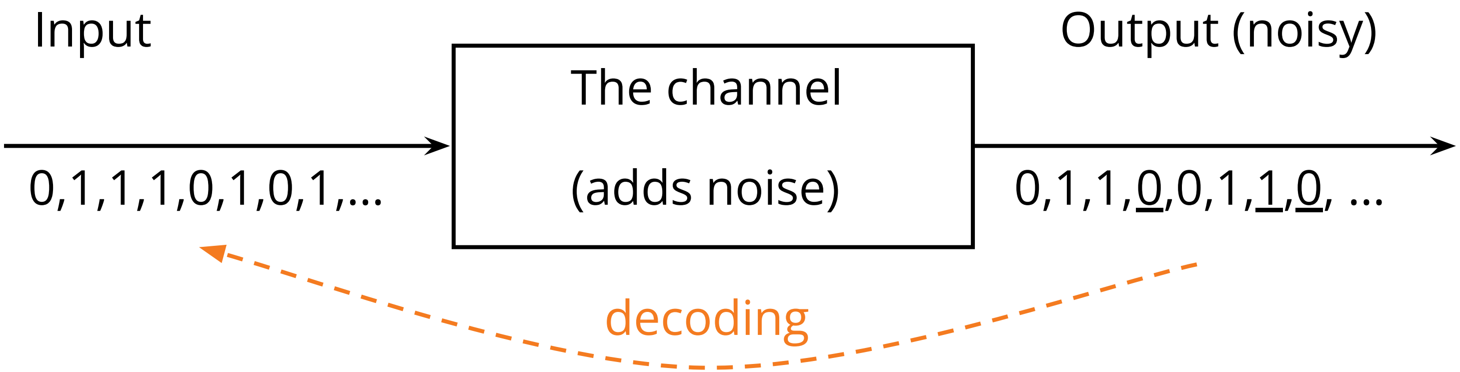
Course Segments

1. Introduction, probability, essential information theory
2. Statistical language modelling (n-gram)
3. Statistical properties of words
4. Word embeddings
5. Hidden Markov models, Tagging

Recap from Last Week

Noisy Channel

Prototypical case:



- Model: probability of error (noise)
- Example: $p(0|1) = 0.25$, $p(1|1) = 0.75$, $p(1|0) = 0.5$, $p(0|0) = 0.5$
- <u>The Task:</u>
  - known: the noisy output; want to know: the input (<u>decoding</u>)

Noisy Channel: The Golden Rule of ... OCR, HR, ASR, MT, ...

- Recall:

$$A_{best} = \text{argmax}_A p(A|B)$$

$$A_{best} = \text{argmax}_A p(B|A)\ \ p(A) / \cancel{p(B)} \qquad \textit{(Bayes Formula)}$$

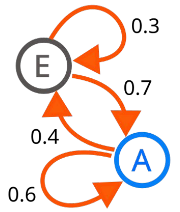
$$\boxed{A_{best} = \text{argmax}_A\ p(B|A)\ p(A)} \qquad \textit{(The Golden Rule)}$$

- Where:
  - $p(B|A)$: the acoustic/image/translation/lexical model
    - application-specific name
    - will explore later
  - $p(A)$: ***<u>the language model</u>***

Markov Chain

- Unlimited memory (cf. previous slide):
  - for w_i, we know <u>all</u> its predecessors $w_1, w_2, w_3, \ldots, w_{i-1}$
- Limited memory:
  - we disregard “too old” predecessors
  - remember only k previous words: $w_{i-k}, w_{i-k+1}, \ldots, w_{i-1}$
  - called “kth order Markov approximation”
- + stationary character (no change over time) → Markov Chain:

$$p(W) \cong \Pi_{i=1..d} p(w_i | w_{i-k}, w_{i-k+1}, \ldots, w_{i-1}), \ d = |W|$$



Picture from Wikipedia

n-gram Language Model

- $(n\text{-}1)^{th}$ order Markov approximation → *n-gram Language Model*:

$$p(W) =^{df} \prod_{i=1..d} p(\underbrace{w_i}_{\text{prediction}} | \overbrace{w_{i-n+1}, w_{i-n+2}, \ldots, w_{i-1}}^{\text{history}})$$

- In particular (assume vocabulary $|V| = 60k$):
  - 0-gram LM: <u>uniform</u> model, $p(w) = 1/|V|$, 1 parameter
  - 1-gram LM: <u>unigram</u> model, $p(w)$, 6×10^4 parameters
  - 2-gram LM: <u>bigram</u> model, $p(w_i|w_{i-1})$, 3.6×10^9 parameters
  - 3-gram LM: <u>trigram</u> model, $p(w_i|w_{i-2},w_{i-1})$, 2.16×10^{14} parameters

Maximum Likelihood Estimate

- MLE: Relative Frequency ...
  - ...best predicts the data at hand (the "training data")
- Trigrams from Training Data T:
  - count sequences of three words in T: $c_3(w_{i-2},w_{i-1},w_i)$
    (*notation: just saying that the three words follow each other*)
  - count sequences of two words in T: $c_2(w_{i-1},w_i)$:
    - either use $\mathbf{c_2(y,z) = \Sigma_w\, c_3(y,z,w)}$
    - or count differently at the beginning (& end) of data!

$$\boxed{p(w_i|w_{i-2},w_{i-1}) =^{est}_{.} c_3(w_{i-2},w_{i-1},w_i) \,/\, c_2(w_{i-2},w_{i-1})}$$

LM: an Example

- Training data: <s> <s> He can buy the can of soda .
- Unigram model (n=8):
  - $p_1(\text{He}) = p_1(\text{buy}) = p_1(\text{the}) = p_1(\text{of}) = p_1(\text{soda}) = p_1(.) = 0.125$, $p_1(\text{can}) = 0.25$
- Bigram model:
  - $p_2(\text{He}|\text{<s>}) = 1$, $p_2(\text{can}|\text{He}) = 1$, $p_2(\text{buy}|\text{can}) = 0.5$, $p_2(\text{of}|\text{can}) = 0.5$, $p_2(\text{the}|\text{buy}) = 1, \ldots$
- Trigram model:
  - $p_3(\text{He}|\text{<s>},\text{<s>}) = 1$, $p_3(\text{can}|\text{<s>},\text{He}) = 1$, $p_3(\text{buy}|\text{He},\text{can}) = 1$, $p_3(\text{of}|\text{the},\text{can}) = 1, \ldots, p_3(.|\text{of},\text{soda}) = 1$
- Entropy: $H(p_1) = 2.75$, $H(p_2) = 0.25$, $H(p_3) = 0$ ← *Great?!*

LM: an Example (The Problem)

- Test data:

$$S = \langle s \rangle \langle s \rangle \text{ It was the greatest buy of all.}$$

- Cross entropy:
    - Even $H_S(p_1)$ fails ($= H_S(p_2) = H_S(p_3) = \infty$), because:
    - all unigrams but $p_1(\text{the})$, $p_1(\text{buy})$, $p_1(\text{of})$ and $p_1(.)$ are 0.
    - all bigram probabilities are 0.
    - all trigram probabilities are 0.
- We want:
    - to make all (theoretically possible) probabilities non-zero.

Language Model Smoothing

The Zero Problem

- "Raw" n-gram language model estimate:
  - necessarily, some zeros
    - !many: trigram model $\rightarrow 2.16\times10^{14}$ parameters, data ~ 10^9 words
  - which are true 0?
    - optimal situation: even the least frequent trigram would be seen several times, in order to distinguish it's probability vs. other trigrams
    - optimal situation cannot happen, unfortunately (open question: how many data would we need?)
  - $\rightarrow$ we don't know
  - we must eliminate the zeros
- Two kinds of zeros: $p(w|h) = 0$, or even $p(h) = 0$!

Why do we need Nonzero Probs?

- To avoid infinite Cross Entropy:
  - happens when an event is found in test data which has not been seen in training data

$H(p) = \infty$: prevents comparing data with > 0 "errors"

- To make the system more robust
  - low count estimates:
    - they typically happen for "detailed" but relatively rare appearances
  - high count estimates:
    - reliable but less "detailed"

Eliminating the Zero Probabilities: Smoothing

- Get new $p'(w)$ (same Ω): almost $p(w)$ but no zeros
- Discount w for (some) $p(w) > 0$: new $p'(w) < p(w)$

$$\Sigma_{w \in \text{discounted}} (p(w) - p'(w)) = D$$

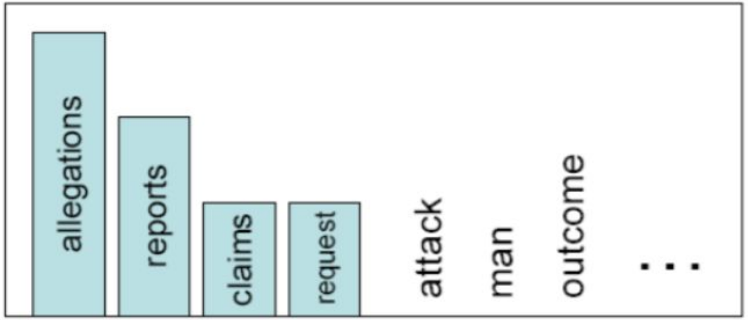
- Distribute D to all w; $p(w) = 0$: new $p'(w) > p(w)$
  - possibly also to other w with low $p(w)$
- For some w (possibly): $p'(w) = p(w)$
- Make sure $\Sigma_{w \in \Omega} p'(w) = 1$
- There are many ways of ***smoothing***

Smoothing Example

- We often want to make predictions from sparse statistics

P(w|denied the)

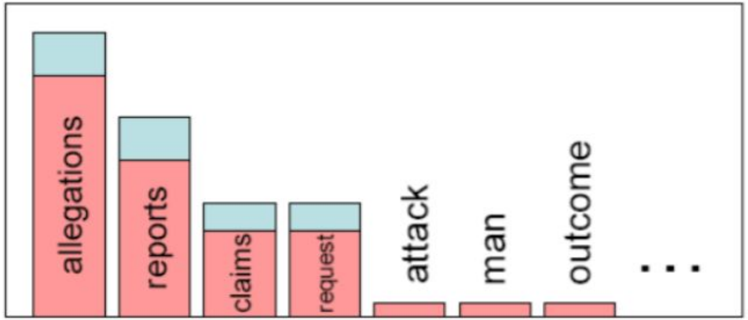
3 allegations
2 reports
1 claims
1 request
7 total



- Smoothing flattens spiky distributions so they generalize better

P(w|denied the)

2.5 allegations
1.5 reports
0.5 claims
0.5 request
2 other
7 total



Smoothing by Adding 1 (Laplace Smoothing)

- Simplest but not really usable:
  - Predicting words w from a vocabulary V, training data T:

    $$\mathbf{p'(w|h) = (c(h,w) + 1) \ / \ (c(h) + |V|)}$$

    - for non-conditional distributions: $p'(w) = (c(w) + 1) / (|T| + |V|)$
  - Problem if $|V| > c(h)$ (as is often the case; even $>> c(h)$!)

Smoothing by Adding 1 (Laplace Smoothing)

- Simplest but not really usable:
  - Predicting words w from a vocabulary V, training data T:

    $$\mathbf{p'(w|h) = (c(h,w) + 1) \ / \ (c(h) + |V|)}$$

    - for non-conditional distributions: $p'(w) = (c(w) + 1) / (|T| + |V|)$
  - Problem if $|V| > c(h)$ (as is often the case; even $>> c(h)$!)
- Example:
  - Training data T: <s> what is it what is small ? $|T| = 8$
  - Vocabulary V = { what, is, it, small, ?, <s>, flying, birds, are, a, bird, . } $|V| = 12$
  - $p(\text{it}) = 0.125$, $p(\text{what}) = 0.25$, $p(.) = 0$
    $p(\text{what is it?}) = 0.25^2 \times 0.125^2 \cong 0.001$, $p(\text{it is flying.}) = 0.125 \times 0.25 \times 0^2 = 0$
  - $p'(\text{it}) = 0.1$, $p'(\text{what}) = 0.15$, $p'(.) = 0.05$
    $p'(\text{what is it?}) = 0.15^2 \times 0.1^2 \cong 0.0002$, $p'(\text{it is flying.}) = 0.1 \times 0.15 \times 0.05^2 \cong 0.00004$

Adding *less* than 1

- Equally simple:
  - Predicting words w from a vocabulary V, training data T:

    $$\mathbf{p'(w|h) = (c(h,w) + \lambda) \ / \ (c(h) + \lambda|V|), \lambda < 1}$$

    - for non-conditional distributions: $p'(w) = (c(w) + \lambda) / (|T| + \lambda|V|)$

Adding *less* than 1

- Equally simple:
  - Predicting words w from a vocabulary V, training data T:

    $$p'(w|h) = (c(h,w) + \lambda) \ / \ (c(h) + \lambda|V|), \lambda < 1$$

    - for non-conditional distributions: $p'(w) = (c(w) + \lambda) / (|T| + \lambda|V|)$
- Example:
  - Training data T: <s> what is it what is small ? $|T| = 8$
  - Vocabulary V = { what, is, it, small, ?, <s>, flying, birds, are, a, bird, . } $|V| = 12$
  - $p(it) = 0.125$, $p(what) = 0.25$, $p(.) = 0$
    $p(\text{what is it?}) = 0.25^2 \times 0.125^2 \cong 0.001$, $p(\text{it is flying.}) = 0.125 \times 0.25 \times 0^2 = 0$
  - **Use $\lambda = 0.1$:** $p'(it) \cong 0.12$, $p'(what) \cong 0.23$, $p'(.) \cong 0.01$
    $p'(\text{what is it?}) = 0.23^2 \times 0.12^2 \cong 0.0007$, $p'(\text{it is flying.}) = 0.12 \times 0.23 \times 0.01^2 \cong 0.00003$

Good-Turing Smoothing

- Suitable for estimation from large data
  - Estimate probability of things that occur c times with the probability of things that occur $c+1$ times:
    $\mathbf{c' = (c+1) \times N(c+1)/N(c)}$
  - Full formula:
    $\mathbf{p_r(w) = (c(w)+1) \times N(c(w)+1) / (|T| \times N(c(w)))}$,
    where $N(c)$ is the count of words with count c (count-of-counts)
    specifically, for $c(w) = 0$ (unseen words), $p_r(w) = N(1) / (|T| \times N(0))$
- good for small counts ($< 5\text{-}10$, where $N(c)$ is high)
- variants (see M&S)
- normalization! (so that we have $\Sigma_w\, p'(w) = 1$)

Good-Turing: An Example

- Good-Turing formula: $p_r(w) = (c(w) + 1) \times N(c(w) + 1) / (|T| \times N(c(w)))$
- Training data T: <s> what is it what is small ? $|T| = 8$
- Vocabulary V = { what, is, it, small, ?, <s>, flying, birds, are, a, bird, . } $|V| = 12$

- $p(\text{it}) = 0.125$, $p(\text{what}) = 0.25$, $p(.) = 0$

  $p(\text{what is it?}) = 0.25^2 \times 0.125^2 \cong 0.001$, $p(\text{it is flying.}) = 0.125 \times 0.25 \times 0^2 = 0$

Good-Turing: An Example

- Good-Turing formula: $p_r(w) = (c(w) + 1) \times N(c(w) + 1) / (|T| \times N(c(w)))$
- Training data T: <s> what is it what is small ? $|T| = 8$
- Vocabulary V = { what, is, it, small, ?, <s>, flying, birds, are, a, bird, . } $|V| = 12$
- p(it) = 0.125, p(what) = 0.25, p(.) = 0

  $p(\text{what is it?}) = 0.25^2 \times 0.125^2 \cong 0.001$, $p(\text{it is flying.}) = 0.125 \times 0.25 \times 0^2 = 0$
- Raw reestimation (N(0) = 6, N(1) = 4, N(2) = 2, N(i) = 0 for i > 2):

  $p_r(\text{it}) = (1+1) \times N(1+1)/(8 \times N(1)) = 2 \times 2/(8 \times 4) = 0.125$

  $p_r(\text{what}) = (2+1) \times N(2+1)/(8 \times N(2)) = 3 \times 0/(8 \times 2) = 0$: keep orig. p(what)

  $p_r(.) = (0+1) \times N(0+1)/(8 \times N(0)) = 1 \times 4/(8 \times 6) \cong 0.083$

Good-Turing: An Example

- Good-Turing formula: $p_r(w) = (c(w) + 1) \times N(c(w) + 1) / (|T| \times N(c(w)))$
- Training data T: <s> what is it what is small ? $|T| = 8$
- Vocabulary $V = \{$ what, is, it, small, ?, <s>, flying, birds, are, a, bird, . $\}$ $|V| = 12$
- $p(\text{it}) = 0.125$, $p(\text{what}) = 0.25$, $p(.) = 0$

  $p(\text{what is it?}) = 0.25^2 \times 0.125^2 \cong 0.001$, $p(\text{it is flying.}) = 0.125 \times 0.25 \times 0^2 = 0$
- Raw reestimation ($N(0) = 6$, $N(1) = 4$, $N(2) = 2$, $N(i) = 0$ for $i > 2$):

  $p_r(\text{it}) = (1+1) \times N(1+1)/(8 \times N(1)) = 2 \times 2/(8 \times 4) = 0.125$

  $p_r(\text{what}) = (2+1) \times N(2+1)/(8 \times N(2)) = 3 \times 0/(8 \times 2) = 0$: keep orig. $p(\text{what})$

  $p_r(.) = (0+1) \times N(0+1)/(8 \times N(0)) = 1 \times 4/(8 \times 6) \cong 0.083$
- Normalize (divide by $1.5 = \Sigma_{w \in |V|} p_r(w)$) and compute:

  $p'(\text{it}) \cong 0.08$, $p'(\text{what}) \cong 0.17$, $p'(.) \cong 0.06$

  $p'(\text{what is it?}) = 0.17^2 \times 0.08^2 \cong 0.0002$, $p'(\text{it is flying.}) = 0.08 \times 0.17 \times 0.06^2 \cong 0.00004$

Smoothing by Combination: Linear Interpolation

- Combine what?
  - distributions of various level of detail vs. reliability
- n-gram models:
  - use (n-1)gram, (n-2)gram, ..., uniform

$\longrightarrow$ reliability

$\longleftarrow$ detail

- Simplest possible combination:
  - sum of probabilities, normalize:
    - bigram: $p(0|0) = 0.8$, $p(1|0) = 0.2$, $p(0|1) = 1$, $p(1|1) = 0$
    - unigram: $p(0) = 0.4$, $p(1) = 0.6$:
    - combined: $p'(0|0) = 0.6$, $p'(1|0) = 0.4$, $p'(0|1) = 0.7$, $p'(1|1) = 0.3$

Language Model Interpolation

- Weight in less detailed distributions using $\lambda = (\lambda_0,\lambda_1,\lambda_2,\lambda_3)$:

$$p'_\lambda(w_i | w_{i-2}, w_{i-1}) = \lambda_3\, p_3(w_i | w_{i-2}, w_{i-1}) + \lambda_2\, p_2(w_i | w_{i-1}) + \lambda_1\, p_1(w_i) + \lambda_0 / |V|$$

- Normalize:

$$\lambda_i > 0,\ \Sigma_{i=0..n} \lambda_i = 1 \qquad (\lambda_0 = 1 - \Sigma_{i=1..n} \lambda_i)\ (n=3)$$

- Estimation using MLE:

1. Fix the p_3, p_2, p_1 and $|V|$ parameters as estimated from training data
2. Find such $\{\lambda_i\}$ which minimizes the cross entropy (maximizes prob. of **data**):

$$-(1/|D|)\Sigma_{i=1..|D|} \log_2(p'_\lambda(w_i|h_i))$$

What Data?

- Training data T? But we will always get $\lambda_3 = 1$!
  - Why? (let p_{iT} be an i-gram distribution ML-estimated from T)
  - minimizing $H_T(p'_\lambda)$ over a vector $\boldsymbol{\lambda}$, $p'_\lambda = \lambda_3 p_{3T} + \lambda_2 p_{2T} + \lambda_1 p_{1T} + \lambda_0/|V|$
    - $H_T(p'_\lambda) = H(p_{3T}) + D(p_{3T} \| p'_\lambda)$; p_{3T} fixed $\rightarrow H(p_{3T})$ fixed, and it is the best
    - which p'_λ minimizes $H_T(p'_\lambda)$?
    - ... a p'_λ for which $D(p_{3T} \| p'_\lambda) = 0$
    - ... and that's p_{3T} (because $D(p \| p) = 0$, as we know).
    - ... and certainly $p'_\lambda = p_{3T}$ if $\lambda_3 = 1$ (maybe in some other cases, too).

$$(p'_\lambda = 1 \times p_{3T} + 0 \times p_{2T} + 0 \times p_{1T} + 0/|V|)$$

What Data?

- Training data T? But we will always get $\lambda_3 = 1$!
  - Why? (let p_{iT} be an i-gram distribution ML-estimated from T)
  - minimizing $H_T(p'_\lambda)$ over a vector $\boldsymbol{\lambda}$, $p'_\lambda = \lambda_3 p_{3T} + \lambda_2 p_{2T} + \lambda_1 p_{1T} + \lambda_0/|V|$
    - $H_T(p'_\lambda) = H(p_{3T}) + D(p_{3T}||p'_\lambda)$; p_{3T} fixed $\rightarrow H(p_{3T})$ fixed, and it is the best
    - which p'_λ minimizes $H_T(p'_\lambda)$?
    - ... a p'_λ for which $D(p_{3T}|| p'_\lambda) = 0$
    - ... and that's p_{3T} (because $D(p||p) = 0$, as we know).
    - ... and certainly $p'_\lambda = p_{3T}$ if $\lambda_3 = 1$ (maybe in some other cases, too).

      $(p'_\lambda = 1 \times p_{3T} + 0 \times p_{2T} + 0 \times p_{1T} + 0/|V|)$
- Thus: do <u>not use the training data for estimation of λ</u>!
  - hold out part of the training data (heldout H); remaining data is the true training data T, the test data S still must be some different data!

The Formulas

- Minimizing $-(1/|H|)\Sigma_{i=1..|H|}\log_2(p'_\lambda(w_i|h_i))$ over $\boldsymbol{\lambda}$:

$$\boxed{\begin{gathered} p'_\lambda(w_i|\, h_i) = p'_\lambda(w_i|\, w_{i-2}, w_{i-1}) = \\ = \lambda_3\, p_3(w_i|\, w_{i-2}, w_{i-1}) + \lambda_2\, p_2(w_i|\, w_{i-1}) + \lambda_1\, p_1(w_i) + \lambda_0 /|V| \end{gathered}}$$

- "Expected Counts (of lambdas)": $j = 0,..,3$

$$\boxed{c(\lambda_j) = \Sigma_{i=1..|H|}\, (\lambda_j p_j(w_i|h_i) / p'_\lambda(w_i|h_i))}$$

- "Next λ": $j = 0,..,3$

$$\boxed{\lambda_{j,next} = c(\lambda_j) / \Sigma_{k=0..3}\, (c(\lambda_k))}$$

The (Smoothing) EM Algorithm

1. Start with some λ, such that $\lambda_j > 0$ for all j $\in$ $0,\ldots,3$.
2. Compute "Expected Counts" for each λ_j.
3. Compute new set of λ_j, using the "Next λ" formula.
4. Start over at step 2, unless a termination condition is met.

- Termination condition: convergence of λ
  - Simply set an ε, and finish if $|\lambda_{j,old} - \lambda_{j,new}| < \varepsilon$ for each j (step 3).
- Guaranteed to converge:
  - Follows from Jensen's inequality, plus a technical proof

Simple Example

- Raw distribution (unigram only; smooth with uniform): $|V|=26$
  $p(a)=0.25$, $p(b)=0.5$, $p(\alpha)=1/64$ for $\alpha \in \{c..r\}$, $= 0$ for the rest: s,t,u,v,w,x,y,z
- Heldout data: baby; use one set of λ (λ_1: unigram, λ_0: uniform)

Simple Example

- Raw distribution (unigram only; smooth with uniform): $|V|=26$
  $p(a)=0.25$, $p(b)=0.5$, $p(\alpha)=1/64$ for $\alpha \in \{c..r\}$, $= 0$ for the rest: s,t,u,v,w,x,y,z
- Heldout data: baby; use one set of $\boldsymbol{\lambda}$ (λ_1: unigram, λ_0: uniform)
- Start with $\lambda_1 = 0.5$, $\lambda_0 = 0.5$;
  - $p'_\lambda(b) = 0.5\times0.5 + 0.5\times1/26 = 0.27$

Simple Example

- Raw distribution (unigram only; smooth with uniform): $|V|=26$
  $p(a)=0.25$, $p(b)=0.5$, $p(\alpha)=1/64$ for $\alpha \in \{c..r\}$, $= 0$ for the rest: s,t,u,v,w,x,y,z
- Heldout data: <u>baby</u>; use one set of $\boldsymbol{\lambda}$ (λ_1: unigram, λ_0: uniform)
- Start with $\lambda_1 = 0.5$, $\lambda_0 = 0.5$;
  - $p'_\lambda(b) = 0.5 \times 0.5 + 0.5 \times 1/26 = 0.27$

Simple Example

- Raw distribution (unigram only; smooth with uniform): $|V|=26$
  $p(a)=0.25$, $p(b)=0.5$, $p(\alpha)=1/64$ for $\alpha \in \{c..r\}$, $= 0$ for the rest: s,t,u,v,w,x,y,z
- Heldout data: <u>baby</u>; use one set of $\boldsymbol{\lambda}$ (λ_1: unigram, λ_0: uniform)
- Start with $\lambda_1 = 0.5$, $\lambda_0 = 0.5$;
  - $p'_\lambda(b) = 0.5\times0.5 + 0.5\times1/26 = 0.27$
  - $p'_\lambda(a) = 0.5\times0.25 + 0.5\times1/26 = 0.14$
  - $p'_\lambda(y) = 0.5\times0 + 0.5\times1/26 = 0.02$

Simple Example

- Raw distribution (unigram only; smooth with uniform): $|V|=26$
  $p(a)=0.25$, $p(b)=0.5$, $p(\alpha)=1/64$ for $\alpha \in \{c..r\}$, $= 0$ for the rest: s,t,u,v,w,x,y,z
- Heldout data: baby; use one set of $\boldsymbol{\lambda}$ (λ_1: unigram, λ_0: uniform)
- Start with $\lambda_1 = 0.5$, $\lambda_0 = 0.5$;
  - $p'_\lambda(b) = 0.5\times0.5 + 0.5\times1/26 = 0.27$
  - $p'_\lambda(a) = 0.5\times0.25 + 0.5\times1/26 = 0.14$
  - $p'_\lambda(y) = 0.5\times0 + 0.5\times1/26 = 0.02$

Simple Example

- Raw distribution (unigram only; smooth with uniform): $|V|=26$
  $p(a)=0.25$, $p(b)=0.5$, $p(\alpha)=1/64$ for $\alpha \in \{c..r\}$, $= 0$ for the rest: s,t,u,v,w,x,y,z
- Heldout data: <u>baby</u>; use one set of $\boldsymbol{\lambda}$ (λ_1: unigram, λ_0: uniform)
- Start with $\lambda_1 = 0.5$, $\lambda_0 = 0.5$;
  - $p'_\lambda(b) = 0.5\times0.5 + 0.5\times1/26 = 0.27$
  - $p'_\lambda(a) = 0.5\times0.25 + 0.5\times1/26 = 0.14$
  - $p'_\lambda(y) = 0.5\times0 + 0.5\times1/26 = 0.02$
  - $c(\lambda_1) = 0.5\times0.5/0.27 + 0.5\times0.25/0.14 + 0.5\times0.5/0.27 + 0.5\times0/0.02 = 2.72$

Simple Example

- Raw distribution (unigram only; smooth with uniform): $|V|=26$
  $p(a)=0.25$, $p(b)=0.5$, $p(\alpha)=1/64$ for $\alpha \in \{c..r\}$, $= 0$ for the rest: s,t,u,v,w,x,y,z
- Heldout data: <u>baby</u>; use one set of $\boldsymbol{\lambda}$ (λ_1: unigram, λ_0: uniform)
- Start with $\lambda_1 = 0.5$, $\lambda_0 = 0.5$;
  - $p'_\lambda(b) = 0.5\times0.5 + 0.5\times1/26 = 0.27$
  - $p'_\lambda(a) = 0.5\times0.25 + 0.5\times1/26 = 0.14$
  - $p'_\lambda(y) = 0.5\times0 + 0.5\times1/26 = 0.02$
  - $c(\lambda_1) = 0.5\times0.5/0.27 + 0.5\times0.25/0.14 + 0.5\times0.5/0.27 + 0.5\times0/0.02 = 2.72$

Simple Example

- Raw distribution (unigram only; smooth with uniform): $|V|=26$

  $p(a)=0.25$, $p(b)=0.5$, $p(\alpha)=1/64$ for $\alpha \in \{c..r\}$, $= 0$ for the rest: s,t,u,v,w,x,y,z
- Heldout data: baby; use one set of $\boldsymbol{\lambda}$ (λ_1: unigram, λ_0: uniform)
- Start with $\lambda_1 = 0.5$, $\lambda_0 = 0.5$;
  - $p'_\lambda(b) = 0.5\times0.5 + 0.5\times1/26 = 0.27$
  - $p'_\lambda(a) = 0.5\times0.25 + 0.5\times1/26 = 0.14$
  - $p'_\lambda(y) = 0.5\times0 + 0.5\times1/26 = 0.02$
  - $c(\lambda_1) = 0.5\times0.5/0.27 + 0.5\times0.25/0.14 + 0.5\times0.5/0.27 + 0.5\times0/0.02 = 2.72$
  - $c(\lambda_0) = 0.5\times0.04/0.27 + 0.5\times0.04/0.14 + 0.5\times0.04/0.27 + 0.5\times0.04/0.02 = 1.28$

Simple Example

- Raw distribution (unigram only; smooth with uniform): $|V|=26$

  $p(a)=0.25$, $p(b)=0.5$, $p(\alpha)=1/64$ for $\alpha \in \{c..r\}$, $= 0$ for the rest: s,t,u,v,w,x,y,z
- Heldout data: baby; use one set of $\boldsymbol{\lambda}$ (λ_1: unigram, λ_0: uniform)
- Start with $\lambda_1 = 0.5$, $\lambda_0 = 0.5$;
  - $p'_\lambda(b) = 0.5 \times 0.5 + 0.5 \times 1/26 = 0.27$
  - $p'_\lambda(a) = 0.5 \times 0.25 + 0.5 \times 1/26 = 0.14$
  - $p'_\lambda(y) = 0.5 \times 0 + 0.5 \times 1/26 = 0.02$

  - $c(\lambda_1) = 0.5 \times 0.5/0.27 + 0.5 \times 0.25/0.14 + 0.5 \times 0.5/0.27 + 0.5 \times 0/0.02 = 2.72$
  - $c(\lambda_0) = 0.5 \times 0.04/0.27 + 0.5 \times 0.04/0.14 + 0.5 \times 0.04/0.27 + 0.5 \times 0.04/0.02 = 1.28$

Simple Example

- Raw distribution (unigram only; smooth with uniform): $|V|=26$

  $p(a)=0.25$, $p(b)=0.5$, $p(\alpha)=1/64$ for $\alpha \in \{c..r\}$, $= 0$ for the rest: s,t,u,v,w,x,y,z
- Heldout data: baby; use one set of $\boldsymbol{\lambda}$ (λ_1: unigram, λ_0: uniform)
- Start with $\lambda_1 = 0.5$, $\lambda_0 = 0.5$;
  - $p'_\lambda(b) = 0.5\times0.5 + 0.5\times1/26 = 0.27$
  - $p'_\lambda(a) = 0.5\times0.25 + 0.5\times1/26 = 0.14$
  - $p'_\lambda(y) = 0.5\times0 + 0.5\times1/26 = 0.02$
  - $c(\lambda_1) = 0.5\times0.5/0.27 + 0.5\times0.25/0.14 + 0.5\times0.5/0.27 + 0.5\times0/0.02 = 2.72$
  - $c(\lambda_0) = 0.5\times0.04/0.27 + 0.5\times0.04/0.14 + 0.5\times0.04/0.27 + 0.5\times0.04/0.02 = 1.28$
  - Normalize: $\lambda_{1,next} = 0.68$, $\lambda_{0,next} = 0.32$
  - Repeat from Step 2 (recompute p'_λ first for efficient computation, then $c(\lambda_i)$, ...)

Simple Example

- Raw distribution (unigram only; smooth with uniform): $|V|=26$

  $p(a)=0.25$, $p(b)=0.5$, $p(\alpha)=1/64$ for $\alpha \in \{c..r\}$, $= 0$ for the rest: s,t,u,v,w,x,y,z
- Heldout data: <u>baby</u>; use one set of $\boldsymbol{\lambda}$ (λ_1: unigram, λ_0: uniform)
- Start with $\lambda_1 = 0.5$, $\lambda_0 = 0.5$;
  - $p'_\lambda(b) = 0.5\times0.5 + 0.5\times1/26 = 0.27$
  - $p'_\lambda(a) = 0.5\times0.25 + 0.5\times1/26 = 0.14$
  - $p'_\lambda(y) = 0.5\times0 + 0.5\times1/26 = 0.02$
  - $c(\lambda_1) = 0.5\times0.5/0.27 + 0.5\times0.25/0.14 + 0.5\times0.5/0.27 + 0.5\times0/0.02 = 2.72$
  - $c(\lambda_0) = 0.5\times0.04/0.27 + 0.5\times0.04/0.14 + 0.5\times0.04/0.27 + 0.5\times0.04/0.02 = 1.28$
  - Normalize: $\lambda_{1,next} = 0.68$, $\lambda_{0,next} = 0.32$
  - Repeat from Step 2 (recompute p'_λ first for efficient computation, then $c(\lambda_i)$, ...)
- Finish when new $\boldsymbol{\lambda}$ almost equal to the old ones (say, < 0.01 difference).

Some More Technical Hints

- Set $V = \{\text{all words from training data}\}$.
  - You may also consider $V = T \cup H$, but it does not make the coding in any way simpler (in fact, harder).
  - But: you must <u>never</u> use the test data for you vocabulary!
- Prepend two "words" in front of all data (<s>):
  - avoids beginning-of-data problems
  - call these index -1 and 0: then the formulas hold exactly
- When $c_n(w,h) = 0$:
  - Assign 0 probability to $p_n(w|h)$ where $c_{n-1}(h) > 0$, but a uniform probability ($1/|V|$) to those $p_n(w|h)$ where $c_{n-1}(h)$ = 0 [this must be done both when working on the heldout data during EM, as well as when computing cross-entropy on the test data!]

Bucketed Smoothing

- Use several λ vectors instead of one, based on (frequency of) history: $\lambda(h)$
- e.g. for h=(micrograms,per) we have $\lambda(h) = (0.999, 0.0009, 0.00009, 0.00001)$

  (because "cubic" is the only word to follow...)
- Actually: not a separate set for each history, but rather a set for "similar" histories ("bucket"):

  $\lambda(b(h))$, where $b: V^2 \rightarrow N$ (in the case of trigrams)
- b classifies histories according to their reliability (~ frequency)

Bucketed Smoothing: The Algorithm

- First, determine the bucketing function b (use heldout!):
  - decide in advance you want e.g. 1000 buckets
  - compute the total frequency of histories in 1 bucket ($f_{max}(b)$)
  - gradually fill your buckets from the most frequent bigrams so that the sum of frequencies does not exceed $f_{max}(b)$ (you might end up with slightly more than 1000 buckets)
- Divide your heldout data according to buckets.
- Apply the previous algorithm to each bucket and its data.

Moodle Quiz

Moodle Quiz

https://dl1.cuni.cz/course/view.php?id=18547