Word Embedding & Query Expansion

Jiayi Chen

Query Expansion with Locally-Trained Word Embeddings

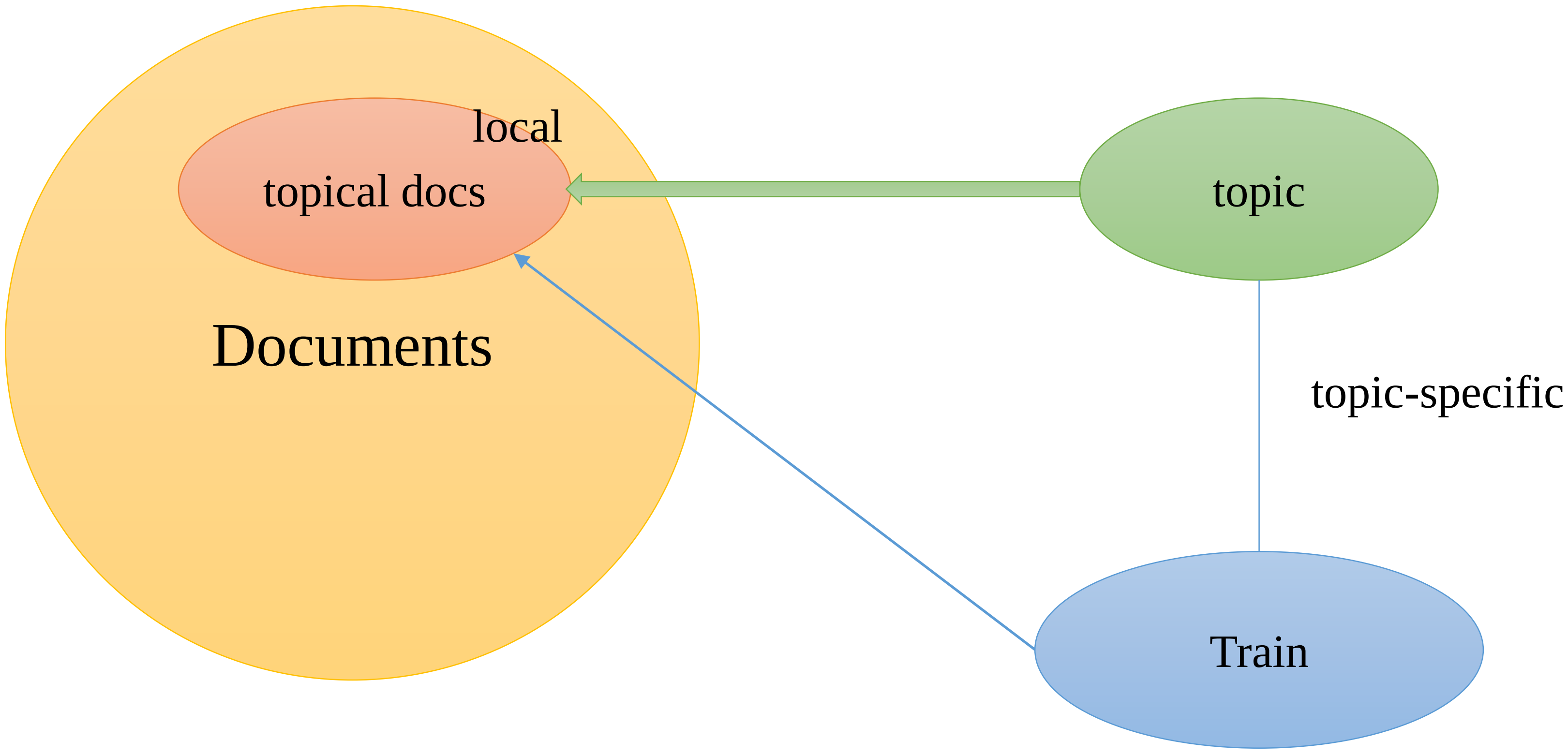
Fernando Diaz, Bhaskar Mitra, Nick Craswell

ACL'16

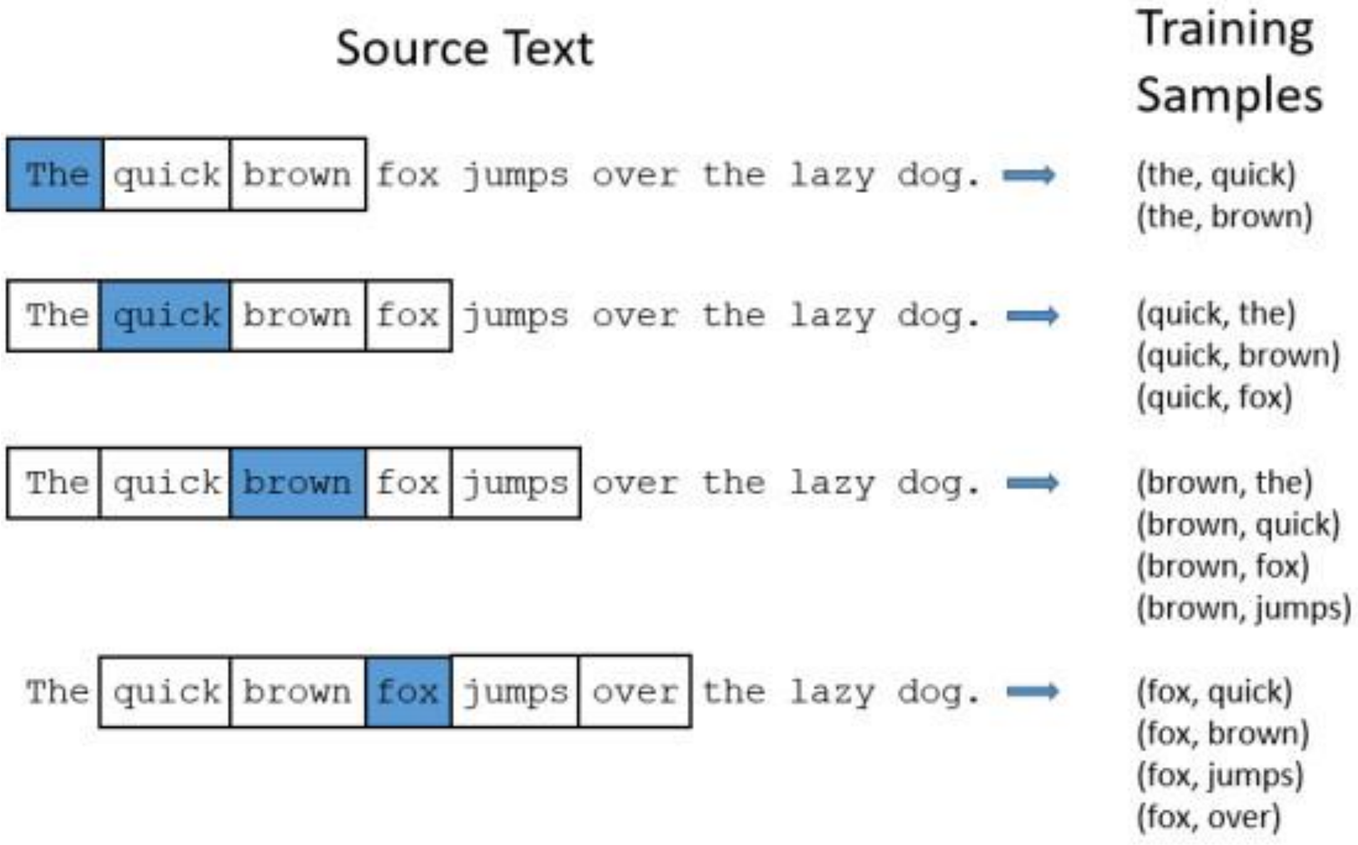
Main Idea

- Global word embeddings capture coarse representations for topics.
- Locally-trained(topical) word embeddings will outperform than global embeddings in retrieval tasks.

Idea



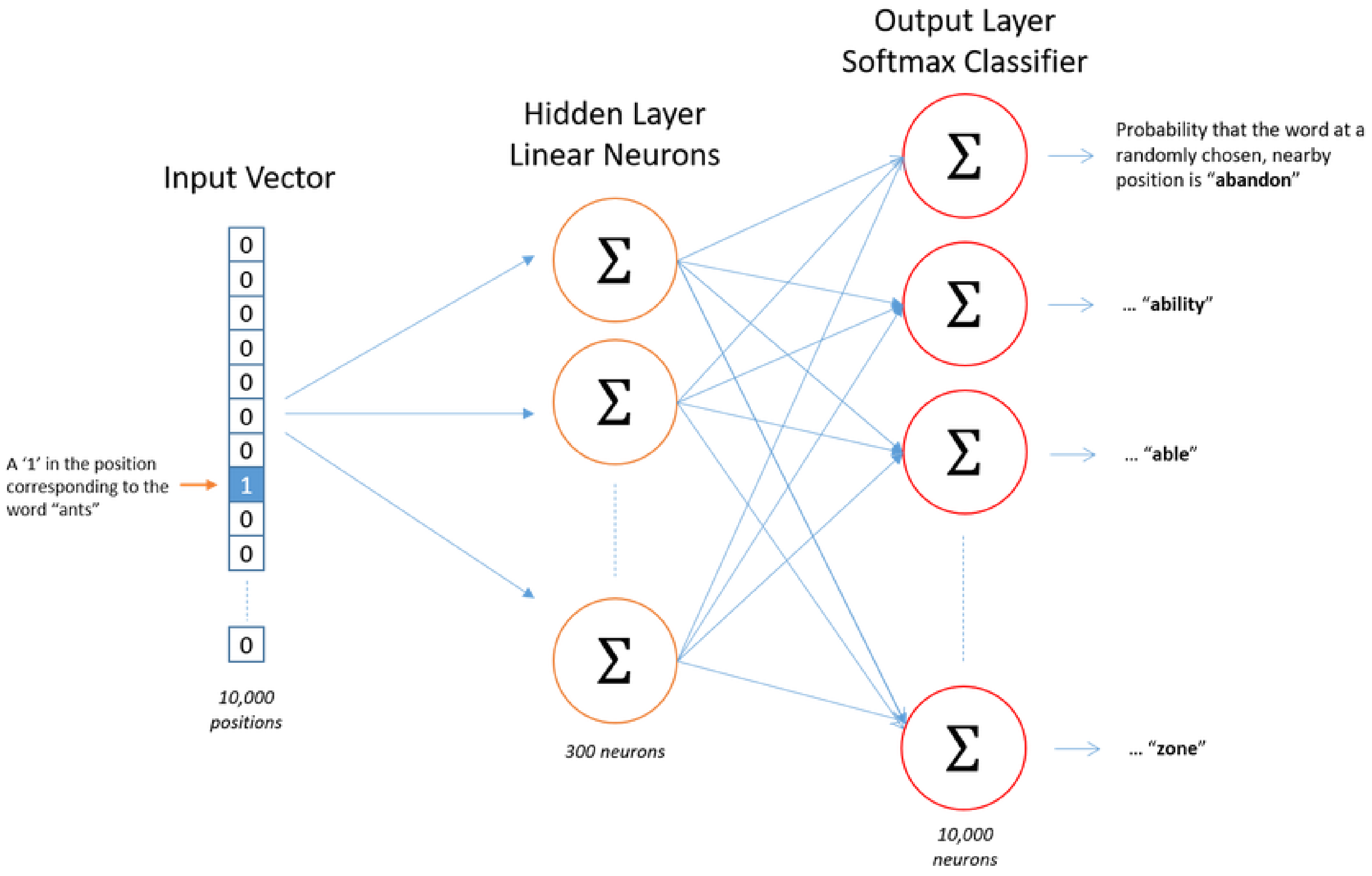
Motivation



word-context pair
(w,c)

Motivation

The architecture of Skip-Gram



Motivation

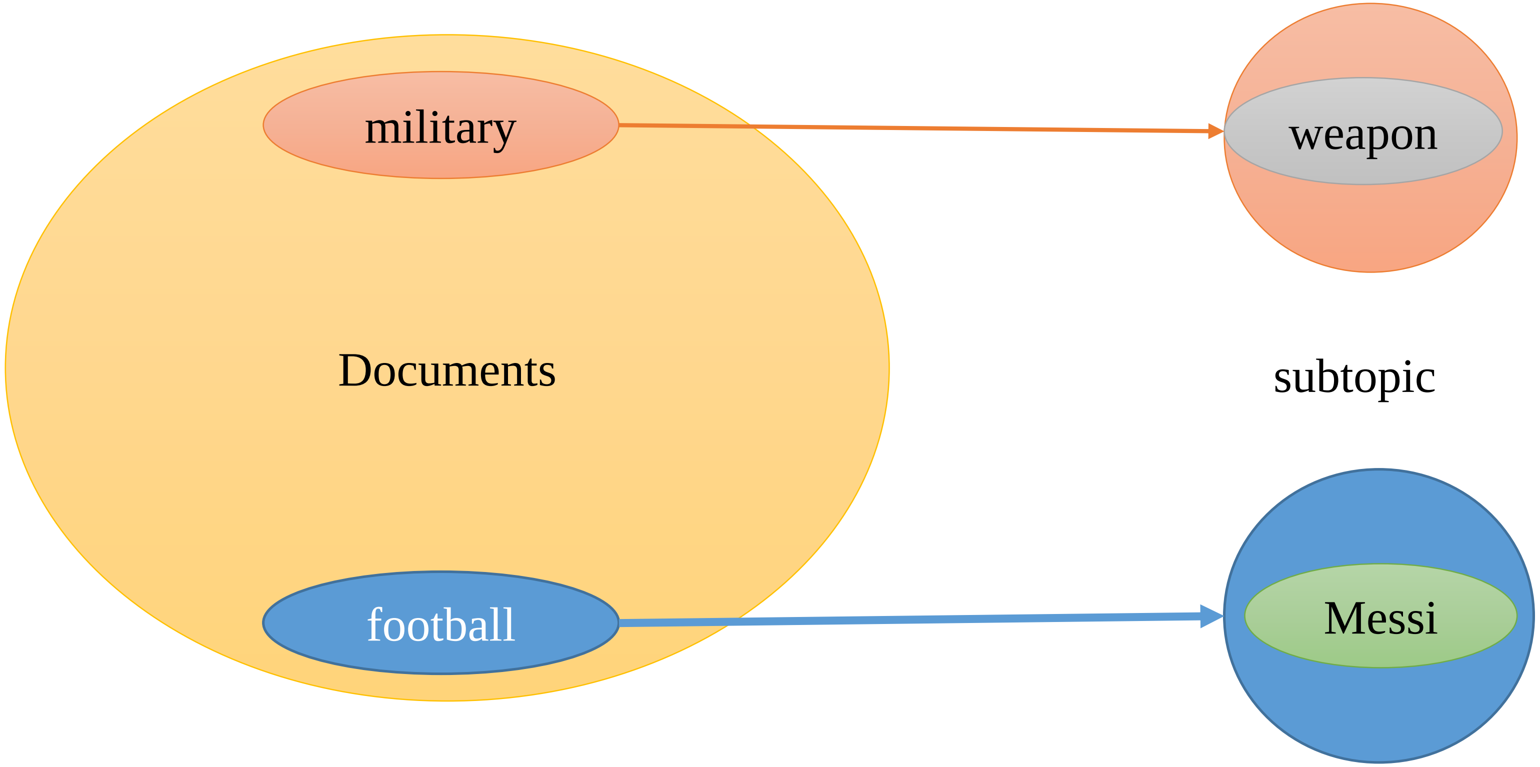
- Expected Loss:

$$\mathcal{L}_c = \mathbb{E}_{w,c\sim p_c}\left[\ell(w,c)\right]$$

- $l(w, c)$: loss of the instance given a word *w* and a context *c*
- p_c: the distribution of word-context pairs that can be estimated from corpus statistics

Motivation

- Documents on subtopics in a collection have very different unigram distributions compared to the whole corpus.



Motivation

- The language in a domain is more specialized. The distribution over word-context pairs is unlikely to be similar to $p_c(w,c)$

$$\mathcal{L}_t = \mathbb{E}_{w,c\sim p_c}\left[\frac{p_t(w,c)}{p_c(w,c)}\ell(w,c)\right]$$

- $p_t(w,c)$: probability of word-context pair observed under topic t
- if (w,c) occurs frequently in the topic, the loss of it will be amplified by

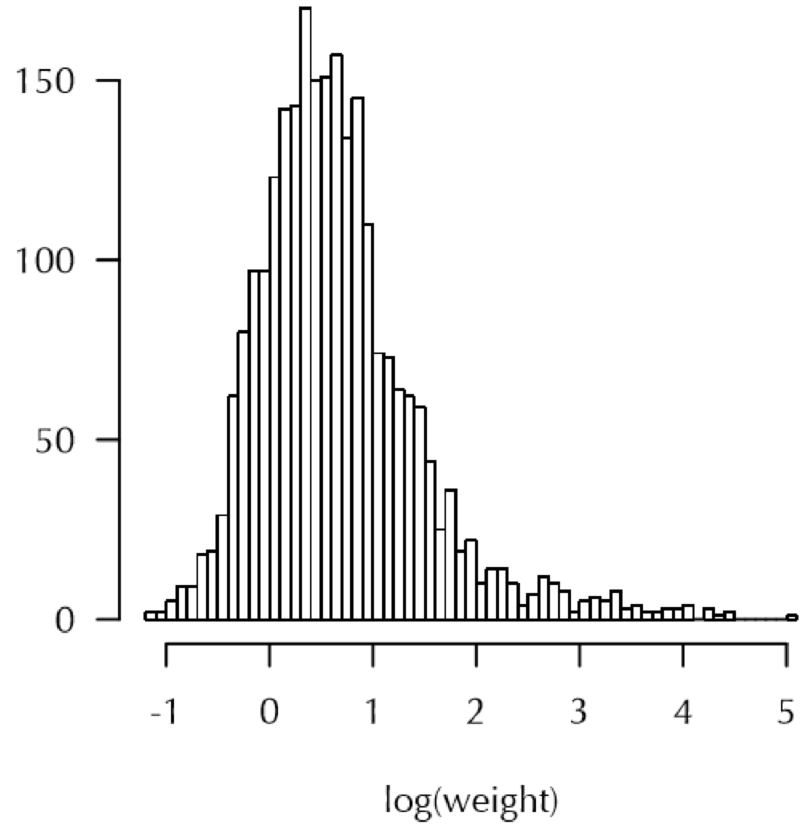
$$\omega = \frac{p_t(w,c)}{p_c(w,c)}$$

Motivation

- These contexts are likely to be underemphasized in

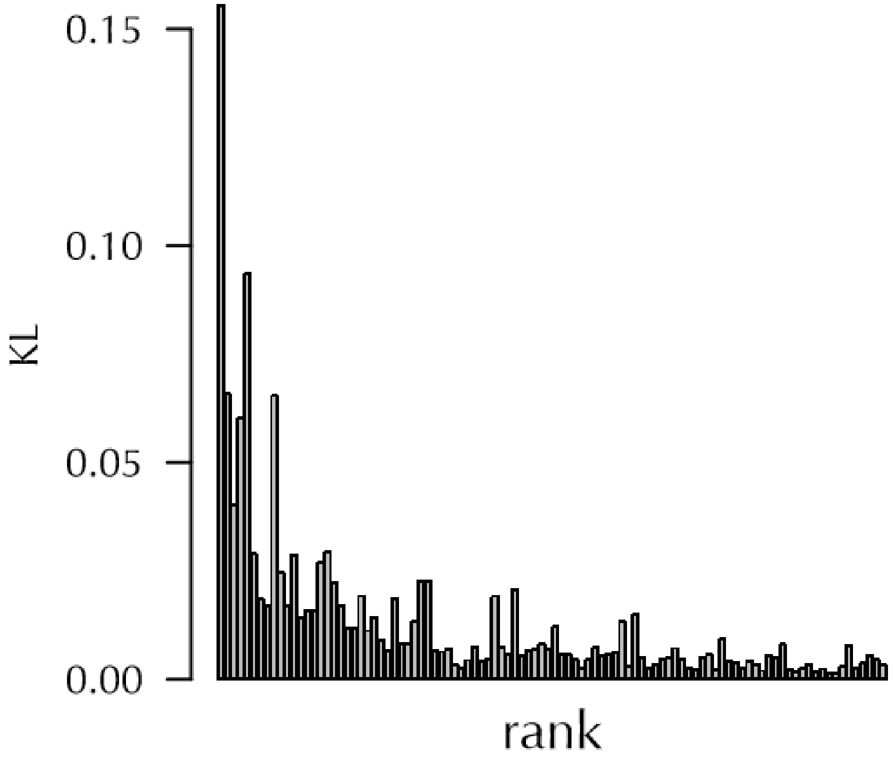
$$\mathcal{L}_c = \mathbb{E}_{w,c \sim p_c} \left[\ell(w, c) \right]$$

Motivation



- Importance weights for terms occurring in documents related to ‘argentina pegging dollar’ relative to frequency in gigaword.

Motivation



- Pointwise Kullback-Leibler divergence for terms occurring in documents related to ‘argentina pegging dollar’ relative to frequency in gigaword.

$$D_w(p_t \| p_c) = p_t(w) \log \frac{p_t(w)}{p_c(w)}$$

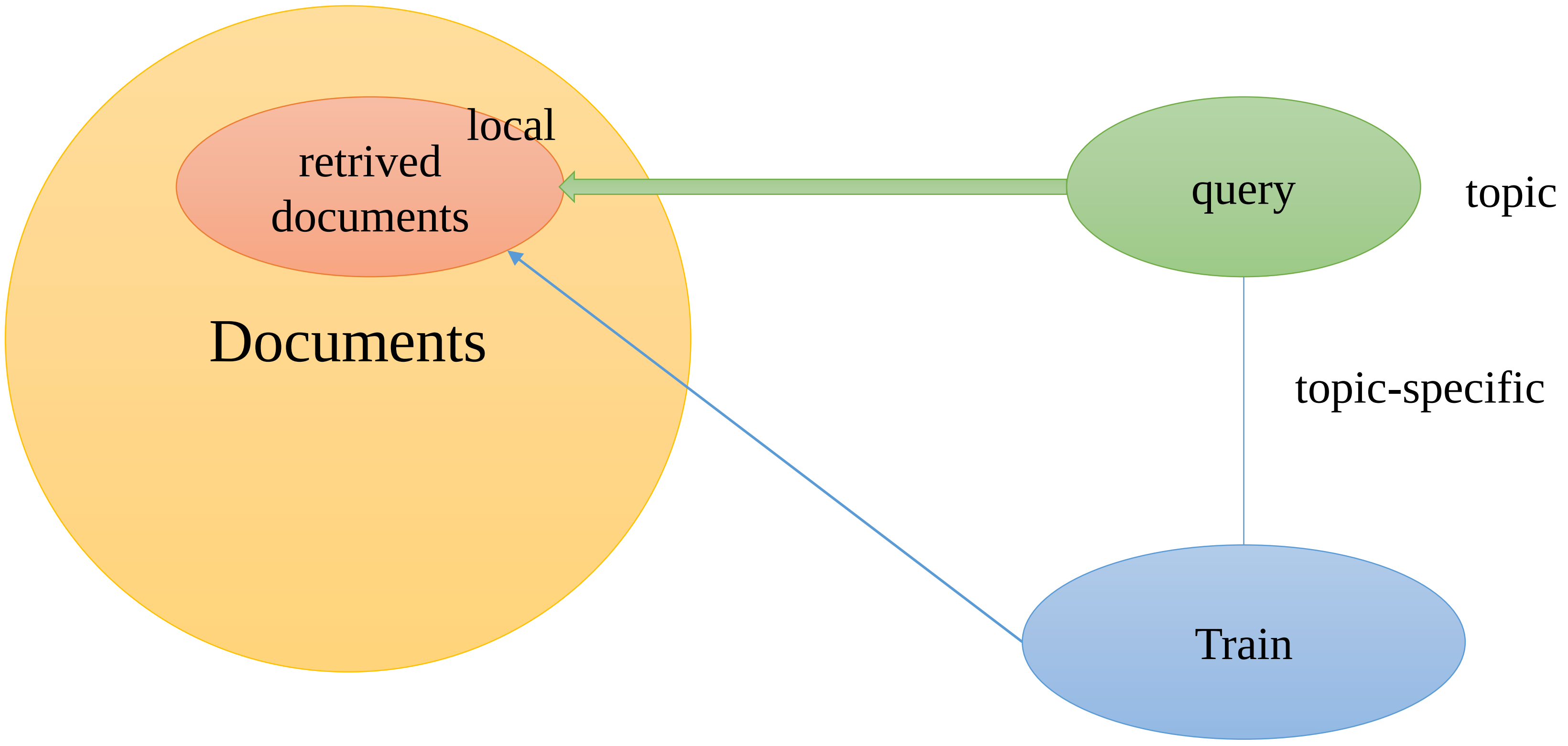
Motivaiton

- The higher ranked terms (i.e. good query expansion candidates) tend to have much higher probabilities than found in $p_c(w)$.
- If the loss on those words is large, this may result in poor embeddings for the most important words for the topic.

Local Word Embedding

- We need topic-specific word embeddings.
- In information retrieval scenarios users rarely provide the system with examples of topic-specific documents, instead providing a small set of keywords.
- How to generate a set of query-specific topical documents?

Idea



local
retrived
documents
Documents
query
topic
topic-specific
Train

Local Word Embedding

- Croft, W. Bruce, and J. Lafferty. *Language Modeling for Information Retrieval*.
- Each document is represented as a maximum likelihood language model estimated from document term frequencies. Query language models are estimated similarly.
- A document score is the KL-Divergence between query and documents.

Similar

$$D(p_q \| p_d) = \sum_{w \in \mathcal{V}} p_q(w) \log \frac{p_q(w)}{p_d(w)}$$

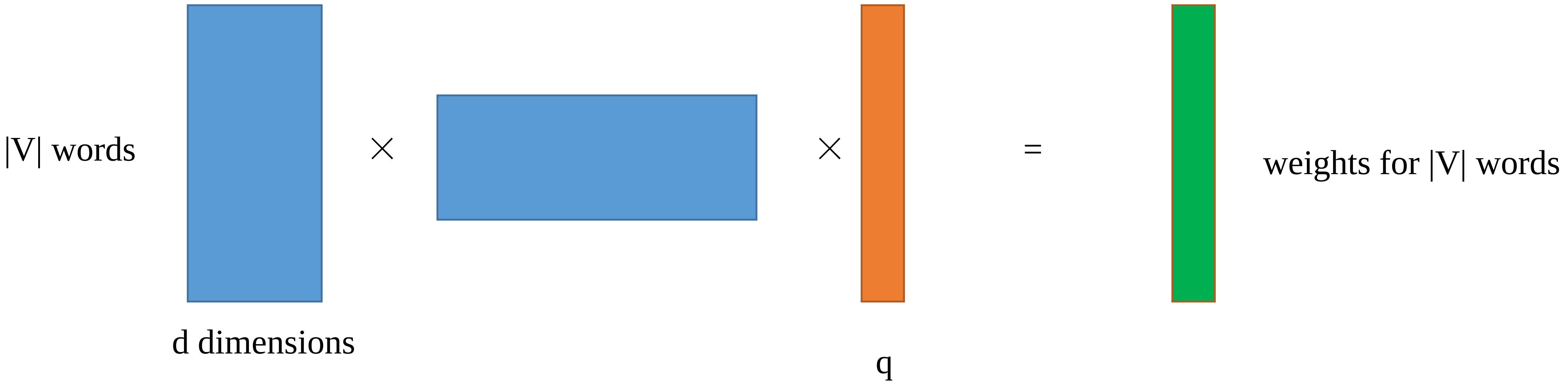
Local Word Embedding

- The scores can be passed through a softmax function to derive a multinomial over the entire corpus :

$$p(d) = \frac{\exp(-D(p_q \| p_d))}{\sum_{d'} \exp(-D(p_q \| p_{d'}))}$$

Query Expansion with Word Embedding

- U: |V| $\times$ d word embedding matrix
- **q:** |V| $\times$ 1 query vector (one-hot)
- $UU^T q$ = expansion term weights



Query Expansion with Word Embedding

- When each expansion term is associated with a weight, the language model of the query will be updated:

$$p_q^1(w) = \lambda p_q(w) + (1 - \lambda) p_{q^+}(w)$$

- p_{q^+}: the expansion language model by normalizing their weights.
- This interpolated language model can then be used with KL-Divergence to rank documents.

Data

- Trec12
- Robust
- ClueWeb2009 Category B
- Gigaword
- Wiki snapshot (December 2014)
- Pre-trained Word Embedding:
  - 4 GloVe embeddings of different dimensions
  - word2vec embedding trained on Google News

	docs	words	queries
trec12	469,949	438,338	150
robust	528,155	665,128	250
web	50,220,423	90,411,624	200
news	9,875,524	2,645,367	-
wiki	3,225,743	4,726,862	-

Local Embedding Training

- Train local embeddings by word2vec CBOW model on 3 retrieval sources:
- Documents retrieved from target corpus.
- Documents retrieved from auxiliary corpora(Gigaword).
  - provide more training data
- Documents retrieved from Wiki snapshot.
  - high fidelity corpus provide cleaner lauguage

Global Embedding Training

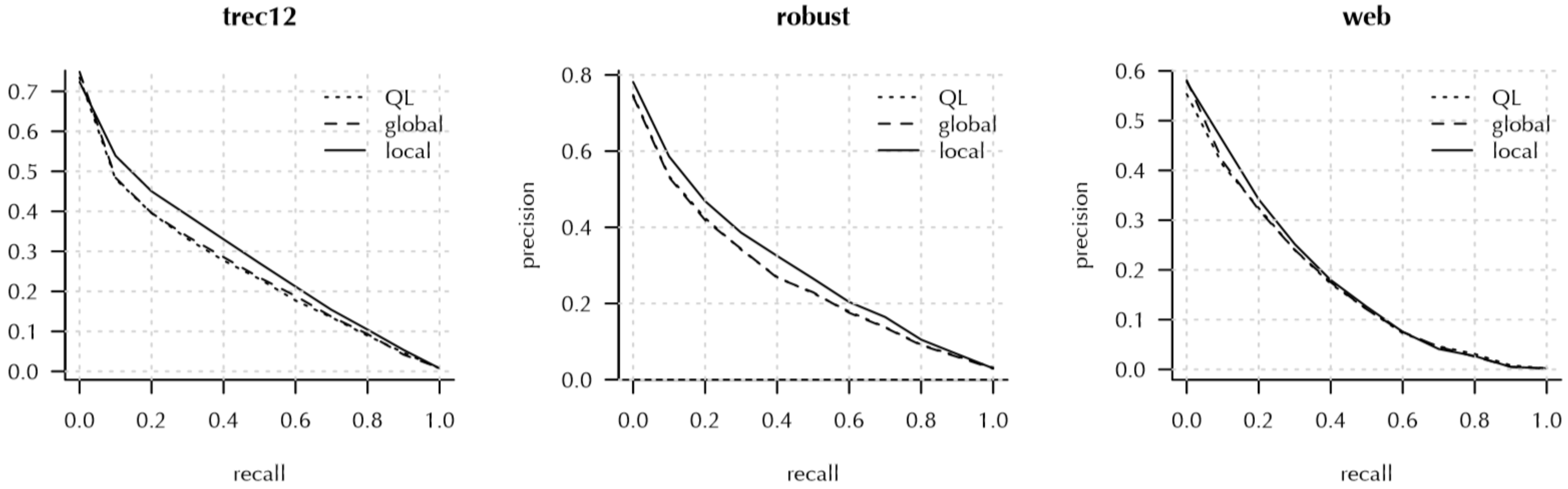
- Train global embeddings on:
  - The entire dataset of Trec12 and Robust
  - GloVe embedding(pre-trained) trained on Common Crawl data instead of ClueWeb

Results

		global						local		
		wiki+giga				gnews	target	target	giga	wiki
	QL	50	100	200	300	300	400	400	400	400
trec12	0.514	0.518	0.518	0.530	0.531	0.530	**0.545**	0.535	**0.563***	0.523
robust	0.467	0.470	0.463	0.469	0.468	**0.472**	0.465	0.475	**0.517***	0.476
web	0.216	0.227	0.229	0.230	**0.232**	0.218	0.216	0.234	0.236	**0.258***

QL: query likelihood model without query expansion

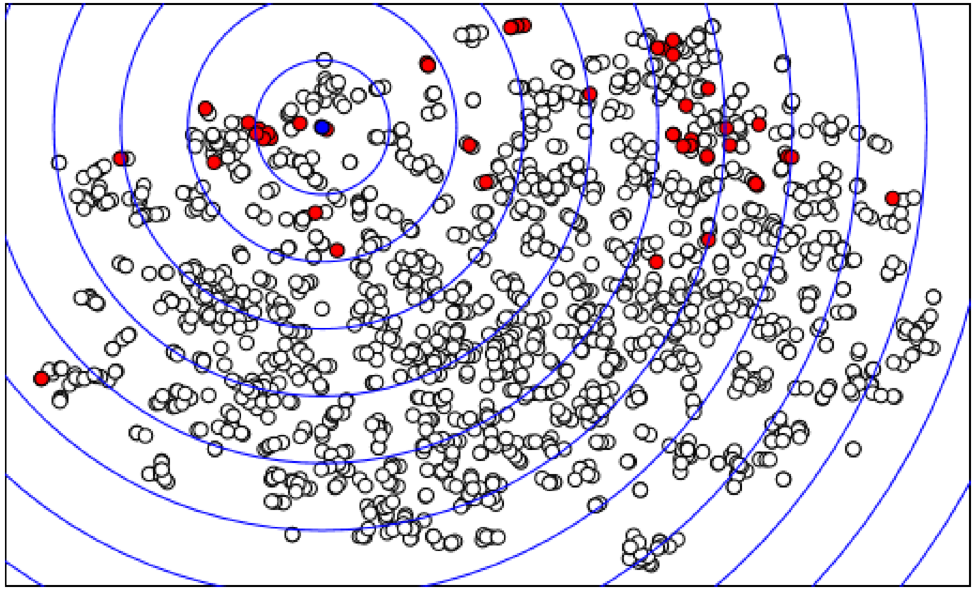
Results



Interpolated precision-recall curves for query likelihood, the best global embedding, and the best local embedding

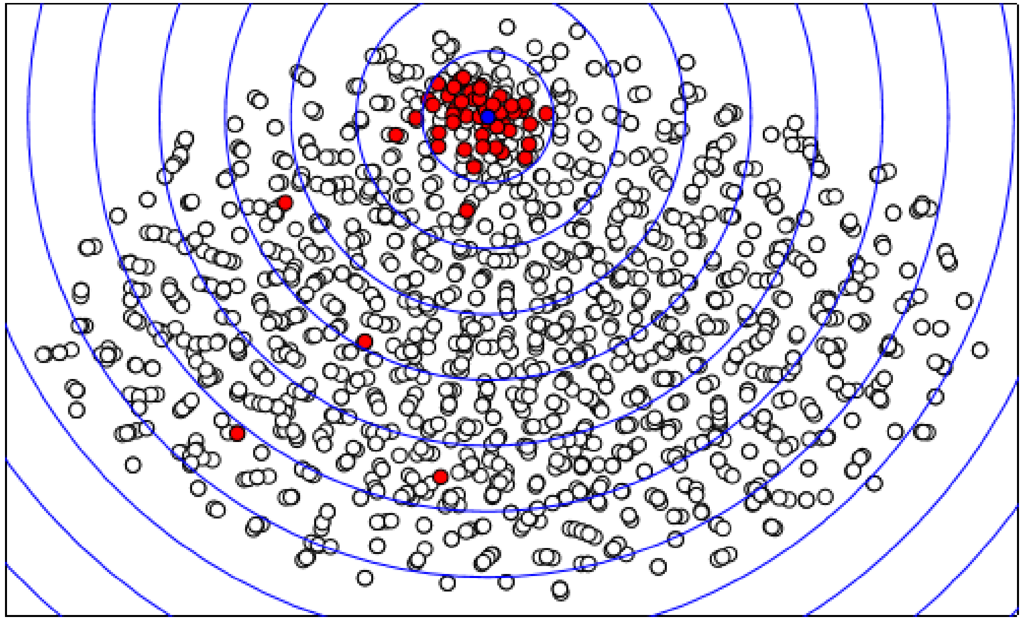
Results

Global



query “ocean remote sensing”

Local



Discussion

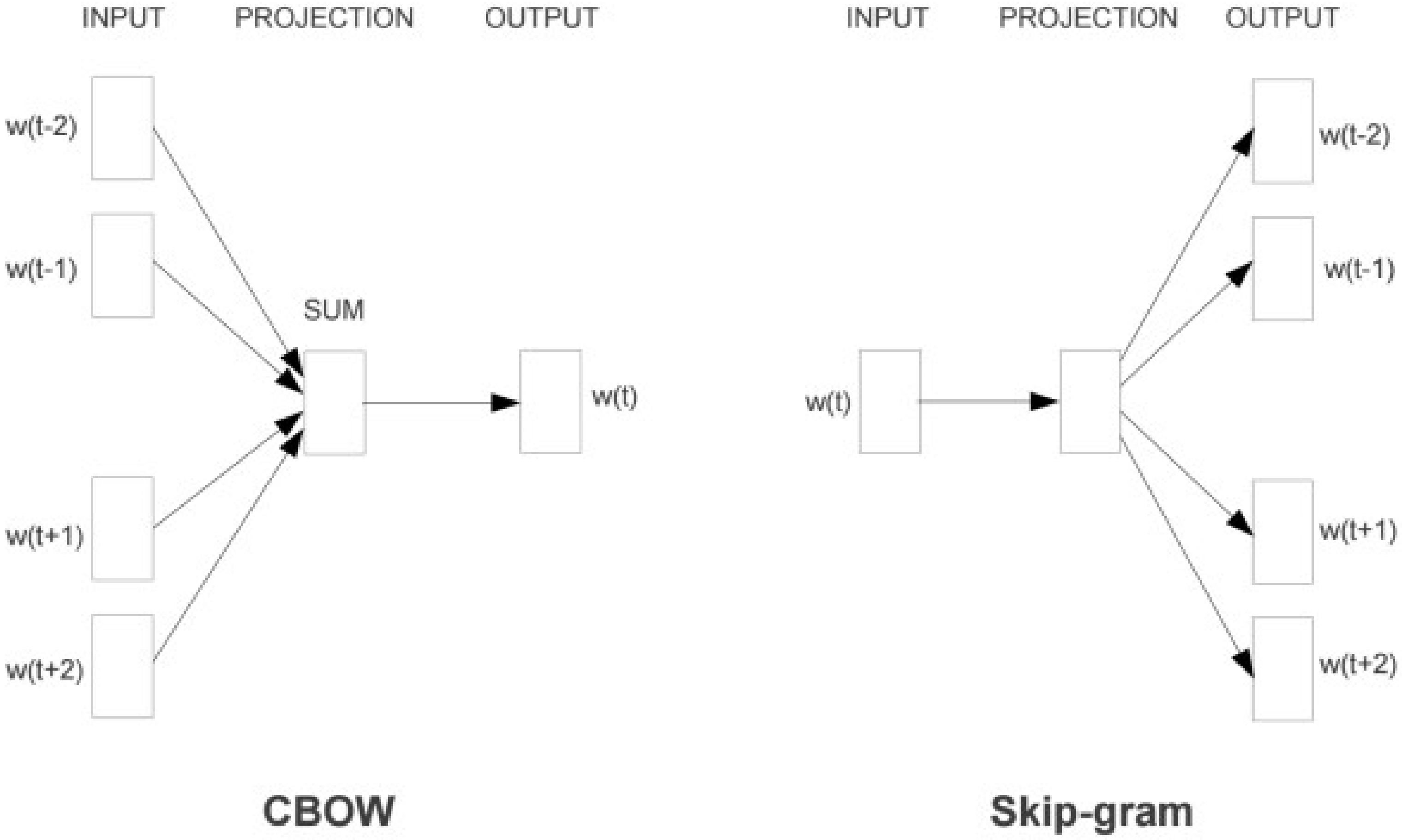
- The approach of learning from large scale of data is only effective if the data is appropriate for the task.
- Much smaller high-quality data can provide much better performance.
- Although local embeddings provide effectiveness gains, they can be quite inefficient compared to global embeddings.
- If the retrieval algorithm is able to select the appropriate embedding at query time, we can avoid training the local embedding.

Relevance-based Word Embedding

Hamed Zamani , W.Bruce Croft

SIGIR’17

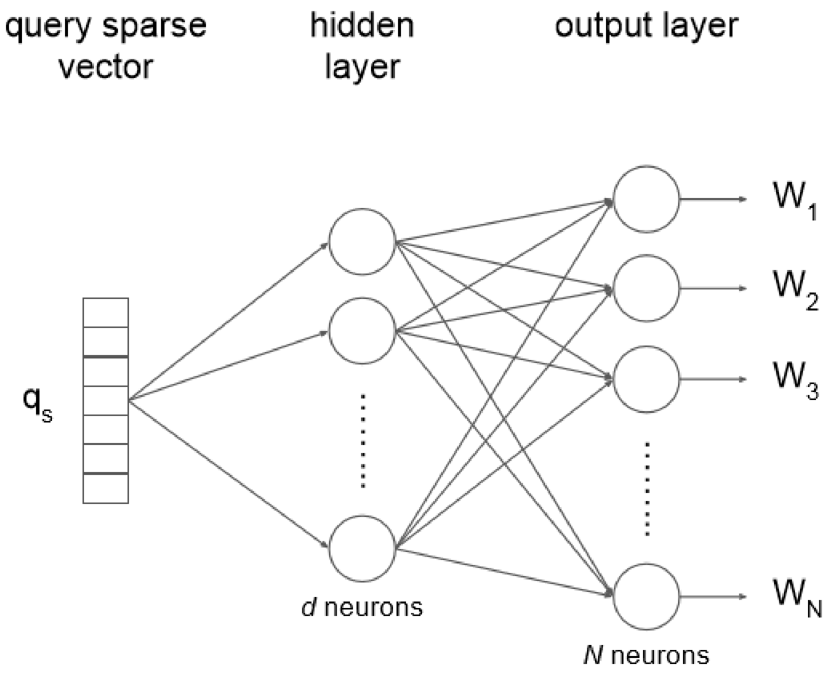
Skip-Gram & CBOW



Motivation

- The well-known word embedding algorithms, like word2vec or GloVe, aim to capture **semantic and syntactic similarities** between terms.
- The objective in IR tasks is to capture **relevance** instead of term proximity.
- Example: query expansion for “dangerous vehicle”
- word2vec: safe. (dangerous & safe share similar contexts)
- Our objective is to predict the terms that are observed in a set of relevant documents to a particular information need.

Architecture



$$\vec{q}_s = \frac{1}{|q|} \sum_{w \in q} \vec{e}_w$$

- $\vec{e}_w$: one-hot representation of term w
- $|q|$: query length

Architecture

- The hidden layer maps the given query sparse vector to a query embedding vector $\vec{q}$

$$\vec{q} = \vec{q}_s \times \mathcal{W}_Q$$

- W_Q : N×d matrix
- The output layer is a fully-connected layer:

$$\sigma(\vec{q} \times \mathcal{W}_w + b_w)$$

Modeling Relevance

- Given training set T={($q_{i,}R_i$)}, i=1 to m
- q_i the i-th query
- R_i the corresponding pseudo-relevance feedback distribution of q_i

$$p(w|\mathcal{R}_i) \propto \sum_{d \in F_i} p(w|d) \prod_{w' \in q_i} p(w'|d)$$

- F_i denotes the a set of top retrieved documents for query q_i

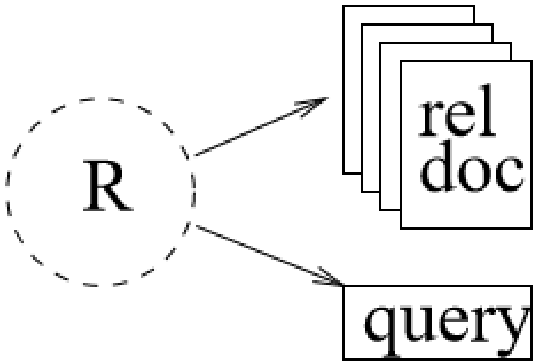
Relevance Model

- We can rank the documents D by the posterior probability they belong to relevant class R.
- It is equivalent to rank by odds:

$$\frac{P(D|R)}{P(D|N)} \sim \prod_{w \in D} \frac{P(w|R)}{P(w|N)}$$

- How to estimate P(w|R)?
- Lavrenko, Victor, and W. B. Croft. "Relevance based language models." *SIGIR'01*

Relevance Model



- Assumption: queries and relevant documents are random samples from an underlying relevance model R.

Relevance Model

- Let query Q=$q_1 \ldots q_k$.
- We have an unknown process R, a black box, from which we can repeatedly sample words.
- After k times we observe the words $q_1 \ldots q_k$.
- What is the probability that the next word we pull out of R will be w?

Relevance Model

- The only information is $q_1 \dots q_k$.
- So the best bet is related the probability to the observed sequence:

$$P(w|R) \approx P(w|q_1 \dots q_k)$$

- in terms of joint probability:

$$P(w|R) \approx \frac{P(w, q_1 \dots q_k)}{P(q_1 \dots q_k)}$$

- The challenge is to estimate the joint probability.

Relevance Model

- Let $\mathcal{M}$ represent the finite universe of unigram distributions from which we could sample.
- We pick a distribution $M \in \mathcal{M}$ with probability P(M) and sample k+1 times. Then the total probability of w and $q_1 \ldots q_k$ is :

$$P(w, q_1 \ldots q_k) = \sum_{M \in \mathcal{M}} P(M) P(w, q_1 \ldots q_k | M)$$

$$P(w, q_1 \ldots q_k | M) = P(w|M) \prod_{i=1}^{k} P(q_i|M)$$

$$P(w, q_1 \ldots q_k) = \sum_{M \in \mathcal{M}} P(M) P(w|M) \prod_{i=1}^{k} P(q_i|M)$$

Relevance Model

- look back to our task:
- $\mathcal{M}$ = Relevant docmuents

$$P(w|R) \approx \frac{P(w, q_1 \ldots q_k)}{P(q_1 \ldots q_k)}$$

$$p(w|\mathcal{R}_i) \propto \sum_{d \in F_i} p(w|d) \prod_{w' \in q_i} p(w'|d)$$

$$P(w, q_1 \ldots q_k) = \sum_{M \in \mathcal{M}} P(M) P(w|M) \prod_{i=1}^{k} P(q_i|M)$$

Relevance Likelihood Maximization

- Given a set of training data, we aim to find a set of parameters θR in order to maximize the likelihood of generating relevance model probabilities for the whole training set. likelihood function:

$$\prod_{i=1}^{m} \prod_{w \in V_i} \widehat{p}(w|q_i; \theta_{\mathcal{R}})^{p(w|\mathcal{R}_i)}$$

- V_i : a subset of vocabulary terms that appeared in the top ranked documents retrieved for the query qi

Relevance Likelihood Maximization

- The probability $\hat{p}$ can be estimated by softmax function:

$$\widehat{p}(w|q;\theta_{\mathcal{R}}) = \frac{\exp(\vec{w}^T \vec{q})}{\sum_{w' \in V} \exp(\vec{w'}^T \vec{q})}$$

σ

- $\vec{w}$: learned embedding vector of w,
- $\vec{q}$: output of the hidden layer
- Objective function:

$$\arg\max_{\theta_{\mathcal{R}}} \sum_{i=1}^{m} \sum_{w \in V_i} p(w|\mathcal{R}_i) \left(\log \exp(\vec{w}^T \vec{q}_i) - \log \sum_{w' \in V} \exp(\vec{w'}^T \vec{q}_i) \right)$$

Relevance Posterior Esitmation

- Assumption: the language model of the top retrieved documents is estimated based on a mixture model:
    - The relevance(topical) language model
    - The background noisy language model
- Estimating relevance distribution R = a classification task:
    - Given a pair of word *w* and query *q*, does *w* come from the relevance distribution of the query *q*?

Relevance Posterior Esitmation

- Binary classification – Logistic regression

$$\widehat{p}(R = 1|\vec{w}, \vec{q}; \theta_{\mathcal{R}}) = \frac{1}{1 + e^{(-\vec{w}^T \vec{q})}} \qquad \sigma$$

- Noise contrastive estimation: a good model can be obtained by only differentiating the data from noise via a logistic regression model.

Relevance Posterior Estimation

- Objective Function:

$$\arg\max_{\theta_{\mathcal{R}}} \sum_{i=1}^{m} \left[\sum_{j=1}^{\eta^{+}} \mathbb{E}_{w_j \sim p(w|\mathcal{R}_i)} \left[\log \widehat{p}(R = 1 | \vec{w}_j, \vec{q}_i ; \theta_{\mathcal{R}}) \right] + \sum_{j=1}^{\eta^{-}} \mathbb{E}_{w_j \sim p_n(w)} \left[\log \widehat{p}(R = 0 | \vec{w}_j, \vec{q}_i ; \theta_{\mathcal{R}}) \right] \right]$$

- $p_n(w)$: noise distribution. $p_n(w) \propto U(w)^{3/4}$
- $U(w)$： unigram distribution in the whole training set

Experiment

- Two tasks:
- 1. Query Expansion
- 2. Query Classification

Training

- Millions of unique queries from publicly available AOL query logs.
- Top-10 documents retrieved by query likelihood retrieval model
- SGD & BP
- Parameters:
  - batch size {64,128,256}
  - learning rate:{0.001,0.01,0.1,1}
  - η^{+}: {20,50,100,200}
  - η^{-}: {5,10,20}*η^{+}
  - dimensionality: 300

Evaluation via Query Expansion

- Data

Table 1: Collections statistics.

ID	collection	queries (title only)	#docs	avg doc length	#qrels
AP	Associated Press 88-89	TREC 1-3 Ad-Hoc Track, topics 51-200	165k	287	15,838
Robust	TREC Disks 4 & 5 minus Congressional Record	TREC 2004 Robust Track, topics 301-450 & 601-700	528k	254	17,412
GOV2	2004 crawl of .gov domains	TREC 2004-2006 Terabyte Track, topics 701-850	25m	648	26,917
ClueWeb	ClueWeb 09 - Category B	TREC 2009-2012 Web Track topics 1-200	50m	1506	18,771

Experiment Setup

$$p(w|\theta_q^*) = \alpha p_{ML}(w|q) + (1-\alpha)p(\vec{w}|\vec{q})$$

- $p_{ML}(w|q) = \frac{c(w)}{|q|}$
- $p(\vec{w}|\vec{q}) = ?$

RLM

RPE

$$\frac{\exp(\vec{w}^T\vec{q})}{\sum_{w' \in V}\exp(\vec{w'}^T\vec{q})}$$

$$\frac{1}{1+e^{(-\vec{w}^T\vec{q})}}$$

Results

Collection	Metric	MLE	word2vec		GloVe		Rel.-based Embedding	
			external	target	external	target	RLM	RPE
AP	MAP	0.2197	0.2399	0.2420	0.2319	0.2389	**0.2580**^{01234}	0.2543^{01234}
	P@20	0.3503	0.3688	0.3738	0.3581	0.3631	**0.3886**^{01234}	0.3812^{034}
	NDCG@20	0.3924	0.4030	0.4181	0.4025	0.4098	**0.4242**^{01234}	0.4226^{01234}
Robust	MAP	0.2149	0.2218	0.2215	0.2209	0.2172	**0.2450**^{01234}	0.2372^{01234}
	P@20	0.3319	0.3357	0.3337	0.3345	0.3281	**0.3476**^{01234}	0.3409^{024}
	NDCG@20	0.3863	0.3918	0.3881	0.3918	0.3844	**0.3982**^{01234}	0.3955^{0}
GOV2	MAP	0.2702	0.2740	0.2723	0.2718	0.2709	**0.2867**^{01234}	0.2855^{01234}
	P@20	0.5132	0.5257	0.5172	0.5186	0.5128	**0.5367**^{01234}	0.5358^{01234}
	NDCG@20	0.4482	0.4571	0.4509	0.4539	0.4485	**0.4576**^{0234}	0.4557^{024}
ClueWeb	MAP	0.1028	0.1033	0.1033	0.1029	0.1026	**0.1066**^{01234}	0.1031
	P@20	0.3025	0.3040	0.3053	0.3033	0.3048	**0.3073**	0.3030
	NDCG@20	0.2237	0.2235	0.2252	0.2244	0.2244	**0.2273**^{01}	0.2241

Results

query: "indian american museum"				query: "tibet protesters"			
word2vec		Rel.-based Embedding		word2vec		Rel.-based Embedding	
external	target	RLM	RPE	external	target	RLM	RPE
history	powwows	chumash	heye	demonstrators	tibetan	tibetan	tibetan
art	smithsonian	heye	collection	protestors	lhasa	lama	tibetans
culture	afro	artifacts	chumash	tibetan	demonstrators	tibetans	lama
british	mesoamerica	smithsonian	smithsonian	protests	tibetans	lhasa	independence
heritage	smithsonians	collection	york	tibetans	marchers	dalai	lhasa
society	native	washington	new	protest	lhasas	independence	dalai
states	heye	institution	apa	activists	jokhang	protest	open
contemporary	hopi	york	native	protesting	demonstrations	open	protest
part	mayas	native	americans	lhasa	dissidents	zone	zone
united	cimam	apa	history	demonstrations	barkhor	followers	jokhang

Top-10 expansion terms obtained by word2vec and rel-based word embedding model for two sample queries.

Results

Collection	Metric	RM3	Local Emb.	ERM	
				Local	RLM
AP	MAP	0.2927	0.2412	0.3047	**0.3119**[12]
	P@20	0.4034	0.3742	0.4105	**0.4233**[12]
	NDCG@20	0.4368	0.4173	0.4411	**0.4495**[123]
Robust	MAP	0.2593	0.2235	0.2643	**0.2761**[123]
	P@20	0.3486	0.3366	0.3498	**0.3605**[123]
	NDCG@20	0.4011	0.3868	0.4080	**0.4173**[123]
GOV2	MAP	0.2863	0.2748	0.2924	**0.2986**[123]
	P@20	0.5318	0.5271	0.5379	**0.5417**[12]
	NDCG@20	0.4503	0.4576	0.4584	**0.4603**[123]
ClueWeb	MAP	0.1079	0.1041	0.1094	**0.1121**[12]
	P@20	0.3111	0.3062	0.3145	**0.3168**
	NDCG@20	0.2309	0.2261	0.2328	**0.2360**[2]

Evaluation via Query Classification

- Data: KDD Cup 2005 , user search query categorization task.
- 800 queries. 67 categories were pre-defined and up to 5 labels were selected for each query.
- Train rel-based embedding on Robust collection

Classify

- compute the probability of each category/label given each query q
- select the top t categories with the highest probabilities

$$p(C_i|q) = \frac{\delta(\vec{C_i}, \vec{q})}{\sum_j \delta(\vec{C_j}, \vec{q})} \propto \delta(\vec{C_i}, \vec{q})$$

- C_i : i-th label
- $\overrightarrow{C_i}$: centroid vector

Similarity Function δ

$$\delta(\vec{w}, \vec{w'}) = \exp\left(\frac{\sum_{i=1}^{d} \vec{w}_i \vec{w'}_i}{\|\vec{w}\| \|\vec{w'}\|} \right)$$

$$\delta(\vec{w}, \vec{w'}) = \frac{1}{1 + \exp\left(-a \left(\frac{\sum_{i=1}^{d} \vec{w}_i \vec{w'}_i}{\|\vec{w}\| \|\vec{w'}\|} - c \right) \right)}$$

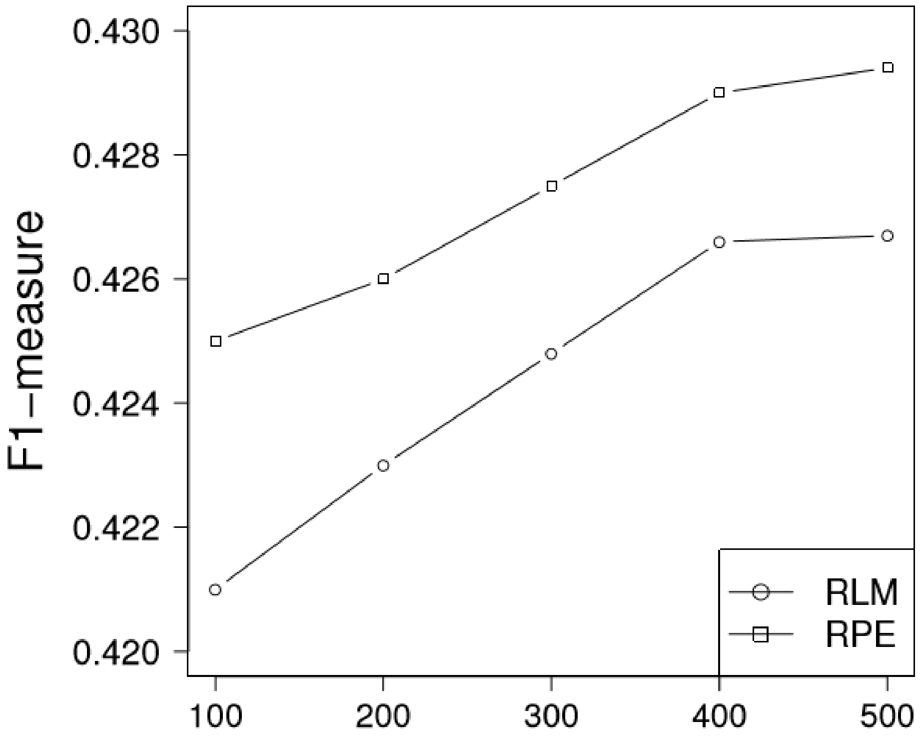
- Zamani, Hamed, and W. B. Croft. "Estimating Embedding Vectors for Queries." *ICTIR'16*

Results

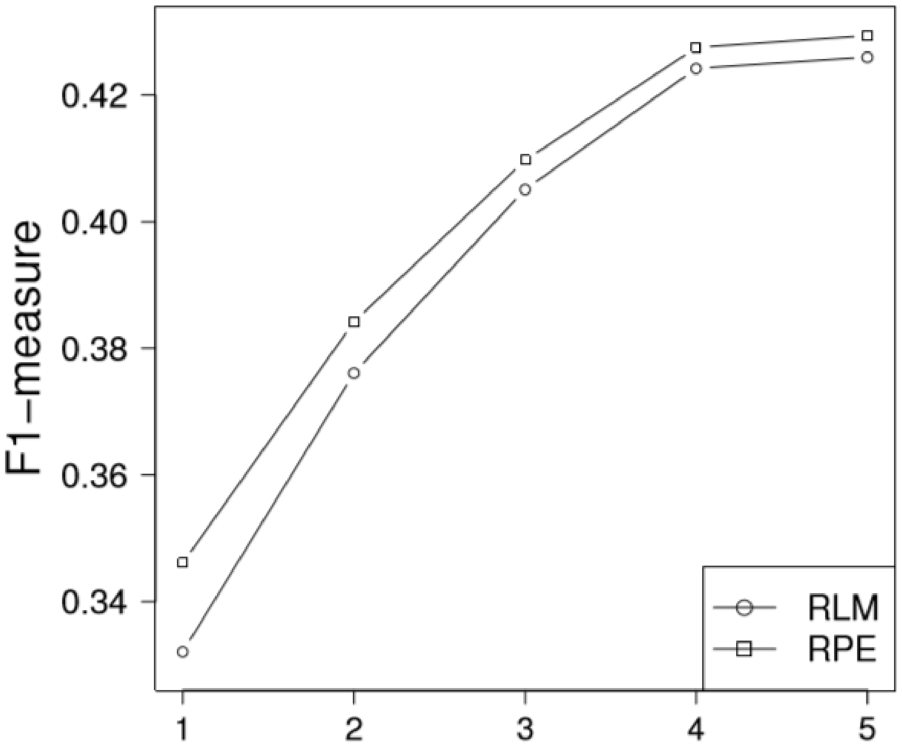
Method	Precision	F1-measure
word2vec	0.3712	0.4008
GloVe	0.3643	0.3912
Rel.-based Embedding - RLM	0.3943[12]	0.4267[12]
Rel.-based Embedding - RPE	**0.3961**[12]	**0.4294**[12]

Results

dimensionality



number of queries(million)



Thanks!