Two Topics in High Dimensional Space: Hubness and Low Dimensional Geometry

Jiaji Huang

July 18th

Agenda

1. Hubness explained
2. Bilingual Lexicon Induction (BLI) and Hubness Reduction [1]
3. Low-dimensionality: Miscellaneous Results [2, 3]

Agenda

1. Hubness explained
2. Bilingual Lexicon Induction (BLI) and Hubness Reduction [1]
3. Low-dimensionality: Miscellaneous Results [2, 3]

Hubness [1]

- Hubs: “popular” nearest neighbors

[1] M. Radovanović *et. al*, JMLR 2010

Hubness [1]

- Hubs: "popular" nearest neighbors
- N_k (k-occurrence against a query set): "the number of times a point being the k-NN of query items"

[1] M. Radovanović *et. al*, JMLR 2010

Hubness

- Example: multivariate Gaussian, $k = 5$
- For each data point, retrieve its 5-NN among all the generated data points

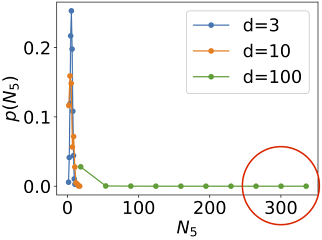


Figure: Distribution of 5-occurrence

- Conclusion: When d big, some data points are being retrieved too many times!

Hubness Degrades NN Search

Evidence seen in

- Bilingual Lexicon Induction [2]
- Document classification [3]
- Audio Retrieval [4]
- Zero-shot Image labeling [5]

[2] Dinu *et. al.* ICLR 2014
[3] Suzuki *et. al.* EMNLP 2013
[4] Aucouturier *et. al.* Pattern recognition 2008
[5] Shigeto *et. al.* KDD 2015

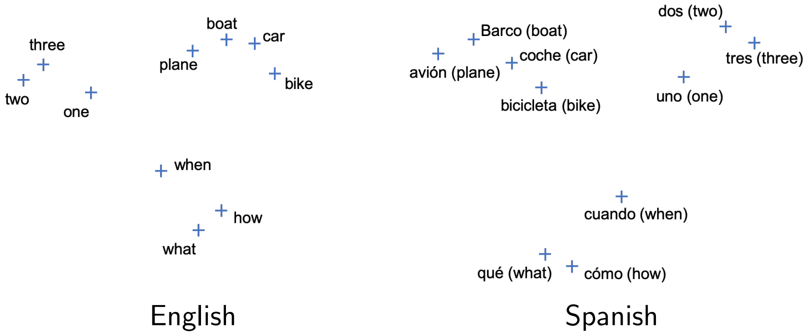
Agenda

Bilingual Lexicon Induction

- How to translate words without parallel corpora?
- isometry between word embedding spaces of two languages

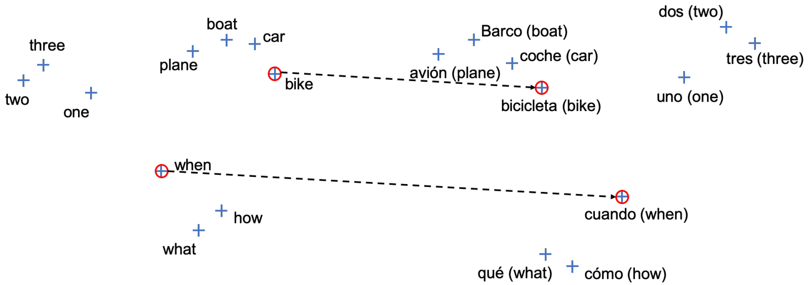
Bilingual Lexicon Induction

- How to translate words without parallel corpora?
- isometry between word embedding spaces of two languages



Bilingual Lexicon Induction

Introduce Some anchors by using a seeding dictionary



- Align the anchors via a rotation
- More translation pairs can be induced by Nearest Neighbor (NN) Search

Inverted Softmax Mitigates Hubness [6]

- Distance matrix $D_{i,j}$ where

  $i = 1, \ldots, m$: index of source; $\quad j = 1, \ldots, n$: index of target

- Kernel matrix $\exp(-D_{i,j}/\epsilon)$

	Target, n=3		
Source, m=3	0.2	0.4	0.8
	0.6	0.2	0.8
	0.4	0.2	0.4

Kernel

[6] Smith *et. al.*, ICLR 2017

Inverted Softmax Mitigates Hubness [6]

- Distance matrix $D_{i,j}$ where

  $i = 1, \ldots, m$: index of source; $j = 1, \ldots, n$: index of target

- Kernel matrix $\exp(-D_{i,j}/\epsilon)$

Target, n=3

Source, m=3			
	0.2	0.4	0.8
	0.6	0.2	0.8
	0.4	0.2	0.4

Kernel

Target, n=3

Source, m=3			
	1/6	1/2	2/5
	1/2	1/4	2/5
	1/3	1/4	1/5

normalize columns

[6] Smith *et. al.*, ICLR 2017

Inverted Softmax Mitigates Hubness [6]

- Distance matrix $D_{i,j}$ where

  $i = 1, \ldots, m$: index of source; $\quad j = 1, \ldots, n$: index of target

- Kernel matrix $\exp(-D_{i,j}/\epsilon)$

Source, m=3 \ Target, n=3			
	0.2	0.4	0.8
	0.6	0.2	0.8
	0.4	0.2	0.4

Kernel

Source, m=3 \ Target, n=3			
	1/6	1/2	2/5
	1/2	1/4	2/5
	1/3	1/4	1/5

normalize columns

Source, m=3 \ Target, n=3			
	5/32	15/32	12/32
	10/23	5/23	8/23
	20/47	15/47	12/47

normalize rows

[6] Smith *et. al.*, ICLR 2017

Doubts ...

- ISF works but not clear why
- What if run the normalization for multiple times?

Let's back off a bit ...

Viewing NN as an Optimization Problem

- NN is equivalent to $\arg\max_j \exp(-D_{i,j}/\epsilon)$.

Viewing NN as an Optimization Problem

- NN is equivalent to $\arg\max_j \exp(-D_{i,j}/\epsilon)$.
- Obviously,

$$\arg\max_j \exp(-D_{i,j}/\epsilon) \equiv \arg\max_j \frac{\exp(-D_{i,j}/\epsilon)}{\sum_{j'} \exp(-D_{i,j'}/\epsilon)}$$

Viewing NN as an Optimization Problem

- NN is equivalent to $\arg\max_j \exp(-D_{i,j}/\epsilon)$.
- Obviously,

$$\arg\max_j \exp(-D_{i,j}/\epsilon) \equiv \arg\max_j \frac{\exp(-D_{i,j}/\epsilon)}{\sum_{j'} \exp(-D_{i,j'}/\epsilon)}$$

- RHS is the solution of the following

$$\begin{aligned} &\min_{\mathbf{P}} \langle \mathbf{D}, \mathbf{P} \rangle + \epsilon \sum_{i,j} P_{i,j} \log P_{i,j} \\ &s.t. P_{i,j} \geq 0, \ \sum_j P_{i,j} = 1 \end{aligned} \tag{1}$$

Viewing NN as an Optimization Problem

Proposition 1 (NN as an optimization problem)

The NN criterion is equivalent to $\arg\max_j P_{i,j}$*, where* $\mathbf{P}$ *is the solution of the following optimization problem,*

$$\begin{aligned} &\min_{\mathbf{P}} \langle \mathbf{D}, \mathbf{P} \rangle + \epsilon \sum_{i,j} P_{i,j} \log P_{i,j} \\ &s.t. P_{i,j} \geq 0, \ \sum_j P_{i,j} = 1 \end{aligned} \quad . \tag{$\mathcal{P}_0$}$$

Equal Preference Assumption

- Column mean of $\mathbf{P}$ encodes how popular each target item is

5/32	15/32	12/32
10/23	5/23	8/23
20/47	15/47	12/47

Equal Preference Assumption

- Column mean of $\mathbf{P}$ encodes how popular each target item is

0.339	0.335	0.326

Equal Preference Assumption

- Column mean of $\mathbf{P}$ encodes how popular each target item is

0.339	0.335	0.326

- Let's enforce them to be equally "popular"

Definition 1 (Equal Preference Assumption)

$$pf_j \triangleq \frac{1}{m}\sum_i P_{i,j} = \frac{1}{n}, \quad \forall j$$

Equal Preference Assumption

- Column mean of $\mathbf{P}$ encodes how popular each target item is

0.339	0.335	0.326

- Let's enforce them to be equally "popular"

Definition 1 (Equal Preference Assumption)

$$pf_j \triangleq \frac{1}{m} \sum_i P_{i,j} = \frac{1}{n}, \quad \forall j$$

- Approximately holds when m, n are huge

Hubless Nearest Neighbor (HNN)

Applying the assumption:

Definition 2 (HNN)

HNN is the criterion that retrieves index $\arg\max_j P_{i,j}$, *where* $\mathbf{P}$ *is the solution of problem*

$$\begin{aligned} &\min_{\mathbf{P}} \langle \mathbf{D}, \mathbf{P} \rangle + \epsilon \sum_{i,j} P_{i,j} \log P_{i,j} \\ &s.t. P_{i,j} \geq 0, \ \sum_j P_{i,j} = 1, \ \frac{1}{m} \sum_i P_{i,j} = \frac{1}{n}. \end{aligned} \qquad (\mathcal{P}_1)$$

Sovling $(\mathcal{P}_1)$

Algorithm 1 Sinkhorn Iteration

Input: $\mathbf{D}$
Output: $\mathbf{P}$

$\mathbf{P} \leftarrow \exp(-\mathbf{D}/\epsilon)$ where $\exp$ is on elements.
while stopping criteria not met **do**
  // normalize columns
  $\mathbf{P} \leftarrow \mathbf{P}\text{diag}\left\{\frac{m}{n}./(\mathbf{P}^\top \mathbf{1})\right\}$
  // normalize rows
  $\mathbf{P} \leftarrow \text{diag}\{1./(\mathbf{P}\mathbf{1})\}\mathbf{P}$
end while

- ISF is a single step of Sinkhorn iteration!
- Less efficient if m, n are huge

Dual of $(\mathcal{P}_1)$ [7]

Proposition 2 (Dual of $(\mathcal{P}_1)$)

The solution of problem $(\mathcal{P}_1)$ can be expressed as

$$P_{i,j} = \frac{\exp\left(\frac{\beta_j - D_{i,j}}{\epsilon}\right)}{\sum_j \exp\left(\frac{\beta_j - D_{i,j}}{\epsilon}\right)}, \tag{2}$$

where β_j is the solution of

$$\min_{\boldsymbol{\beta}} \sum_i \left\{ \ell_i \triangleq \left[\epsilon \log \sum_j \exp\left(\frac{\beta_j - D_{i,j}}{\epsilon}\right) - \frac{1}{n} \sum_j \beta_j \right] \right\} \tag{$\mathcal{D}$}$$

- $\exp(-\beta_j/\epsilon)$ is a column normalizer
- ISF simply sets the normalizer as $\sum_j \exp(-D_{i,j}/\epsilon)$

[7] Genevay *et. al.*, 2016

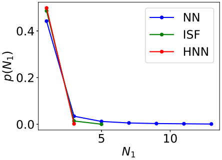
Dual Solver

Algorithm 2 Dual Solver for Problem ($\mathcal{P}_1$)

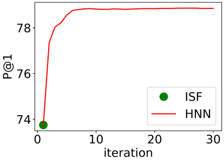
Input: $\mathbf{D}$
Output: $\mathbf{P}$

$\boldsymbol{\beta} \leftarrow \mathbf{0}$
while stopping criteria not met **do**
  for $i = 1, \ldots, m$ **do**
    // parallelizable
    Compute gradient $\nabla \ell_i$.
  end for
  $\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} - \eta \cdot \frac{1}{m} \sum_i \nabla \ell_i$
end while
Compute $\mathbf{P}$ by E.q. (2) and return.

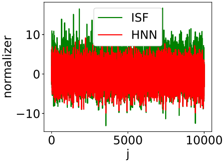
- dimension $d = 300$, number of classes $C = 10K$, $m = n = 10K$
- $p(\mathbf{x}|c) = \mathcal{N}(\boldsymbol{\mu}_c, 0.01), c = 1, \ldots, C$, uniform prior $p(c)$



(a) 1-occurrence



(b) Algo. 1 iterations



(c) Column Normalizer

Illustrative Experiments (contd.)

Top-1, top-5, top-10 retrieval accuracies

	P@1	P@5	P@10
NN	51.71	71.88	78.51
ISF	73.75	88.57	92.36
HNN (Algorithm 1)	**78.85**	**91.43**	**94.60**
HNN (Algorithm 2)	**78.84**	**91.41**	**94.59**

BLI experiments[8]

source \ target		en	es	fr	it	pt	de
en	NN		54.98	55.66	46.30	37.02	53.03
	ISF		70.35	72.31	63.75	53.02	65.75
	CSLS		71.21	72.62	64.11	53.54	**66.50**
	HNN		**71.34**	**73.65**	**64.91**	**54.03**	64.61
es	NN	59.87		60.76	61.95	66.94	48.73
	ISF	73.11		76.82	76.33	78.67	61.70
	CSLS	73.02		76.44	76.44	80.29	**62.29**
	HNN	**74.38**		**78.24**	**77.86**	**81.09**	60.78
fr	NN	61.60	61.73		59.43	46.31	57.10
	ISF	74.46	75.72		73.78	60.89	69.07
	CSLS	74.88	76.68		74.34	62.06	**70.34**
	HNN	**75.97**	**77.23**		**75.12**	**63.10**	67.94
it	NN	51.38	64.63	61.45		51.91	50.68
	ISF	65.57	77.76	76.64		67.32	63.58
	CSLS	65.32	78.46	76.74		68.85	**64.57**
	HNN	**67.57**	**79.75**	**78.56**		**70.33**	62.96
pt	NN	42.21	68.93	47.48	50.98		37.95
	ISF	55.76	81.67	64.37	68.37		51.07
	CSLS	54.75	81.98	63.68	67.92		**51.77**
	HNN	**57.42**	**83.96**	**66.19**	**70.44**		49.93
de	NN	56.06	44.33	52.78	45.44	33.20	
	ISF	**69.74**	**60.77**	**71.59**	**65.99**	**52.74**	
	CSLS	68.65	59.21	69.88	63.69	50.72	
	HNN	69.20	60.22	70.71	65.09	52.08	

[8] Follow setups in `https://github.com/facebookresearch/MUSE`

Analysis

- Why HNN is less impressive on German?

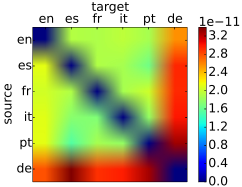
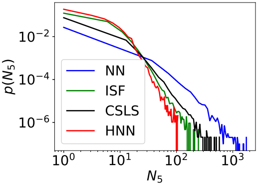


Figure: var$[pf_j]$ for all the pairs

- Hubness reduced?



Analysis

Table: Some representative "hubs"

word	N_5 by NN	N_5 by HNN	frequency rank
conspersus	1,776	0	484,387
oryzopsis	1,235	5	472,161
these	1,042	25	122
s+bd	912	16	440,835
were	798	24	40
you	474	20	50
would	467	40	73

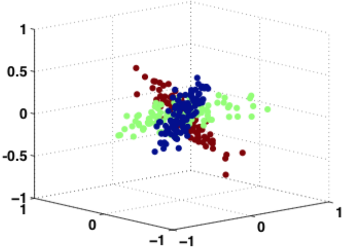
- Typos (extremely low-frequency) are likely to be hubs [9]
- Some functional words (very frequent) can also be hubs

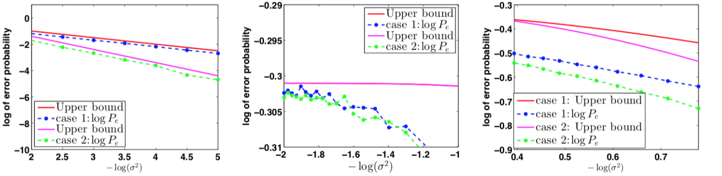
[9] Dinu *et. al.*, ICLR 2014

Agenda

1. Hubness explained
2. Bilingual Lexicon Induction (BLI) and Hubness Reduction [1]
3. Low-dimensionality: Miscellaneous Results [2, 3]

Principal Angles and Subspace Classification [2]





(a) high SNR regime (b) low SNR regime (c) moderate SNR regime

Robustness and Local Geometry Preservation [3]

- Application of (K, ϵ)-robustness [10]
- Deep Network + $\sum_{(i,j) \in NB} |d(f_i, f_j) - d(x_i, x_j)|$
  where NB is local neighborhood

[10] Xu *et. al.*, Machine Learning 2012

Reference I

[1] J. Huang *et. al*. Hubless Nearest Neighbor Search for Bilingual Lexicon Induction. ACL 2019.

[2] J. Huang *et. al*. The Role of Principal Angles in Subspace Classification. IEEE Transactions on Signal Processing. 2015

[3] J. Huang *et. al*. Discrimnative Robust Feature Transformation. NIPS 2015