Learning Bigrams from Unigrams

Xiaojin Zhu, Andrew B. Goldberg, Michael Rabbat[†], and Robert Nowak

University of Wisconsin–Madison
[†]McGill University

Privacy attack through index file

your file

Privacy attack through index file

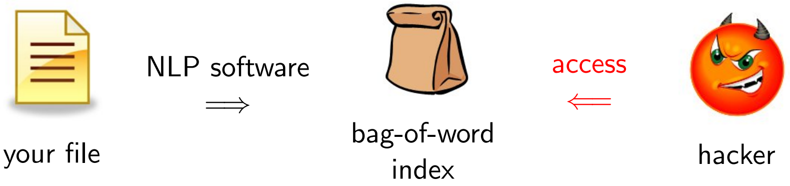
your file

NLP software

$\Longrightarrow$

bag-of-word index

Privacy attack through index file



Privacy attack through index file

your file

NLP software $\Longrightarrow$

bag-of-word index

access $\Longleftarrow$

hacker

What can the hacker learn?

Bag-of-word (BOW) representation

- A document in its original order $\mathbf{z}_1 =$ "$\langle \mathrm{d} \rangle$ really really neat"
- Its BOW: unigram count vector

$$\mathbf{x}_1 = (x_{11}, \ldots, x_{1W}) = (10 \ldots 010 \ldots 020 \ldots)$$

- Can the hacker recover word order from $\mathbf{x}_1$, without extra knowledge of the language?

Bag-of-word (BOW) representation

- A document in its original order $\mathbf{z}_1 =$"$\langle \mathrm{d} \rangle$ really really neat"
- Its BOW: unigram count vector

$$\mathbf{x}_1 = (x_{11}, \ldots, x_{1W}) = (10 \ldots 010 \ldots 020 \ldots)$$

- Can the hacker recover word order from $\mathbf{x}_1$, without extra knowledge of the language? No: $\mathbf{x}_1$ could be from "$\langle \mathrm{d} \rangle$ really neat really" too

Bag-of-word (BOW) representation

- A document in its original order $\mathbf{z}_1 =$ "$\langle \mathrm{d} \rangle$ really really neat"
- Its BOW: unigram count vector

$$\mathbf{x}_1 = (x_{11}, \ldots, x_{1W}) = (10 \ldots 010 \ldots 020 \ldots)$$

- Can the hacker recover word order from $\mathbf{x}_1$, without extra knowledge of the language? No: $\mathbf{x}_1$ could be from "$\langle \mathrm{d} \rangle$ really neat really" too
- What if the hacker has $n \gg 1$ BOWs $\mathbf{x}_1, \ldots, \mathbf{x}_n$?

Bag-of-word (BOW) representation

- A document in its original order $\mathbf{z}_1 =$"$\langle \mathrm{d} \rangle$ really really neat"
- Its BOW: unigram count vector

$$\mathbf{x}_1 = (x_{11}, \ldots, x_{1W}) = (10 \ldots 010 \ldots 020 \ldots)$$

- Can the hacker recover word order from $\mathbf{x}_1$, without extra knowledge of the language? No: $\mathbf{x}_1$ could be from "$\langle \mathrm{d} \rangle$ really neat really" too
- What if the hacker has $n \gg 1$ BOWs $\mathbf{x}_1, \ldots, \mathbf{x}_n$? Traditional wisdom: all it can learn is a unigram LM (word frequencies).

Bag-of-word (BOW) representation

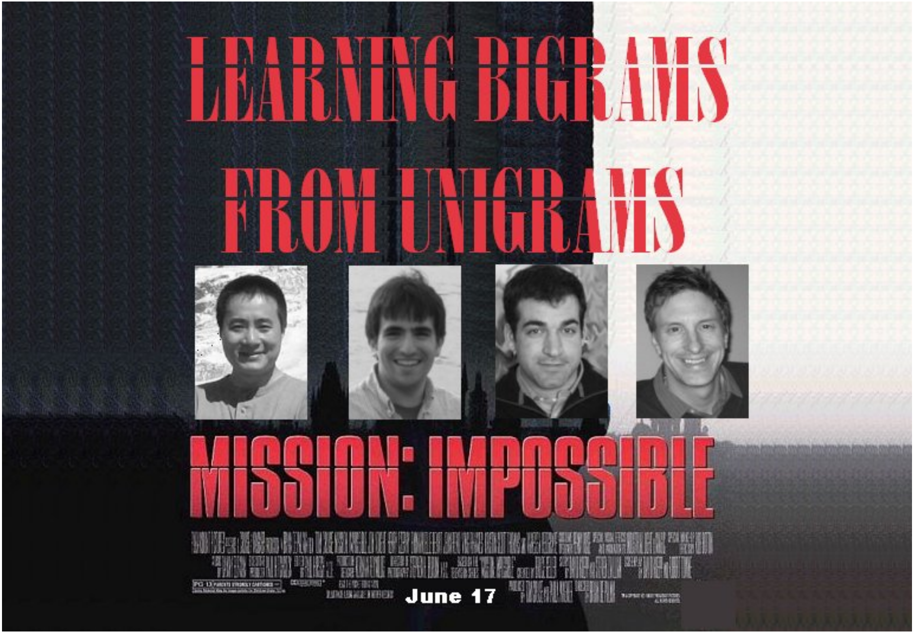
- A document in its original order $\mathbf{z}_1 =$"$\langle \mathrm{d} \rangle$ really really neat"
- Its BOW: unigram count vector

$$\mathbf{x}_1 = (x_{11}, \ldots, x_{1W}) = (10 \ldots 010 \ldots 020 \ldots)$$

- Can the hacker recover word order from $\mathbf{x}_1$, without extra knowledge of the language? No: $\mathbf{x}_1$ could be from "$\langle \mathrm{d} \rangle$ really neat really" too
- What if the hacker has $n \gg 1$ BOWs $\mathbf{x}_1, \ldots, \mathbf{x}_n$? Traditional wisdom: all it can learn is a unigram LM (word frequencies).

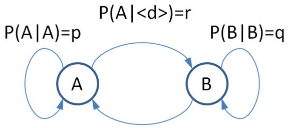
Perhaps surprisingly . . .

We will learn a bigram LM from $\mathbf{x}_1, \ldots, \mathbf{x}_n$, as if we have the ordered documents $\mathbf{z}_1, \ldots, \mathbf{z}_n$.


LEARNING BIGRAMS
FROM UNIGRAMS
MISSION: IMPOSSIBLE
June 17

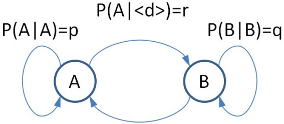
Mission: possible

An example of exact bigram LM recovery:



Mission: possible

An example of exact bigram LM recovery:



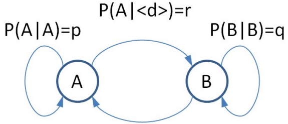
Generative model: 1. $\mathbf{z} \sim \boldsymbol{\theta} = \{p, q, r\}$; 2. $\mathbf{z} \rightarrow \mathbf{x}$ by removing word order.

$$P(\mathbf{x}|\boldsymbol{\theta}) = \sum_{\mathbf{z} \in \sigma(\mathbf{x})} P(\mathbf{z}|\boldsymbol{\theta}) = \sum_{\mathbf{z} \in \sigma(\mathbf{x})} \prod_{j=2}^{|\mathbf{x}|} P(z_j | z_{j-1})$$

e.g., $\mathbf{x} =$($\langle \mathrm{d} \rangle$:1, A:2, B:1) has unique permutations $\sigma(\mathbf{x}) = \{$"$\langle \mathrm{d} \rangle$ A A B", "$\langle \mathrm{d} \rangle$ A B A", "$\langle \mathrm{d} \rangle$ B A A"$\}$.

Mission: possible

An example of exact bigram LM recovery:



Generative model: 1. $\mathbf{z} \sim \boldsymbol{\theta} = \{p, q, r\}$; 2. $\mathbf{z} \rightarrow \mathbf{x}$ by removing word order.

$$P(\mathbf{x}|\boldsymbol{\theta}) = \sum_{\mathbf{z}\in\sigma(\mathbf{x})} P(\mathbf{z}|\boldsymbol{\theta}) = \sum_{\mathbf{z}\in\sigma(\mathbf{x})} \prod_{j=2}^{|\mathbf{x}|} P(z_j|z_{j-1})$$

e.g., $\mathbf{x}$ =($\langle$d$\rangle$:1, A:2, B:1) has unique permutations $\sigma(\mathbf{x}) = \{$"$\langle$d$\rangle$ A A B", "$\langle$d$\rangle$ A B A", "$\langle$d$\rangle$ B A A"$\}$.

Assuming all docs have length $|\mathbf{x}| = 4$, then only 4 kinds of BOWs:

($\langle$d$\rangle$:1, A:3, B:0)	rp^2
($\langle$d$\rangle$:1, A:2, B:1)	$rp(1-p) + r(1-p)(1-q) + (1-r)(1-q)p$
($\langle$d$\rangle$:1, A:0, B:3)	$(1-r)q^2$
($\langle$d$\rangle$:1, A:1, B:2)	1-above

Mission: possible

Let true $\boldsymbol{\theta} = \{r = 0.25, p = 0.9, q = 0.5\}$. Given $\mathbf{x}_1 \dots \mathbf{x}_n, n \to \infty$, the observed frequency of BOWs will be:

($\langle$d$\rangle$:1, A:3, B:0)	20.25%
($\langle$d$\rangle$:1, A:2, B:1)	37.25%
($\langle$d$\rangle$:1, A:0, B:3)	18.75%
($\langle$d$\rangle$:1, A:1, B:2)	100%-above

Mission: possible

Let true $\boldsymbol{\theta} = \{r = 0.25, p = 0.9, q = 0.5\}$. Given $\mathbf{x}_1 \ldots \mathbf{x}_n, n \to \infty$, the observed frequency of BOWs will be:

($\langle$d$\rangle$:1, A:3, B:0)	20.25%
($\langle$d$\rangle$:1, A:2, B:1)	37.25%
($\langle$d$\rangle$:1, A:0, B:3)	18.75%
($\langle$d$\rangle$:1, A:1, B:2)	100%-above

Matching probability with observed frequency

$$\begin{cases} rp^2 = 0.2025 \\ rp(1-p) + r(1-p)(1-q) \\ \qquad\qquad +(1-r)(1-q)p = 0.3725 \\ (1-r)q^2 = 0.1875 \end{cases}$$

exactly recovers $\boldsymbol{\theta}$.

Let's get real

Real documents are not generated from a bigram LM. Maximize log likelihood instead. Parameter $\boldsymbol{\theta} = [\theta_{uv} = P(v|u)]_{W \times W}$.

$$\text{loglik:} \quad \ell(\boldsymbol{\theta}) \equiv 1/C \sum_{i=1}^{n} \log P(\mathbf{x}_i|\boldsymbol{\theta}), \quad C = \sum_{i=1}^{n}(|\mathbf{x}_i| - 1)$$

Multiple local optima. Regularize with prior bigram LM ϕ (estimated from BOWs too). Average KL-divergence over all histories:

$$\mathcal{D}(\boldsymbol{\phi}, \boldsymbol{\theta}) \equiv \frac{1}{W} \sum_{u=1}^{W} KL(\boldsymbol{\phi}_{u\cdot} \| \boldsymbol{\theta}_{u\cdot}).$$

Our optimization problem:

$$\begin{aligned} \max_{\boldsymbol{\theta}} \quad & \ell(\boldsymbol{\theta}) - \mathcal{D}(\boldsymbol{\phi}, \boldsymbol{\theta}) \\ \text{subject to} \quad & \boldsymbol{\theta}\mathbf{1} = \mathbf{1}, \quad \boldsymbol{\theta} \geq 0. \end{aligned}$$

The EM algorithm

It is possible to derive an EM update:

$$\theta_{uv}^{(t)} \equiv P(v|u; \boldsymbol{\theta}^{(t)}) \propto \sum_{i=1}^{n} \sum_{\mathbf{z} \in \sigma(\mathbf{x}_i)} P(\mathbf{z}|\mathbf{x}_i, \boldsymbol{\theta}^{(t-1)}) c_{uv}(\mathbf{z}) + \frac{C}{W} \phi_{uv}$$

- $c_{uv}(\mathbf{z})$ is count of "uv" in $\mathbf{z}$
- Normalize over $v = 1 \ldots W$
- Initialize $\boldsymbol{\theta}^{(0)} = \boldsymbol{\phi}$
- $\sigma(\mathbf{x})$ can be huge. Estimate $\sum_{\mathbf{z} \in \sigma(\mathbf{x}_i)} P(\mathbf{z}|\mathbf{x}_i, \boldsymbol{\theta}^{(t-1)}) c_{uv}(\mathbf{z})$ with importance sampling.

A prior bigram LM ϕ

- Our prior uses no extra language knowledge (can and should be included for specific domains)
- Frequency of document co-occurrence

$$\phi_{uv} \equiv P(v|u; \boldsymbol{\phi}) \propto \sum_{i=1}^{n} \delta(u, v|\mathbf{x})$$

- $\delta(u, v|\mathbf{x}) =$
  - 1, if words u, v co-occur (regardless of their counts) in BOW $\mathbf{x}$
  - 0, otherwise
- Other priors possible, see paper.

Data

Smallish, due to efficiency issues

- SVitchboard 1: small vocabulary Switchboard, with different vocabulary sizes [King et al. 2005]
- SumTime-Meteo: weather forecasts for offshore oil rigs in the North Sea [Sripada et al. 2003]

Corpus	$W-1$	# Docs	# Tokens	$\|\mathbf{x}\|-1$
SV10	10	6775	7792	1.2
SV25	25	9778	13324	1.4
SV50	50	12442	20914	1.7
SV100	100	14602	28611	2.0
SV250	250	18933	51950	2.7
SV500	500	23669	89413	3.8
SumTime	882	3341	68815	20.6

We recover sensible bigrams in $\boldsymbol{\theta}$

Most demoted and promoted bigrams in $\boldsymbol{\theta}$ compared to prior ϕ (sorted by the ratio θ_{hw}/ϕ_{hw} on SV500)

h	$w \downarrow$	$w \uparrow$
i	yep, bye-bye, ah, goodness, ahead	mean, guess, think, bet, agree
you	let's, us, fact, such, deal	thank, bet, know, can, do
right	as, lot, going, years, were	that's, all, right, now, you're
oh	thing, here, could, were, doing	boy, really, absolutely, gosh, great
that's	talking, home, haven't, than, care	funny, wonderful, true, interesting, amazing
really	now, more, yep, work, you're	sad, neat, not, good, it's

Our $\boldsymbol{\theta}$ has good test set perplexity

- Train on $\mathbf{x}_1 \ldots \mathbf{x}_n$, test on ordered documents $\mathbf{z}_{n+1} \ldots \mathbf{z}_m$ (5-fold cross validation, all differences statistically significant)
- "Oracle" bigram trained on $\mathbf{z}_1 \ldots \mathbf{z}_n$ to provide lower bound (Good-Turing)

Corpus	unigram	prior ϕ	$\boldsymbol{\theta}$	oracle	1 EM iter
SV10	7.48	6.52	6.47	6.28	$<$1s
SV25	16.4	12.3	11.8	10.6	0.1s
SV50	29.1	19.6	17.8	14.9	4s
SV100	45.4	29.5	25.3	20.1	11s
SV250	91.8	60.0	47.3	33.7	8m
SV500	149.1	104.8	80.1	50.9	3h
SumTime	129.7	103.2	77.7	10.5	4h

Our $\boldsymbol{\theta}$ reconstructs $\mathbf{z}$ from $\mathbf{x}$ better

- $\mathbf{z} = \mathrm{argmax}_{\mathbf{z}\in\sigma(\mathbf{x})} P(\mathbf{z}|\boldsymbol{\theta}$ or $\phi)$.
- Memory-bounded A* search with admissible heuristic

Accuracy %	whole doc	word pair	word triple
ϕ	30.2	33.0	11.4
$\boldsymbol{\theta}$	31.0	35.1	13.3

(SV500, 5-fold CV, all differences statistically significant)

$\mathbf{z}$ by ϕ	$\mathbf{z}$ by $\boldsymbol{\theta}$
just it's it's it's just going	**it's just it's just it's going**
it's probably out there else something	**it's probably something else out there**
the the have but it doesn't	**but it doesn't have the the**
you to talking nice was it yes	**yes it was nice talking to you**
that's well that's what i'm saying	**well that's that's what i'm saying**
a little more here home take	**a little more take home here**
and they can very be nice too	**and they can be very nice too**
i think well that's great i'm	**well i think that's great i'm**
but was he because only always	**but only because he was always**

We thank

Wisconsin Alumni Research Foundation
NSF CCF-0353079, CCF-0728767
NSERC of Canada

and you.