Text Mining
(Part II: Embedding, Information Extraction & Text Cube)

JIAWEI HAN
COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

MARCH 30, 2017

Outline

- Learning Embeddings in Networks and Text
  - LINE: Large-scale Information Network Embedding
  - Biological Relationship Discovery with Network Embedding
- LAKI: Representing Documents via Latent Keyphrase Inference
- MetaPAD: Meta Pattern Discovery from Massive Text Corpora
- TextCube, EventCube and CaseOLAP
- Summary

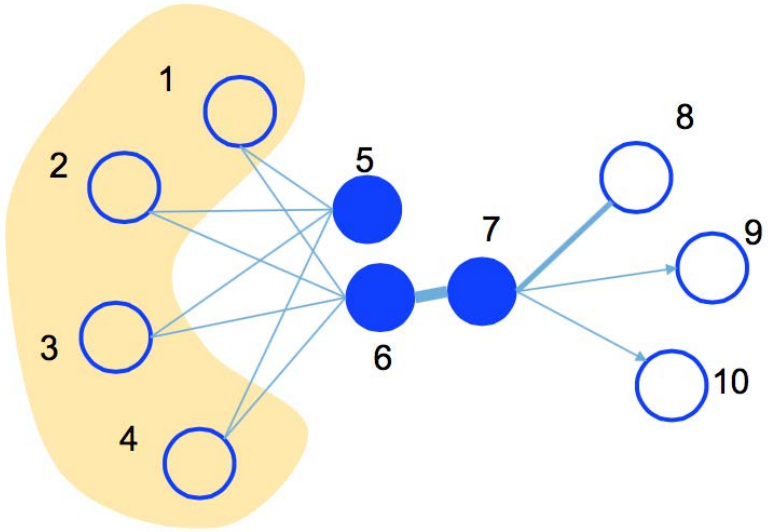
Information Network Analysis

- A lot of interesting problems
  - Node classification
  - Link prediction
  - Visualization

- Input of most algorithms
  - Low-dimensional feature vector of every vertex
- How to get the feature vectors?
  - From the context information of vertices 
  - From the network structure 
- How?

LINE and Its Hypothesis

- LINE paper: J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, “LINE: Large-scale information network embedding”, WWW'15
- Nodes with strong ties turn to be similar
  - 1st order similarity
- Nodes share many neighbors turn to be similar
  - 2nd order similarity
- **Well-learnt embedding should preserve both 1st order and 2nd order similarity**



Nodes 6 & 7: high 1st order similarity

Nodes 5 & 6: high 2nd order similarity

LINE: 1st Order Similarity

- Try to capture local pairwise proximity between the nodes (vertices) in the network
  - Vertices that are connected with larger weight are more similar
- Embedding vector for each vertex: $\vec{v}$
- To model the 1st order proximity, for each undirected edge (*i*, *j*), we define the joint probability between vertex v_i and v_j as $p_1(v_i, v_j) = \frac{1}{1 + \exp(-\vec{u}_i^T \cdot \vec{u}_j)}$
  - where $\vec{u}_i \in R^d$ is the low-dimensional vector representation of vertex v_i
  - *Joint probability* between vertex v_i and v_j : $P(v_i, v_j) = \frac{\exp(\vec{v_i} \cdot \vec{v_j})}{\sum_{i,j} \exp(\vec{v_i} \cdot \vec{v_j})}$
  - *Empirical joint probability* between vertex v_i and v_j : $\hat{P}(v_i, v_j) = \frac{w_{ij}}{\sum_{i,j} w_{ij}}$
- Objective: Minimize the KL-divergence of two probability distributions
  - $O_1 = KL(\hat{P}||P) = -\sum_{i,j} w_{ij} log P(v_i, v_j)$

KL Divergence: Comparing Two Probability Distributions

- *The Kullback-Leibler (KL) divergence:* Measure the difference between two probability distributions over the same variable *x*
  - From information theory, closely related to *relative entropy*, *information divergence*, and *information for discrimination*
- $D_{KL}(p(x) \,||\, q(x))$: divergence of $q(x)$ from $p(x)$, measuring the information lost when $q(x)$ is used to approximate $p(x)$
  - Discrete form:

$$D_{KL}(p(x)||q(x)) = \sum_{x \in X} p(x) \ln \frac{p(x)}{q(x)}$$

- The KL divergence measures the expected number of extra bits required to code samples from $p(x)$ ("true" distribution) when using a code based on $q(x)$, which represents a theory, model, description, or approximation of $p(x)$
- Its continuous form:

$$D_{KL}(p(x)||q(x)) = \int_{-\infty}^{\infty} p(x) \ln \frac{p(x)}{q(x)} dx$$

- Not a distance measure, not a metric: asymmetric, not satisfy triangular inequality

LINE: 2nd Order Similarity

- 2nd order similarity: proximity between two vertices (u, v) is the similarity between their neighborhood network structures
- Each vertex is treated as a specific context, and vertices with similar distributions over the contexts are assumed to be similar
- Introduce two vectors for each vertex
  - Embedding vector: $\vec{v}$, and context vector: $\vec{v}'$
- The *probability of "context"* v_j generated by vertex v_i and its *empirical probability* as,

$$P_i(v_j|v_i) = \frac{\exp(\overrightarrow{v_i} \cdot \overrightarrow{v_j}')}{\sum_{i,j} \exp(\overrightarrow{v_i} \cdot \overrightarrow{v_j}')} \qquad \widehat{P}_i(v_j|v_i) = \frac{w_{ij}}{\sum_j w_{ij}}$$

- Minimizing the objective function:
  - $O_2 = \sum_i d_i KL(\widehat{P}_i || P_i) = -\sum_{i,j} w_{ij} log P_i(v_j|v_i)$

Combining LINE 1st and LINE 2nd Order Similarity

- Concatenate the embedding learnt by LINE 1st and that learnt by LINE 2nd
  - 1st order similarity and 2nd order similarity are complementary
- Jointly optimize O_1 and O_2
  - Difficult to determine the weights of the two objective functions
- **Optimization**
  - Directly optimizing the objective function is difficult
    - Softmax units
  - Negative sampling:
    - Approximation algorithm
    - Can retain the quality of the embedding
      - $O_1 = \sum_{i,j} \hat{P}_{ij} log\sigma(\overrightarrow{v_i} \cdot \overrightarrow{v_j}) + K \sum_{i,j} \hat{P}_i \hat{P}_j log\sigma(-\overrightarrow{v_i} \cdot \overrightarrow{v_j})$
      - $O_2 = \sum_{i,j} \hat{P}_{ij} log\sigma(\overrightarrow{v_i} \cdot \overrightarrow{v_j}') + K \sum_{i,j} \hat{P}_i \hat{P}_j log\sigma(-\overrightarrow{v_i} \cdot \overrightarrow{v_j}')$

Algorithm LINE and Some Practical Issues

- LINE:
  - Train the LINE model which preserves the *first-order* proximity and *second-order* proximity separately
  - Then concatenate the embeddings trained by the two methods for each vertex
- A more principled way (future work)
  - Jointly train the two objective functions
- Implementation consideration:
  - Information networks can be very sparse: Reconstruct them to make them denser
    - Strategy: Add links between every vertex and its high order neighbors
    - Add second order neighbors
      - $w_{ij} = \sum_{k \in N(i)} w_{ik} \frac{w_{kj}}{d_k}$

Previous Work

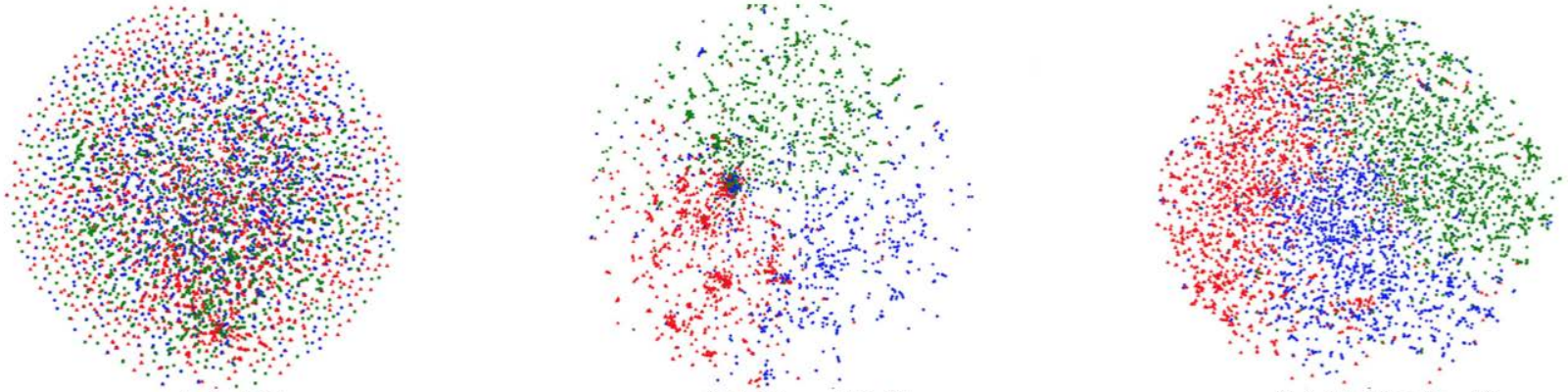
- Graph Embedding
  - Construct an affinity matrix & compute eigenvectors of the affinity matrix
  - Weakness: High time complexity and space complexity, and
    - Only preserve 1st order similarity
- Matrix Factorization
  - Objective: $\sum_{i,j}(x_i^T x_j - W_{ij})^2$ Optimization: SGD (Stochastic Gradient Descent Alg.)
  - Weakness: Can't converge when the range of weights is very large
    - Only preserve 1st order similarity
- Deep Walk (Perozzi, et al., KDD'14)
  - Sample a chain of vertex by truncated random walk
  - Treat the chain as a sentence and run word2vec on this sentence
  - Weakness: Only preserve 2nd order similarity

Experiment Setup

- Dataset

	Language Network	Social Network		Citation Network	
Name	WIKIPEDIA	FLICKR	YOUTUBE	DBLP(AUTHORCITATION)	DBLP(PAPERCITATION)
Type	undirected,weighted	undirected,binary	undirected,binary	dircted,weighted	directed,binary
|V|	1,985,098	1,715,256	1,138,499	524,061	781,109
|E|	1,000,924,086	22,613,981	2,990,443	20,580,238	4,191,677
Avg. degree	504.22	26.37	5.25	78.54	10.73
#Labels	7	5	47	7	7
#train	70,000	75,958	31,703	20,684	10,398

- Task
  - Word analogy: Evaluated on Accuracy
  - Document classification: Evaluated on Macro-F1 Micro-F1
  - Vertex classification: Evaluated on Macro-F1 Micro-F1
- Result visualization



(a) GF (b) DeepWalk (c) LINE(2nd)

Results: Language Networks

- Word Analogy
  - GF (Graph Factorization) Ahmed et al., WWW2013)

Algorithm	Semantic (%)	Syntactic (%)	Overall (%)	Running time
GF	61.38	44.08	51.93	2.96h
DeepWalk	50.79	37.70	43.65	16.64h
SkipGram	69.14	57.94	63.02	2.82h
LINE-SGD(1st)	9.72	7.48	8.50	3.83h
LINE-SGD(2nd)	20.42	9.56	14.49	3.94h
LINE(1st)	58.08	49.42	53.35	2.44h
LINE(2nd)	**73.79**	**59.72**	**66.10**	2.55h

- Document Classification

Metric	Algorithm	10%	20%	30%	40%	50%	60%	70%	80%	90%
Micro-F1	GF	79.63	80.51	80.94	81.18	81.38	81.54	81.63	81.71	81.78
	DeepWalk	78.89	79.92	80.41	80.69	80.92	81.08	81.21	81.35	81.42
	SkipGram	79.84	80.82	81.28	81.57	81.71	81.87	81.98	82.05	82.09
	LINE-SGD(1st)	76.03	77.05	77.57	77.85	78.08	78.25	78.39	78.44	78.49
	LINE-SGD(2nd)	74.68	76.53	77.54	78.18	78.63	78.96	79.19	79.40	79.57
	LINE(1st)	79.67	80.55	80.94	81.24	81.40	81.52	81.61	81.69	81.67
	LINE(2nd)	79.93	80.90	81.31	81.63	81.80	81.91	82.00	82.11	82.17
	LINE(1st+2nd)	**81.04****	**82.08****	**82.58****	**82.93****	**83.16****	**83.37****	**83.52****	**83.63****	**83.74****
Macro-F1	GF	79.49	80.39	80.82	81.08	81.26	81.40	81.52	81.61	81.68
	DeepWalk	78.78	79.78	80.30	80.56	80.82	80.97	81.11	81.24	81.32
	SkipGram	79.74	80.71	81.15	81.46	81.63	81.78	81.88	81.98	82.01
	LINE-SGD(1st)	75.85	76.90	77.40	77.71	77.94	78.12	78.24	78.29	78.36
	LINE-SGD(2nd)	74.70	76.45	77.43	78.09	78.53	78.83	79.08	79.29	79.46
	LINE(1st)	79.54	80.44	80.82	81.13	81.29	81.43	81.51	81.60	81.59
	LINE(2nd)	79.82	80.81	81.22	81.52	81.71	81.82	81.92	82.00	82.07
	LINE(1st+2nd)	**80.94****	**81.99****	**82.49****	**82.83****	**83.07****	**83.29****	**83.42****	**83.55****	**83.66****

Significantly outperforms GF at the: ** 0.01 and * 0.05 level, paired t-test.

Results: Social Networks

- Flickr dataset

Metric	Algorithm	10%	20%	30%	40%	50%	60%	70%	80%	90%
Micro-F1	GF	53.23	53.68	53.98	54.14	54.32	54.38	54.43	54.50	54.48
	DeepWalk	60.38	60.77	60.90	61.05	61.13	61.18	61.19	61.29	61.22
	DeepWalk(256dim)	60.41	61.09	61.35	61.52	61.69	61.76	61.80	61.91	61.83
	LINE(1st)	63.27	63.69	63.82	63.92	63.96	64.03	64.06	64.17	64.10
	LINE(2nd)	62.83	63.24	63.34	63.44	63.55	63.55	63.59	63.66	63.69
	LINE(1st+2nd)	**63.20****	**63.97****	**64.25****	**64.39****	**64.53****	**64.55****	**64.61****	**64.75****	**64.74****
Macro-F1	GF	48.66	48.73	48.84	48.91	49.03	49.03	49.07	49.08	49.02
	DeepWalk	58.60	58.93	59.04	59.18	59.26	59.29	59.28	59.39	59.30
	DeepWalk(256dim)	59.00	59.59	59.80	59.94	60.09	60.17	60.18	60.27	60.18
	LINE(1st)	62.14	62.53	62.64	62.74	62.78	62.82	62.86	62.96	62.89
	LINE(2nd)	61.46	61.82	61.92	62.02	62.13	62.12	62.17	62.23	62.25
	LINE(1st+2nd)	**62.23****	**62.95****	**63.20****	**63.35****	**63.48****	**63.48****	**63.55****	**63.69****	**63.68****

Significantly outperforms DeepWalk at the: ** 0.01 and * 0.05 level, paired t-test.

- Youtube dataset

Metric	Algorithm	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
Micro-F1	GF	25.43 (24.97)	26.16 (26.48)	26.60 (27.25)	26.91 (27.87)	27.32 (28.31)	27.61 (28.68)	27.88 (29.01)	28.13 (29.21)	28.30 (29.36)	28.51 (29.63)
	DeepWalk	39.68	41.78	42.78	43.55	43.96	44.31	44.61	44.89	45.06	45.23
	DeepWalk(256dim)	39.94	42.17	43.19	44.05	44.47	44.84	45.17	45.43	45.65	45.81
	LINE(1st)	35.43 (36.47)	38.08 (38.87)	39.33 (40.01)	40.21 (40.85)	40.77 (41.33)	41.24 (41.73)	41.53 (42.05)	41.89 (42.34)	42.07 (42.57)	42.21 (42.73)
	LINE(2nd)	32.98 (36.78)	36.70 (40.37)	38.93 (42.10)	40.26 (43.25)	41.08 (43.90)	41.79 (44.44)	42.28 (44.83)	42.70 (45.18)	43.04 (45.50)	43.34 (45.67)
	LINE(1st+2nd)	39.01* (**40.20**)	41.89 (**42.70**)	43.14 (**43.94****)	44.04 (**44.71****)	44.62 (**45.19****)	45.06 (**45.55****)	45.34 (**45.87****)	45.69** (**46.15****)	45.91** (**46.33****)	46.08** (**46.43****)
Macro-F1	GF	7.38 (11.01)	8.44 (13.55)	9.35 (14.93)	9.80 (15.90)	10.38 (16.45)	10.79 (16.93)	11.21 (17.38)	11.55 (17.64)	11.81 (17.80)	12.08 (18.09)
	DeepWalk	28.39	30.96	32.28	33.43	33.92	34.32	34.83	35.27	35.54	35.86
	DeepWalk (256dim)	28.95	31.79	33.16	34.42	34.93	35.44	35.99	36.41	36.78	37.11
	LINE(1st)	28.74 (29.40)	31.24 (31.75)	32.26 (32.74)	33.05 (33.41)	33.30 (33.70)	33.60 (33.99)	33.86 (34.26)	34.18 (34.52)	34.33 (34.77)	34.44 (34.92)
	LINE(2nd)	17.06 (22.18)	21.73 (27.25)	25.28 (29.87)	27.36 (31.88)	28.50 (32.86)	29.59 (33.73)	30.43 (34.50)	31.14 (35.15)	31.81 (35.76)	32.32 (36.19)
	LINE(1st+2nd)	**29.85** (29.24)	31.93 (**33.16****)	33.96 (**35.08****)	35.46** (**36.45****)	36.25** (**37.14****)	36.90** (**37.69****)	37.48** (**38.30****)	38.10** (**38.80****)	38.46** (**39.15****)	38.82** (**39.40****)

Significantly outperforms DeepWalk at the: ** 0.01 and * 0.05 level, paired t-test.

Outline

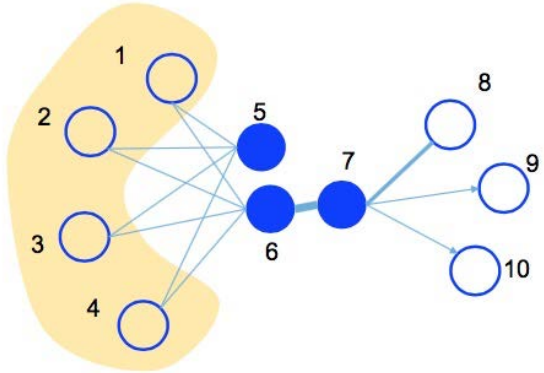
- Learning Embeddings in Networks and Text
  - LINE: Large-scale Information Network Embedding
  - Biological Relationship Discovery with Network Embedding
- LAKI: Representing Documents via Latent Keyphrase Inference
- MetaPAD: Meta Pattern Discovery from Massive Text Corpora
- TextCube, EventCube and CaseOLAP
- Summary

Biological Relationship Extraction

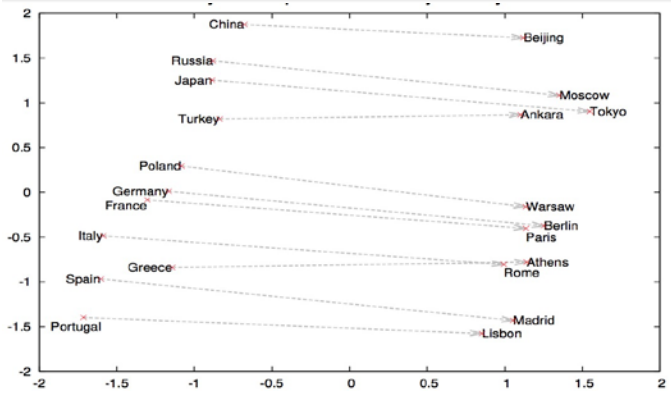
- Problem description: Automatic extraction of relationships between different biological entities from biological research papers
  - Examples
    - Gene – Disease; Drug - Disease; Drug - Pathway; Drug - Target gene
- Challenges
  - Entity detection
    - Most biological entities consist of several words
      - E.g., Non-small Cell Lung Cancer, Acute Myeloid Leukemia
  - Sparsity
    - Most biological entities co-occur only a few times in research papers
    - Most relationships are not explicitly described in papers
  - Few Labeled data

Overview of Embedding-Based Methods

- Goal: Find low-dimensional vector representation for entities and similar entities should have similar representations
- Data: Raw text or co-occurrence network
- Hypothesis: Entities which share many neighbors turn to be similar to each other
- Observation: Most entities co-occur only few times in papers, but they always share many neighbors
- Examples on word co-occurrence network
  - Word similarity: San Francisco
  - Word relation: $A{:}\,B \approx C{:}\,?$
    - France: Paris, China: ? => Beijing
    - $Argmax_X\ Sim(\vec{B}\text{-}\vec{A}\text{+}\vec{C}, \vec{X})$



Word	Cosine distance
los_angeles	0.666175
golden_gate	0.571522
oakland	0.557521
california	0.554623
san_diego	0.534939
pasadena	0.519115
seattle	0.512098
taiko	0.507570
houston	0.499762
chicago_illinois	0.491598

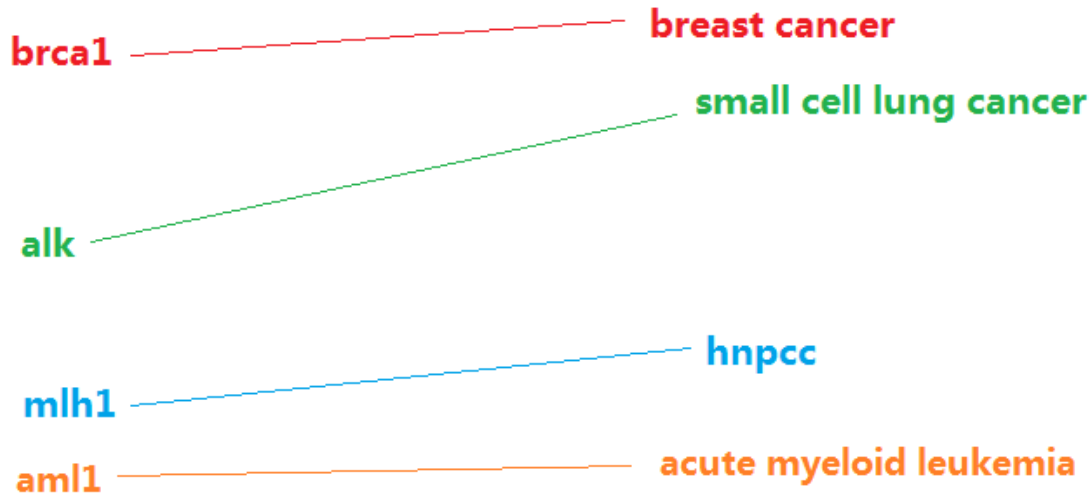


Framework

- Construct a biological entity list
  - SegPhrase
  - Biological vocabulary
- Detect and extract biological entities from biological text (research papers), using
  - SegPhrase
  - Maximum Matching
- Construct a co-occurrence network between words and biological entities
- Learn embedding vector for each entity by using a network embedding technique LINE (i.e., Run LINE-2nd to learn entity embeddings)
  - Entities share many neighbors in the co-occurrence network tend to be similar
- Given seed entity pair (a, b) and a query entity x, return an entity y according to their embedding vectors, so that the relation between x and y is similar to the relation between a and b.

Key Property to Learn Embedding & Experiments

- Key Property to Learn Embedding
  - The lines between genes and diseases are parallel
  - Given a seed pair (A,B) and a query X, we can find an entity Y which satisfies
    - $(A,B) \approx (X, Y)$
    - $Y = Argmax/sim(B - \vec{A} + X, Y)$



- Experimental Settings
  - Sample 10% Pubmed abstracts
  - Detect phrases by using a 200K phrase list
  - Build a co-occurrence network for all words and phrases
  - Learn entity embedding from the co-occurrence network

Experimental Results: Find Related Entities

- t-cell

Word	Cosine distance
b_cell	0.899536
t-lymphocyte	0.897891
nk-cell	0.874473
b-cell	0.871003
natural_killer_cell	0.857540
t_lymphocyte	0.855409
nk_cell	0.838348
b-lymphocyte	0.826705
cd8	0.825255
b_lymphocyte	0.818053

- Leukemia

Word	Cosine distance
leukaemia	0.968356
lymphoblastic	0.907420
leukemias	0.869621
myelocytic	0.860447
acute_myelogenous_leukemia	0.845294
myelogenous	0.843615
ph1+	0.831842
philadelphia-positive	0.830873
myelomonocytic	0.830090
acute_myeloid_leukemia	0.830058

- Doxorubicin

Word	Cosine distance
adriamycin	0.921346
paclitaxel	0.906449
epirubicin	0.904076
etoposide	0.889664
dox	0.887675
daunorubicin	0.883601
mitoxantrone	0.883223
cisplatin	0.881845
cddp	0.872368
docetaxel	0.847687

- Tumor Suppressor Gene

Word	Cosine distance
tumor_suppressor	0.900334
tumor-suppressor	0.883250
suppressor_gene	0.853865
tsgs	0.812559
dbccr1	0.810981
mmac1	0.809297
mts1	0.806708
tumour-suppressor	0.803866
smarca4	0.789171
cdkn2a	0.788514

Extracted Relations (from 10% PubMed Abstracts)

Relation	Seed Pair	Query Entity	Top Ranked Entities
Gene-Disease	Breast Cancer, BRCA1	Acute Myeloid Leukemia	AML1, E2A-PBX1, NPM1, RUNX1, PBX1
		Acute Lymphocytic Leukemia	E2A-PBX1, NPM1, EVI1, BCL6, ALL1
		HNPCC	MLH1, MSH6, hMSH2, hMLH1, MSH2
	BRCA1, Breast Cancer	ALK	Small Cell Lung Cancer, Non-small Cell Lung Cancer
		AML1	Leukemia, AML, CML
		MLH1	Colorectal Cancer, HNPCC, Colon Cancer
Drug-Disease	Leukemia, Doxorubicin	Small Cell Lung Cancer	Paclitaxel, Gemcitabine, Docetaxel, Cisplatin
		Depressive Disorder	Sertraline, Desvenlafaxine, Duloxetine, Paliperidone
		HIV	Zidovudine, Ritonavir, Lamivudine, Atazanavir
	Doxorubicin, Leukemia	Aspirin	Peptic Ulcer Bleeding, Venous Thromboembolic
		Sertraline	Depressive Disorder, Social Anxiety Disorder
		Penicillin	Bacterial Meningitis, Scabies, Streptococcus

Relation: Drug-Target Gene

- Very difficult as there are too many genes
- Use (Camptothecin: top1) as seed pair
- Metric: rank of the first-hit target gene
- Compare with a paper published on CELL

Drug	Embedding	CELL Paper
cycloheximide	10	1
doxorubicin hydrochloride	1998	1000
etoposide	9	13
geldanamycin	426	5
methotrexate	3	6
monastrol	1116	1000
rapamycin	5	11
trichostatin	7	100

Future Work

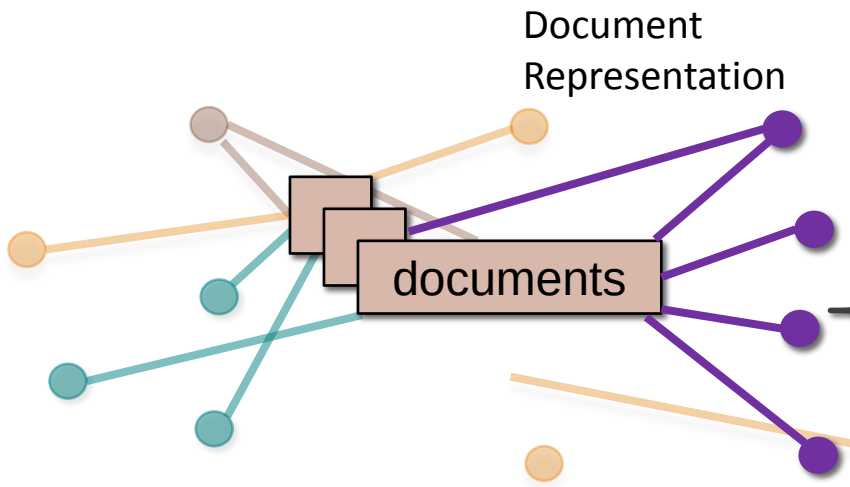
- Use more context information
  - By using topic assignment of words as extra features, the performance on word analogy task is improved
- Use biological networks
  - Well constructed with little noise
- Exploring multi-typed information networks
  - Integration of expert-provided type information and automated type extraction (e.g., ClusType)
  - Exploring meta-path
- Add a regularization term to the model

Outline

- Learning Embeddings in Networks and Text
  - LINE: Large-scale Information Network Embedding
  - Biological Relationship Discovery with Network Embedding
- LAKI: Representing Documents via Latent Keyphrase Inference 
- MetaPAD: Meta Pattern Discovery from Massive Text Corpora
- TextCube, EventCube and CaseOLAP
- Summary

LAKI: Representing Documents via Latent Keyphrase Inference

- Jialu Liu, Xiang Ren, Jingbo Shang, Taylor Cassidy, Clare Voss and Jiawei Han, "Representing Documents via Latent Keyphrase Inference", WWW'16
- **Document Representation**



Words:
dbscan, methods, clustering, process, …

Topics:
[k-means, clustering, clusters, dbscan, …]
[clusters, density, dbscan, clustering, …]
[machine, learning, knowledge, mining, …]

Knowledge base concepts:
data mining: /m/0blvg
clustering analysis: /m/031f5p
dbscan: /m/03cg_k1

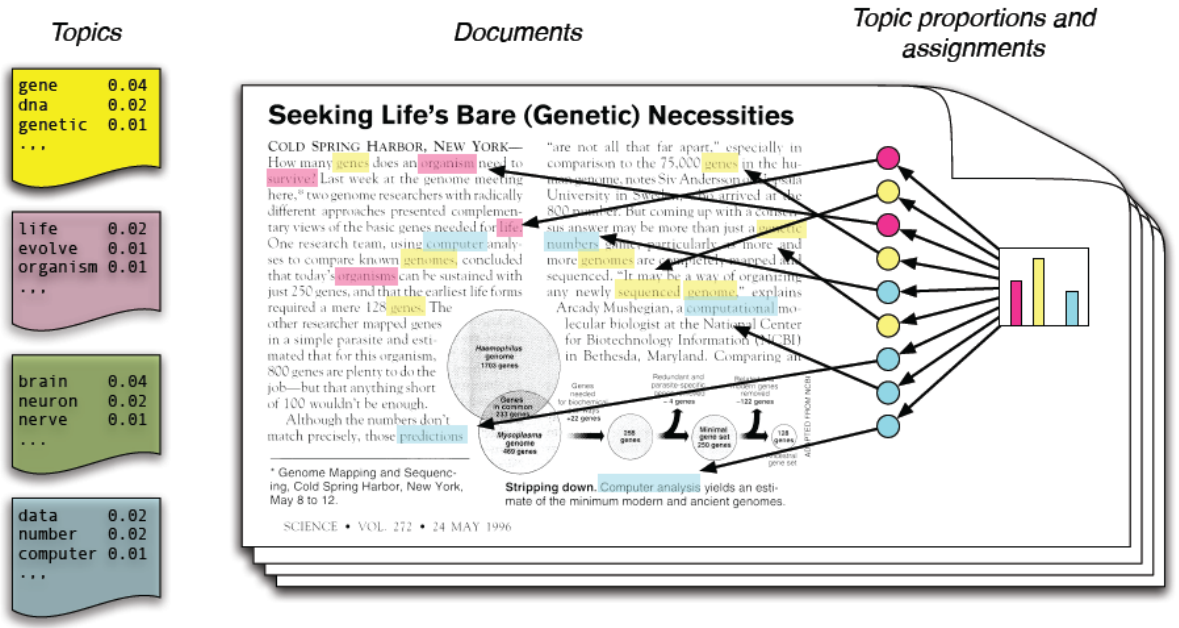
Document keyphrase:
dbscan: [dbscan, density, clustering, ...]
clustering: [clustering, clusters, partition, ...]
data mining: [data mining, knowledge, ...]

- A document can be represented by
  - A set of works, topics, KB concepts, Keyphrases, …

Document Representation: Traditional Methods

- Bag-of-Words or Bag-of-Phrases
  - Cons: Sparse on short texts
- Topic models [LDA]
  - Each **topic** is a distribution over words; each **document** is a mixture of corpus-wide topics
  - Cons: Difficult for human to infer topic semantics

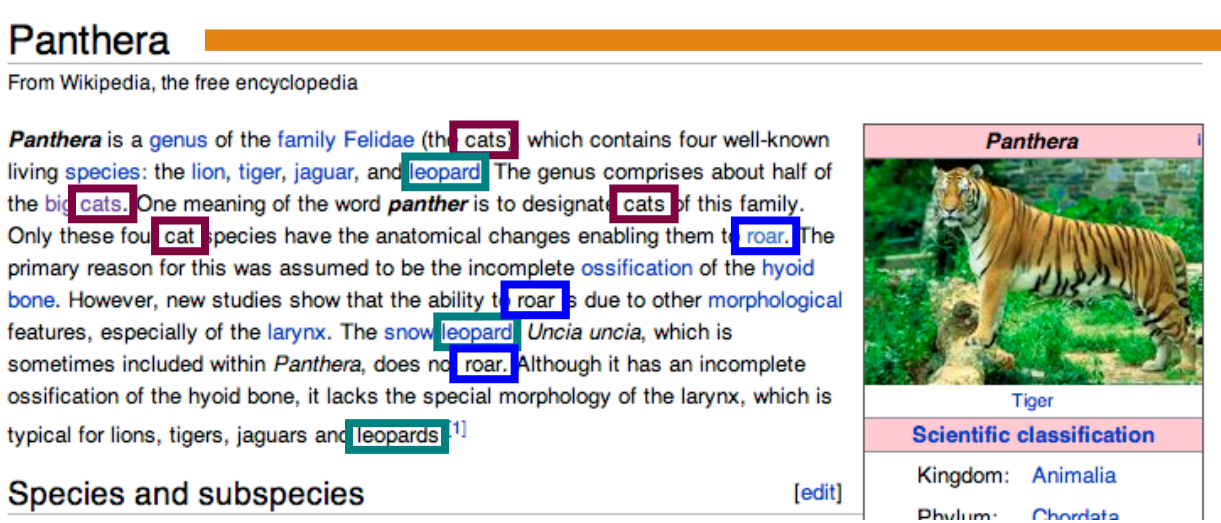
	doc1	doc2	doc3
I	1	0	0
like	1	0	0
football	1	1	0
John	0	1	1
likes	0	1	1
football	1	1	0
basketball	0	0	1

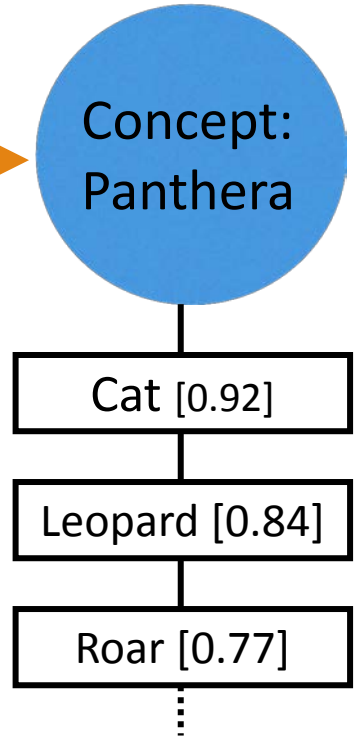


Document Representation: Concept-based models [ESA]

- Concept-based models [ESA]
  - Cons: Low coverage of concepts in human-curated knowledge base

Every Wikipedia article represents a **concept**

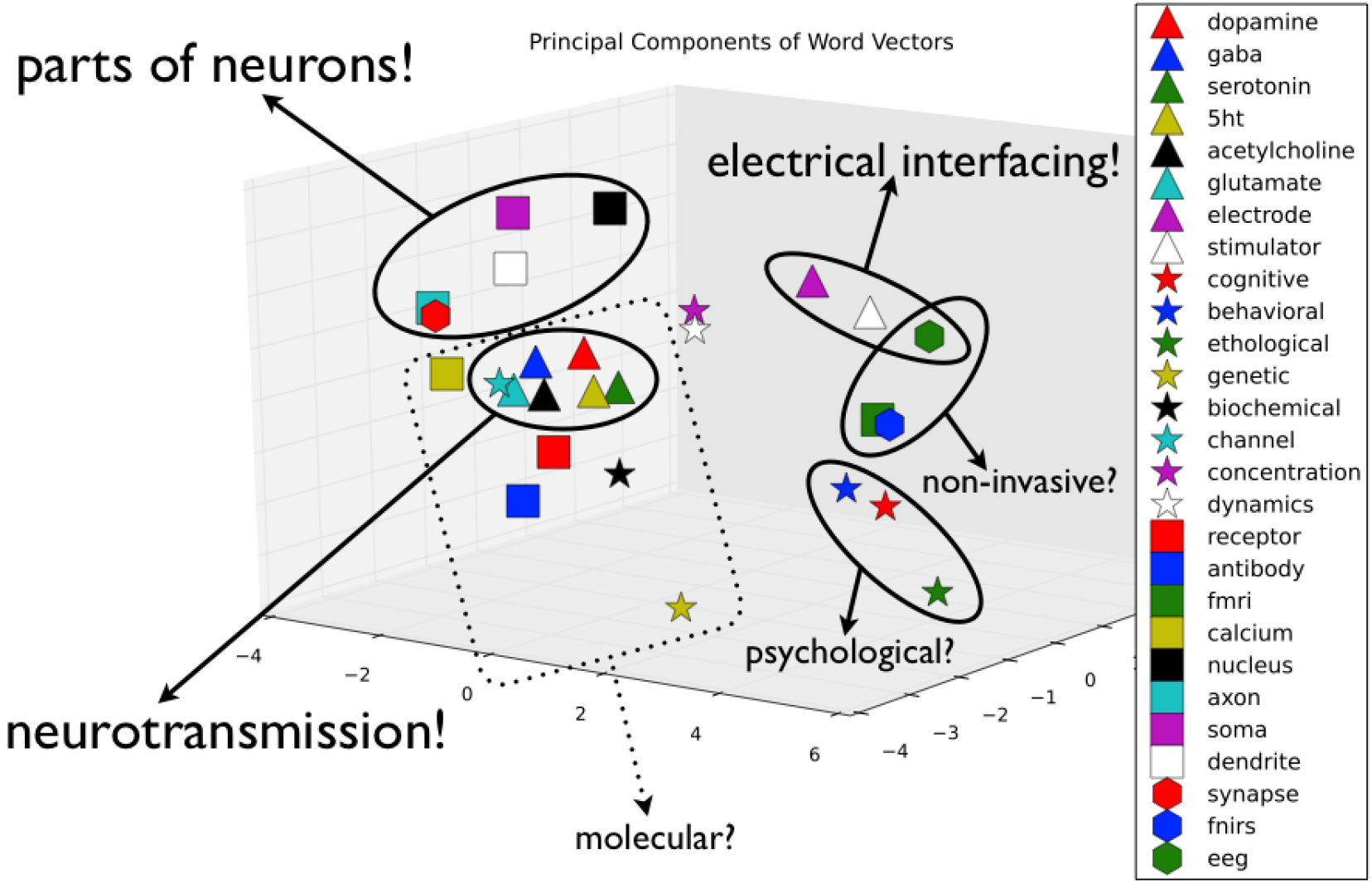




Article words are associated with the concept (TF.IDF), which help infer concepts from document

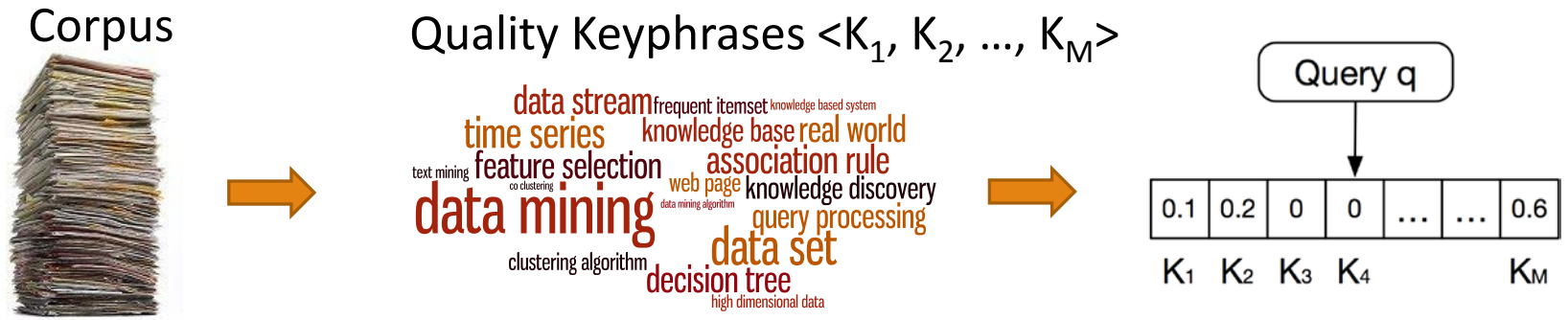
Document Representation: Word2Vect

- Word/document embedding models [word2vec]
  - Cons: Difficult to explain what each dimension means



Document Representation Using Keyphrases

- Use quality phrases as the entries in the vector and identify document keyphrases (subset of quality phrases) by evaluating relatedness between (doc, quality phrase)
- Unsupervised model

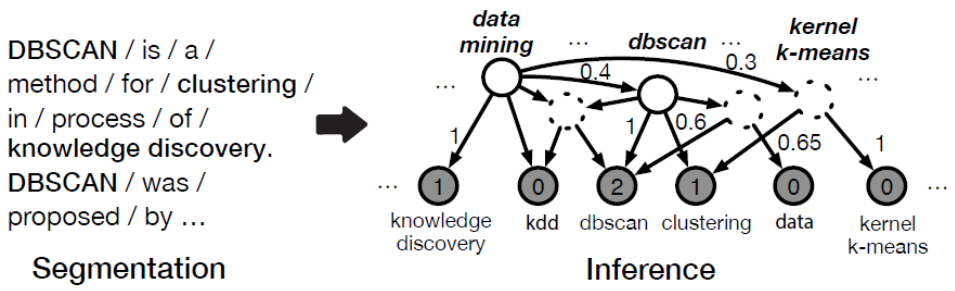


- Challenges
  - Where to get quality phrases from a given corpus?
    - Mining Quality Phrases from Massive Text Corpora [SIGMOD15]
  - How to identify document keyphrases?
    - Can be latent mentions
    - Relatedness scores
  - How to deal with relationship between quality phrases?

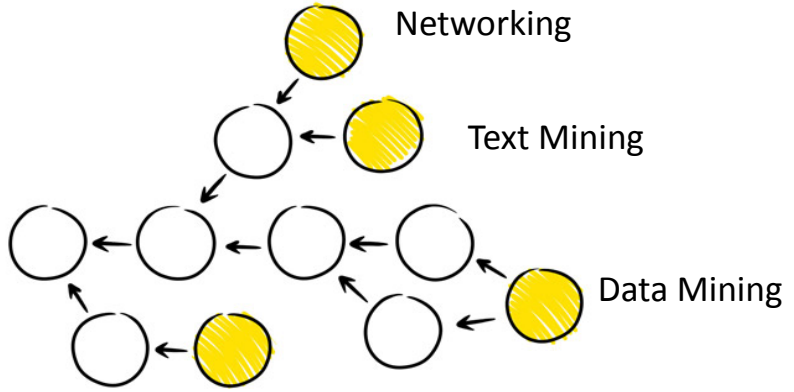
Document Representation Using Keyphrases: General Ideas

- How to identify document keyphrases?
  - Powered by Bayesian Inference on "Quality Phrase Silhouette"
    - Quality Phrase Silhouette: Topic centered on quality phrase
      - "Reverse" topic models
      - "Pseudo content" for quality phrase

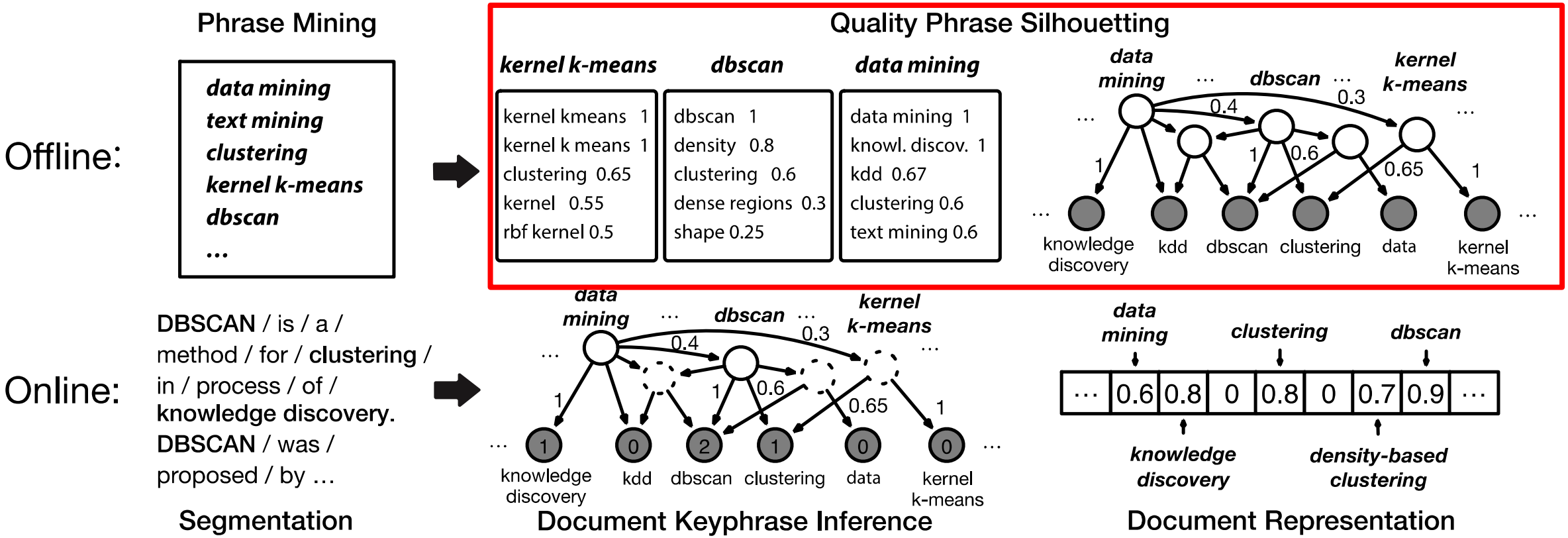
kernel k-means	*dbscan*	*data mining*
kernel kmeans 1	dbscan 1	data mining 1
kernel k means 1	density 0.8	knowl. discov. 1
clustering 0.65	clustering 0.6	kdd 0.67
kernel 0.55	dense regions 0.3	clustering 0.6
rbf kernel 0.5	shape 0.25	text mining 0.6



- How to deal with relationship between quality phrases?
  - Phrases are interconnected as a Directed Acyclic Graph

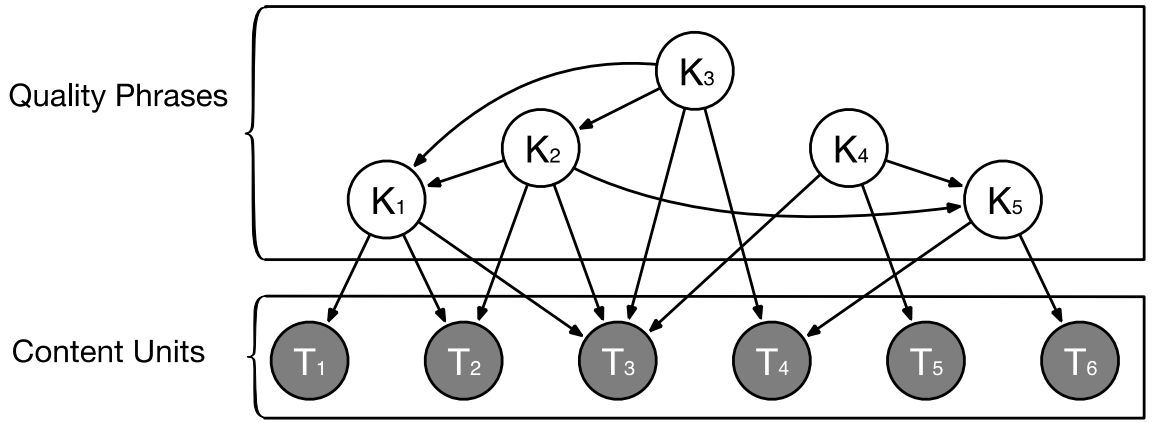


Framework for Latent Keyphrase Inference (LAKI)



LAKI: Deriving Quality Phrase Silhouette

- Learning Hierarchical Bayesian Network (DAG)



Binary Variables

Task 1: Model Learning: Learning link weights

Task 2: Structure Learning: Learning network structure

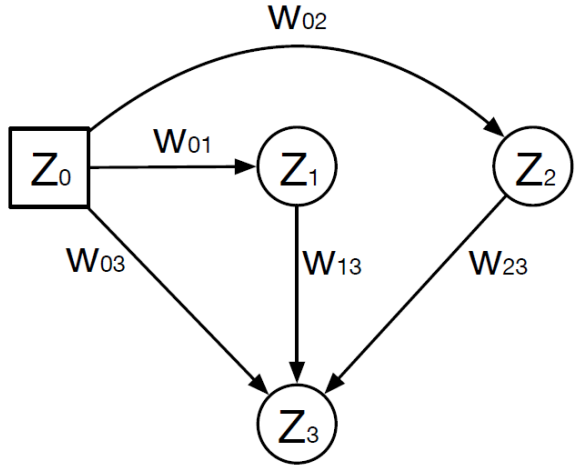
Deriving Quality Phrase Silhouette Task 1: Model Learning Given Structure

- Use Z to represent K (quality phrases) and T (content units)
- Noisy-OR
  - A parent node is easier to activate its children when the link weight is larger
  - A child node is influenced by all its parents

$$p(Z_j = 1 | Pa(Z_j)) = 1 - exp\left(-W_{0j} - \sum_i W_{ij} \mathbb{1}_{Pa_j^i}\right)$$

Noise / Prior

Aggregated over all other links connected with Z_j



Toy example

- Maximum Likelihood Estimation
  - Training data: Documents
  - Expectation-step: For each document, collect sufficient statistics
    - Link firing (Parent, child both being activated) probability
    - Node activation probability
  - Maximization-step: Update link weight

Partially observed document keyphrases

$$L(D) = \sum_{d=1}^{N} \log \sum_{k \in \Omega^{(d)}} p(K = k, T = t^{(d)})$$

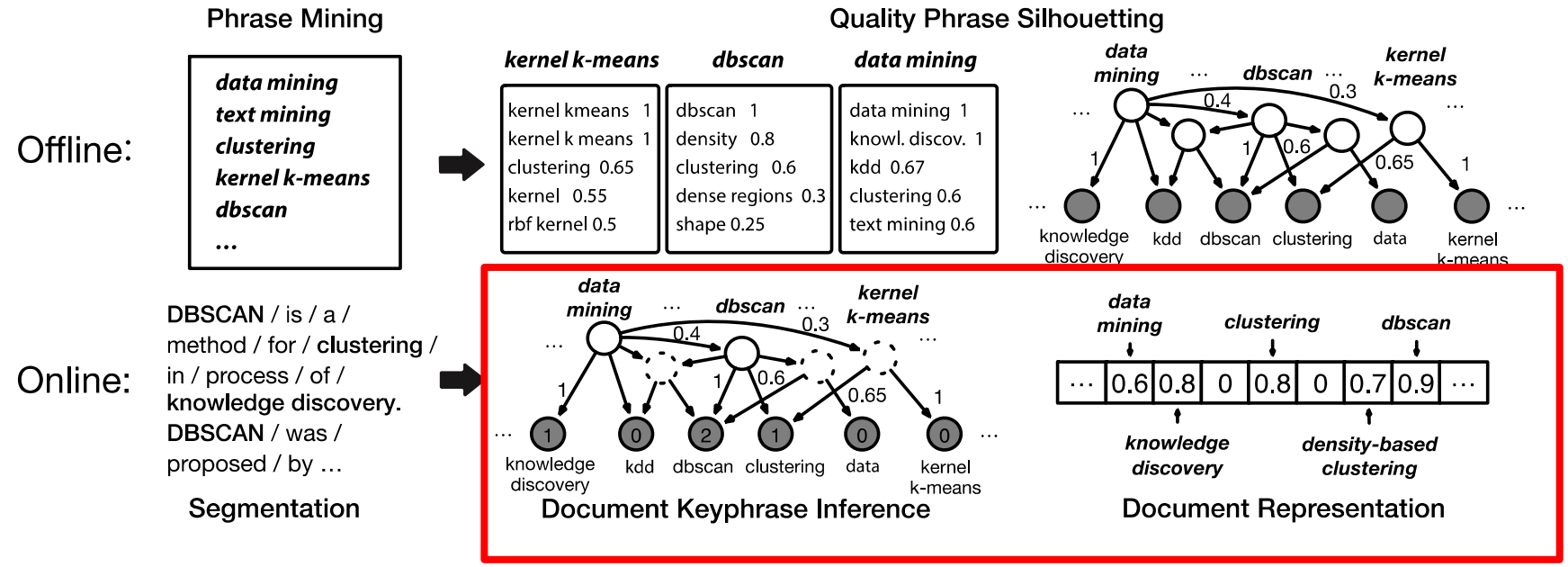
Fully observed content units

Deriving Quality Phrase Silhouette Task 2: Structure Learning

- Structure Learning
  - Quality phrases are connected to content units
    - Help infer document keyphrases from content units
  - Quality phrases are interconnected
    - Help infer document keyphrases from other keyphrases
- A Heuristic Approach
  - Data-Driven, DAG, similar to ontology
  - Heuristic:
    - Two nodes are connected only
      - Closely Related: word2vec
      - Co-occur frequently
    - Links are always pointing to less frequent nodes
  - Works well in practice

LAKI: Inference

- When do we need inference？
  - Expectation step in model learning
  - New documents
- Why is it slow?
  - NP hard to compute posterior probability for Noisy-Or networks



- Method: Approximate inference instead
  - Pruning irrelevant nodes using an efficient scoring function
  - Gibbs sampling

LAKI: Experiment Setting

- Two text-related tasks to evaluate document representation quality
  - Phrase relatedness
  - Document classification
- Two datasets:
- Methods:
  - **ESA** (Explicit Semantic Analysis)
  - **KBLink** uses link structure in Wikipedia
  - **BoW** (bag-of-words)
  - **ESA-C:** extends ESA by replacing Wiki with domain corpus
  - **LSA** (Latent Semantic Analysis)
  - **LDA** (Latent Dirichlet Allocation)
  - **Word2Vec** is a neural network computing word embeddings
  - **EKM** uses explicit keyphrase detection

Dataset	#Docs	#Words	Content type
Academia	0.43M	28M	title & abstract
Yelp	0.47M	98M	review

Method	Semantic Space	Input Source
ESA	KB concepts	KB
KBLink	KB concepts	KB
BoW	Words	-
ESA-C	Documents	Corpus
LSA	Topics	Corpus
LDA	Topics	Corpus
Word2Vec	-	Corpus
EKM	Explicit Keyphrases	Corpus
LAKI	Latent Keyphrases	Corpus

LAKI: Experimental Results

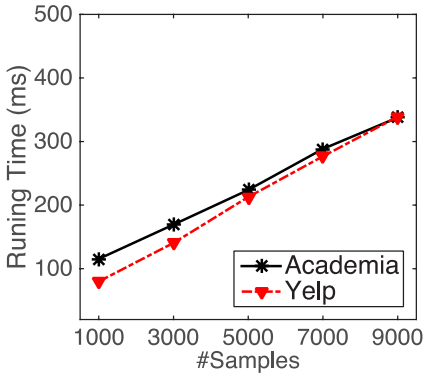
- **Phrase Relatedness Correlation**

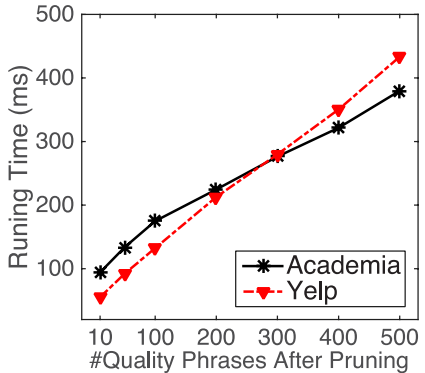
Method	Academia (w/ phrase)	Yelp (w/ phrase)
ESA	37.61 (-)	46.56 (-)
KBLink	36.37 (-)	35.94 (-)
BoW	48.05 (45.60)	51.26 (45.97)
ESA-C	39.75 (42.20)	49.13 (54.51)
LSA	72.50 (79.22)	66.55 (78.57)
LDA	77.27 (80.52)	75.55 (82.65)
EKM	45.46	40.57
LAKI	**84.42**	**90.58**

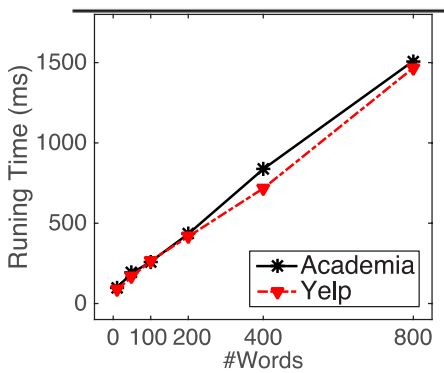
- **Document Classification**

Method	Academia (w/ phrase)	Yelp (w/ phrase)
ESA	0.4320 (-)	0.4567 (-)
KBLink	0.1878 (-)	0.4179 (-)
ESA-C	0.4905 (0.5243)	0.4655 (0.5029)
LSA	0.5877 (0.6383)	0.6700 (0.7229)
LDA	0.3610 (0.5391)	0.3928 (0.5405)
Word2Vec	0.6674 (0.7281)	0.7143 (0.7419)
LAKI	**0.7504**	**0.7609**

- **Time Complexity**







Case Study

- Query on phrases
  - Academia
  - Yelp
- Query on short documents (paper titles or sentences)
  - Academia
  - Yelp

Query	*LDA*	*BOA*
Keyphrases	linear discriminant analysis, latent dirichlet allocation, topic models, topic modeling, face recognition, sda, latent dirichlet, generative model, topic, subspace models, ...	boa steakhouse, bank of america, stripsteak, agnolotti, credit card, santa monica, restaurants, wells fargo, steakhouse, prime rib, bank, vegas, las vegas, cash, cut, dinner, bank, money, ...
Query	*LDA topic*	*BOA steak*
Keyphrases	latent dirichlet allocation, topic, topic models, topic modeling, probabilistic topic models, latent topics, topic discovery, generative model, mixture, text mining, topic distribution, plsi, ...	steak, stripsteak, boa steakhouse, steakhouse, ribeye, craftsteak, santa monica, medium rare, prime, vegas, entrees, potatoes, french fries, filet mignon, mashed potatoes, texas roadhouse, ...
Query	*SVM*	*deep dish pizza*
Keyphrases	support vector machines, svm classifier, multi class, training set, margin, knn, classification problems, kernel function, multi class svm, multi class support vector machine, support vector, ...	deep dish pizza, chicago, deep dish, amore taste of chicago, amore, pizza, oregano, chicago style, chicago style deep dish pizza, thin crust, windy city, slice, pan, oven, pepperoni, hot dog, ...

Query	*Mining Frequent Patterns without Candidate Generation*	*I am a huge fan of the All You Can Eat Chinese food buffet.*
Keyphrases	mining frequent patterns, candidate generation, frequent pattern mining, candidate, prune, fp growth, frequent pattern tree, apriori, subtrees, frequent patterns, candidate sets, ...	all you can eat, chinese food, buffet, chinese buffet, dim sum, orange chicken, chinese restaurant, asian food, asian buffet, crab legs, lunch buffet, fan, salad bar, all you can drink, ...
Query	*Text mining, also referred to as text data mining, roughly equivalent to text analytics, refers to the process of deriving high-quality information from text. High-quality information is typically derived through means such as statistical pattern learning.*	*It's the perfect steakhouse for both meat and fish lovers. My table guest was completely delirious about his Kobe Beef and my lobster was perfectly cooked. Good wine list, they have a lovely Sancerre! Professional staff, quick and smooth.*
Keyphrases	text analytics, text mining, patterns, text, textual data, topic, information, text documents, information extraction, machine learning, data mining, knowledge discovery, ...	kobe beef, fish lovers, steakhouse, sancerre, wine list, guests, perfectly cooked, lobster, staff, meat, fillet, fish, lover, seafood, ribeye, filet, sea bass, risotto, starter, scallops, steak, beef, ...
	Academia	Yelp

Outline

- Learning Embeddings in Networks and Text
  - LINE: Large-scale Information Network Embedding
  - Biological Relationship Discovery with Network Embedding
- LAKI: Representing Documents via Latent Keyphrase Inference
- MetaPAD: Meta Pattern Discovery from Massive Text Corpora
- TextCube, EventCube and CaseOLAP
- Summary

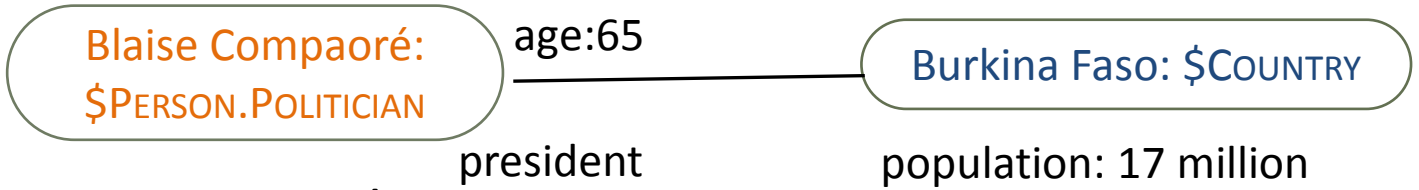
MetaPAD: Meta Pattern-driven Attribute Discovery from Massive Text Corpora

- Meng Jiang, Jingbo Shang, Xiang Ren, Taylor Cassidy, Lance Kaplan, Timothy Hanratty, and Jiawei Han, "MetaPAD: Meta Pattern-driven Attribute Discovery from Massive Text Corpora", submitted in 2017

- Motivation:

Given a sentence in a large corpus, "President Blaise Compaoré's government of **Burkina Faso** was founded…", …

We may find:



- Attribute Discovery: Two tasks

Task 1: ⟨entity, attribute name, attribute value⟩

⟨Burkina Faso, president, Blaise Compaoré⟩

⟨Burkina Faso, population, 17 million⟩

⟨Blaise Compaoré, age, 65⟩

Instance-level

Task 2: ⟨entity type, attribute name⟩

⟨$COUNTRY, president⟩

⟨$COUNTRY, population⟩

⟨$PERSON, age⟩

Type-level

Previous Work on Finding E-A-V and Typed Patterns

- Task 1: Finding E-A-V at the Instance Level

Ignore entity-typing information!

  - Stanford OpenIE [ACL'15], AI^2's Open IE-Ollie [EMNLP'12]
    - Learn syntactic and lexical patterns of expressing relations
  - Input: "President Blaise Compaoré**'s** government of Burkina Faso was founded…"
  - Output: ⟨President Blaise Compaoré, **have**, government of Burkina Faso⟩ ☹
- Task 2: Finding Typed Patterns
  - Google's Biperpedia+ARI [VLDB'14, WWW'16], ReNoun [EMNLP'15]:

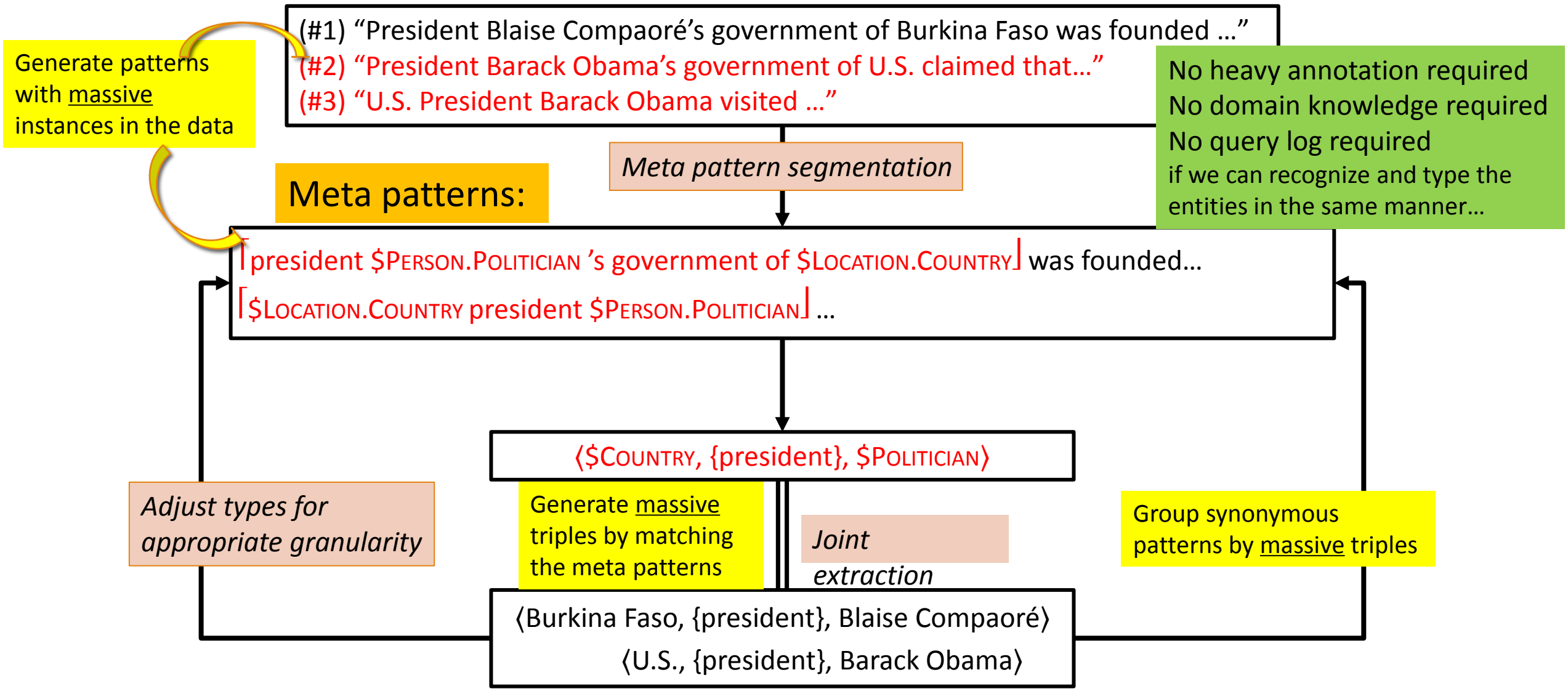
"president of united states" → "A of E", "E 's A", "E A", "A, E"

"Barack Obama, President of U.S., " → "O, A of S,", "S A O"

Query log: Highly constrained and unavailable

  - Input: "…Sunday night, Burkina Faso…" and the "A, E" pattern
  - Output: ⟨$COUNTRY, Sunday night⟩ ☹

Our Meta-Pattern Methodology



Pattern Discovery by Phrase Mining and Entity Typing

"President Blaise Compaoré's government of Burkina Faso was founded …"

↓

Phrase mining (SegPhrase and AutoPhrase)

↓

"president blaise_compaoré's government of burkina_faso was founded …"

↓

Entity recognition and typing with Distant Supervision (ClusType)

↓

"president $PERSON's government of $LOCATION was founded …"

↓

Fine-grained typing (PLE by Ren et al. KDD'16)

↓

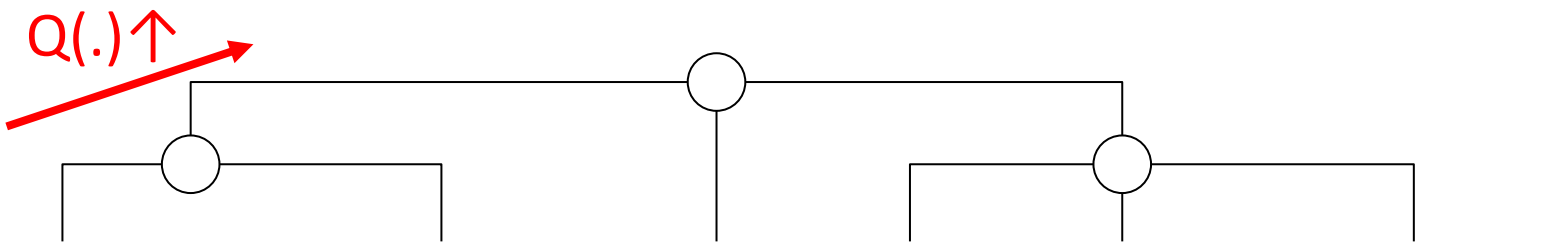
"president $PERSON.POLITICIAN's government of $LOCATION.COUNTRY was founded …"

Meta-Pattern Quality Assessment and Segmentation

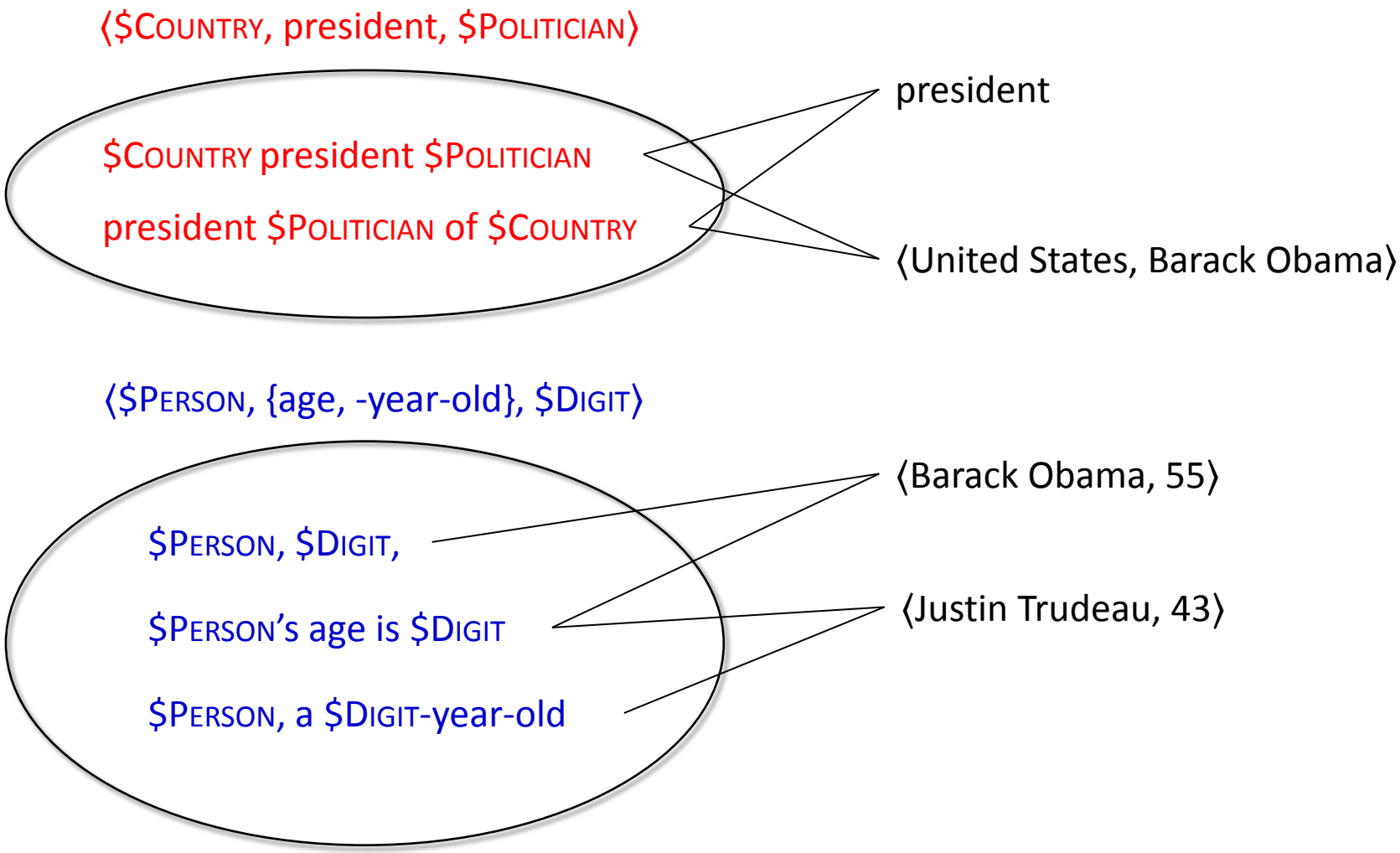
A rich set of features:

- ✓ Frequency
- ✓ Concordance: "$PERSON's wife"
- ✓ Completeness: "$COUNTRY president" vs. "$COUNTRY president $POLITICIAN"
- ✓ Informativeness: "$PERSON and $PERSON" vs. "$PERSON 's wife, $PERSON"

Regression Q(.): random forest with only 300 labels



Grouping Synonymous Patterns

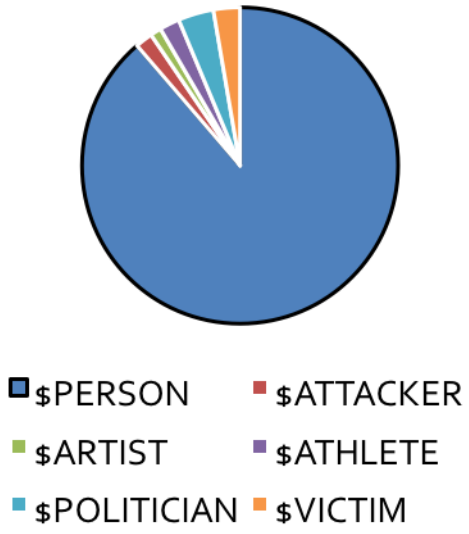


Adjusting Types in Meta Patterns for Appropriate Granularity

$PERSON, $DIGIT,

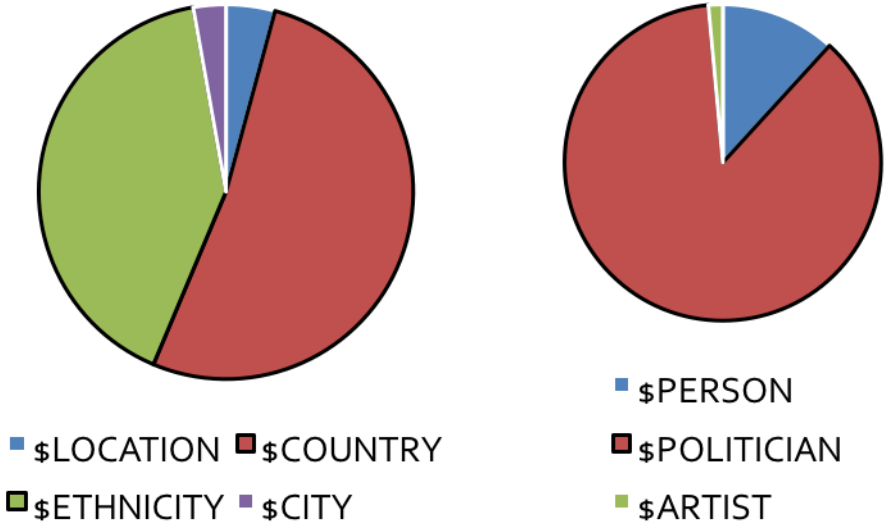
$PERSON's age is $DIGIT

$PERSON, a $DIGIT -year-old



$COUNTRY president $POLITICIAN

president $POLITICIAN of $COUNTRY



Results: Patterns, Entities and Attribute Values in News Corpus

Meta patterns	Entity	Attribute value
$Country President $Politician $Country's president $Politician President $Politician of $Country … $Politician's government of $Country	United States	Barack Obama
	Russia	Vladimir Putin
	France	Francois Hollande
	…	…
	Burkina Faso	Blaise Compaoré

Meta patterns	Entity	Attribute value
$Company CEO $Person $Company chief executive $Person $Person, the $Company CEO, … $Company former CEO $Person $Person, the $Company former CEO,	Apple	Tim Cook
	Facebook	Mark Zuckerberg
	Hewlett-Packard	Carly Fiorina
	…	…
	Infor	Charles Phillips
	Afghan Citadel	Roya Mahboob

Patterns and Entities Found in Medical Science Corpus

Meta patterns	Entity	Attribute value
$Treatment was used to treat $Disease $Disease using the $Treatment $Treatment has been used to treat $Disease $Treatment of patients with $Disease …	zoledronic acid therapy	Paget's disease of bone
	bisphosphonates	osteoporosis
	calcitonin	Paget's disease of bone
	calcitonin	osteoporosis
	…	…
Meta patterns	**Entity**	**Attribute value**
$Bacteria was resistant to $Antibiotics $Bacteria are resistant to $Antibiotics $Bacteria is the most resistant to $Antibiotics … $Bacteria, particularly those resistant to $Antibiotics	corynebacterium striatum BM4687	gentamicin
	corynebacterium striatum BM4687	tobramycin
	methicillin-susceptible S aureus	vancomycin
	multidrug-resistant enterobacteriaceae	gentamicin
	…	…

Comparative Experimental Results

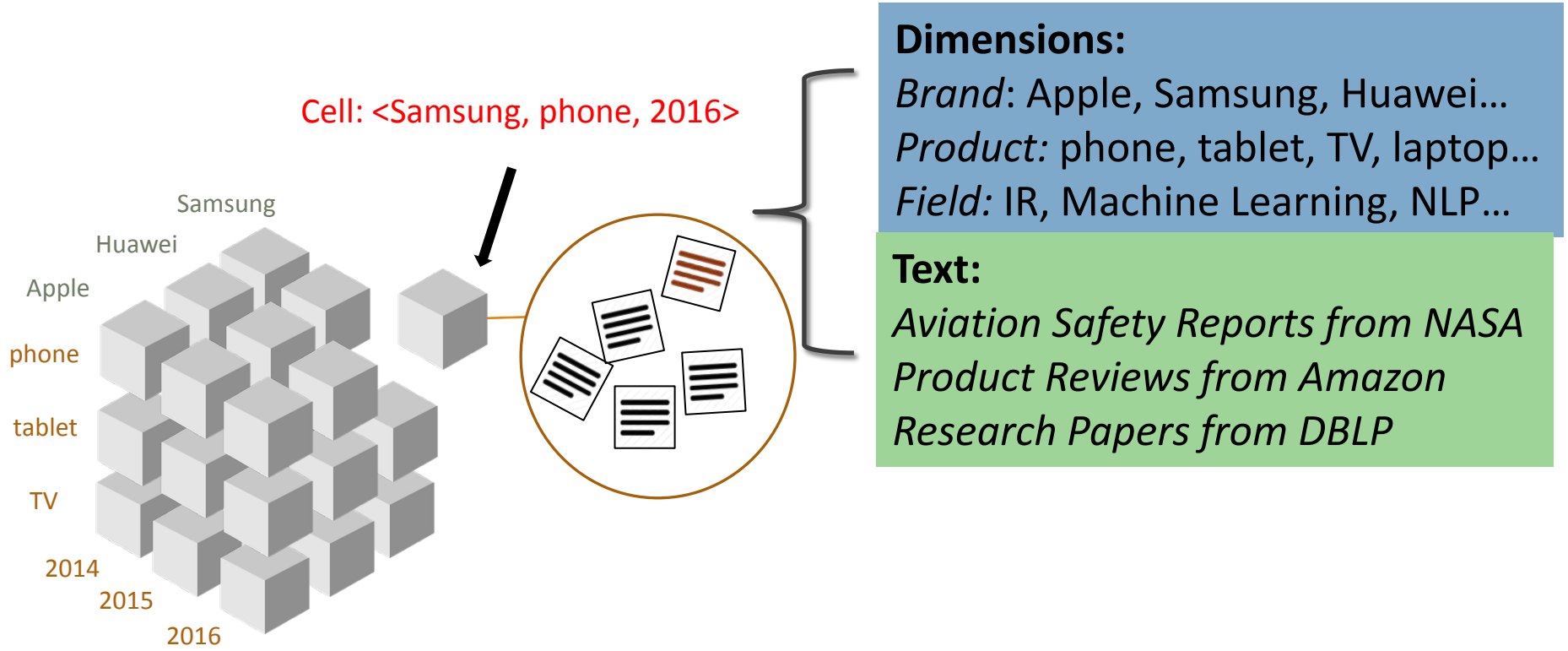
F1 score	⟨entity type, attribute name⟩	⟨entity, attribute name, attribute value⟩
Baselines		Stanford's OpenIE: 0.035
		AI2's Ollie: 0.131
	Biperpedia: 0.324	Google's ReNoun: 0.309
+Segmentation	+40.0%	+19.4%
+Type Adjustment	+6.5%	+15.0%
+Synonymous	+2.6%	
All	0.495 **relatively +52.9%**	0.424 **relatively +37.3%**

Outline

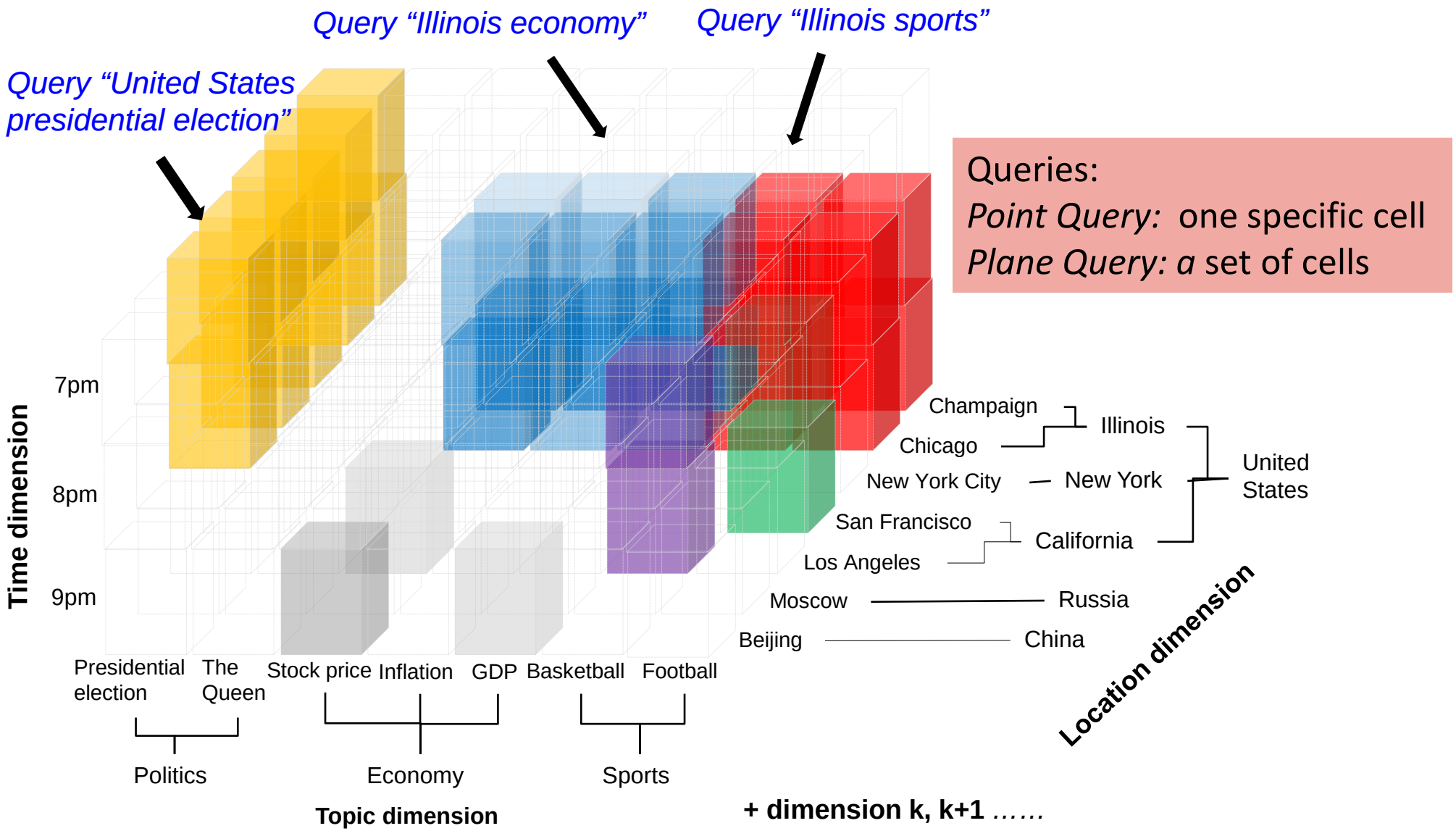
- Learning Embeddings in Networks and Text
  - LINE: Large-scale Information Network Embedding
  - Biological Relationship Discovery with Network Embedding
- LAKI: Representing Documents via Latent Keyphrase Inference
- MetaPAD: Meta Pattern Discovery from Massive Text Corpora
- TextCube, EventCube and CaseOLAP
- Summary

Multi-Dimensional Text Cube

- Numerical data cube (each cell is a numerical value) has been extensively studied
  - Measures: Numerical aggregations as *sum* & *avg*.
- Text cube: Each cell contains a set of documents (e.g., Apple, TV, 2016>)
  - There is an imminent need to do OLAP analysis on text cubes

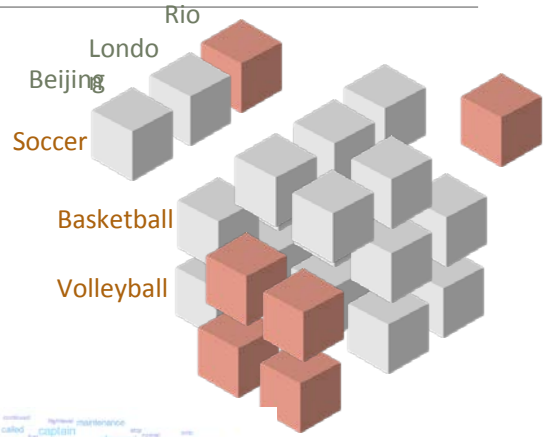


Multi-dimensional Text Cube with Queries & Hierarchies

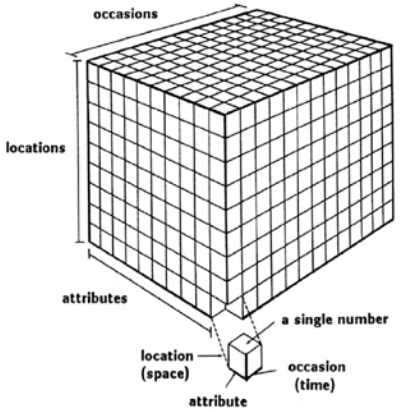


Exploration of Text Cube—Semantic Analysis

- EventCube [KDD'13 demo]: Point Query
  - Simple summary to support keyword/document search
- CASeOLAP [EngBul'16]: Plane Query
  - Comparative summary/mining



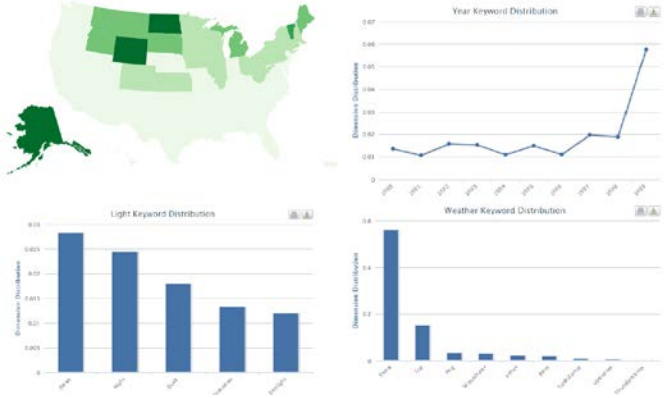
Cube Structure



Slice
Roll-up
Drill-down
Dice
...

Textual Analysis

Text Data



Structural Analysis

EventCube

- Multiple functions supported by EvenCube

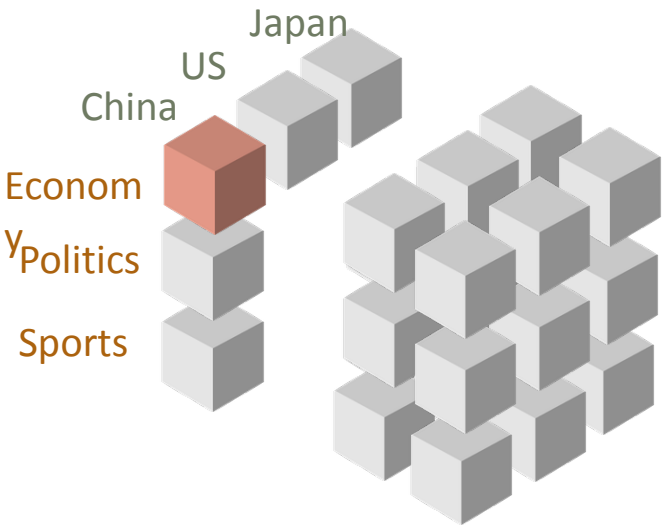
Multi-gram Summarization

y Distribution

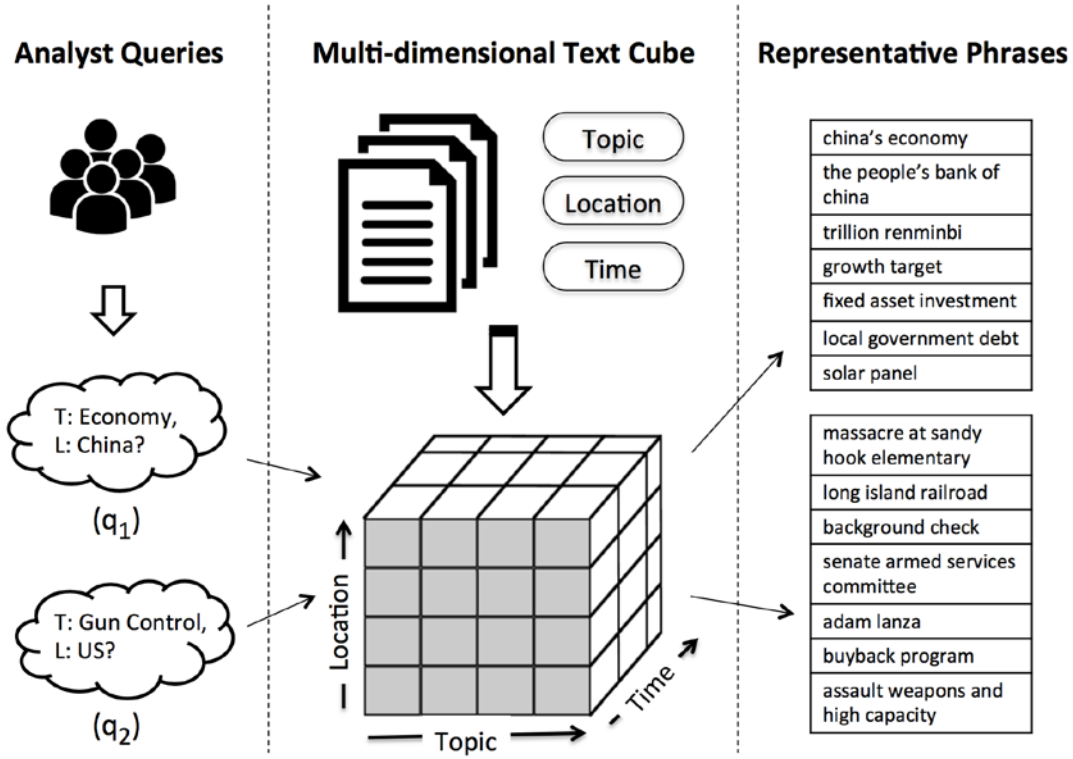
Query 1	Score
assign altitud	13.5010
altitud deviat	13.1992
cross restrict	12.7517
altitud restrict assign	12.2687
altitud excurs	11.6587
temporari flight restrict	11.0693
cross restrict cross	10.8610
altitud assign	10.3631
altitud restrict	10.2753
cross altitud	9.9490
cross restrict cross 90	9.2516
altitud captur	9.0427
4000 feet	9.0394
5000 feet	8.6173
250 deg	8.5411

CASE (Context-Aware SEmantic) OLAP

- A cell has comparative context
- Comparative study is meaningful
  - Given a query <**China, Economy**>
  - Target documents have frequent phrases
  - Be specific to “China”+“Economy”

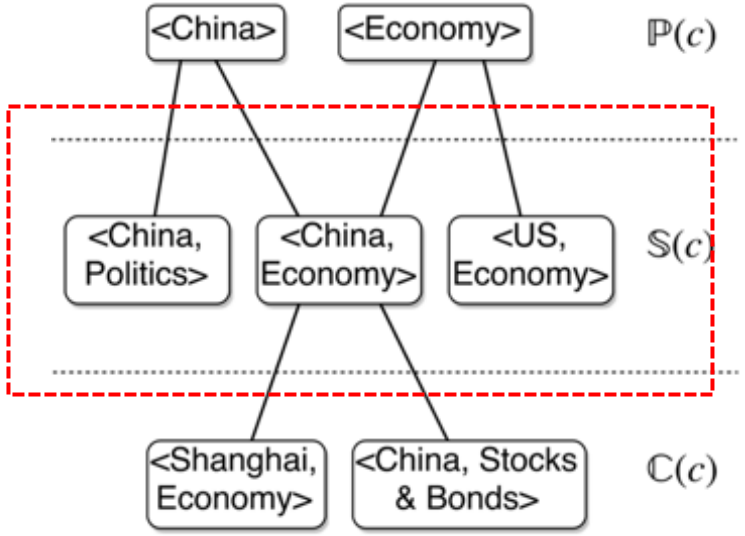


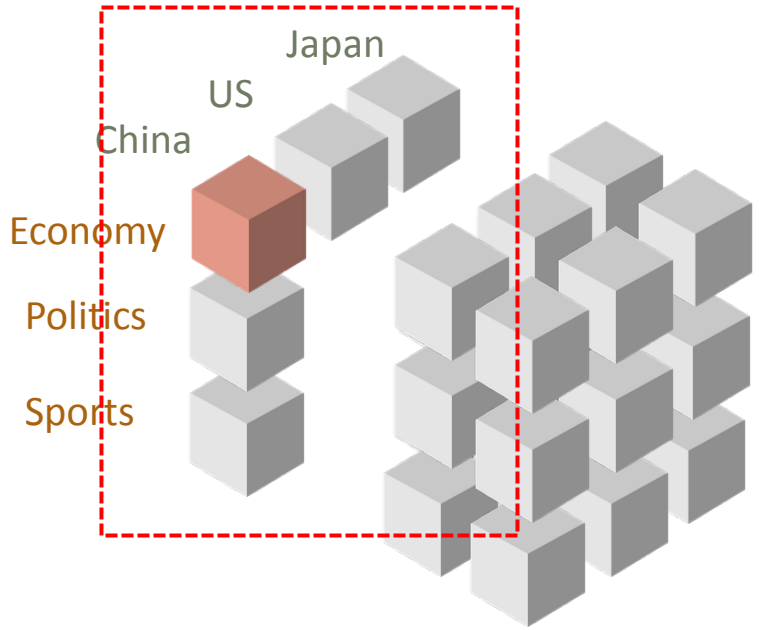
hong kong
united states
prime minister
double digit
communist party
economic growth
the united states
retail sales
G.D.P
monetary policy



Design Question I: Which Comparative Groups to Pick?

- Option 1: User-specified (too much burden to users): undesirable
- Option 2: **Sibling cells** in every dimension (comparable cells)





Design Question: How to Score Important Phrases?

- Three ingredients
  - Integrity: **meaningful, high-quality phrase**
    - Using SegPhrase as score (>0.7)
  - Popularity: large # of occurrences in the cell

$$pop(p, c) = \frac{\log(tf(p, c) + 1)}{\log cntP(c)} \qquad (2)$$

  - Distinctness: distinguish the target cell from context cells
    - A key to have a crisp definition
- Combining with geometric mean:

$$r(p, c) = \sqrt[3]{int(p, c) \cdot pop(p, c) \cdot disti(p, c)} \qquad (1)$$

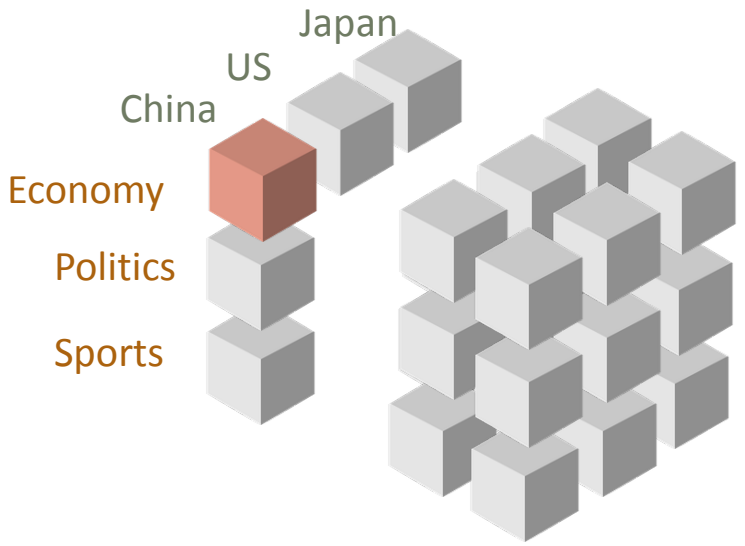
How to Find or Evaluate Distinct Phrases in a Cell?

- Judge if a phrase p is distinct in cell c: Transform it into a dual problem
  - *Original problem: Find distinctive phrases for cell c, compared to sibling cells*
  - *Transformed problem: Classify phrases into one of the most relevant cell*
- For a distinct phrase p, if we measure relevance(p, c) for all c
  - rel(p, c*) >> rel(p, sibling)
- Adopt Softmax function as

$$disti(p,c) = \frac{e^{rel(p,c)}}{1 + \sum_{c' \in \mathbb{S} \bigcup \{c\}, p \in c'} e^{rel(p,c')}} \quad (4)$$

Smoothing

Sibling relevance



How to Design Relevance Score for a Phrase to a Cell?

- Normalized Term Frequency
  - Treat each cell as a **super document**
  - Apply **BM25**

$$ntf(p,c) = \frac{tf(p,c)\cdot(k_1+1)}{tf(p,c)+k_1\cdot(1-b+b\cdot\frac{cntP(c)}{avgCP(c)})} \quad (5)$$

Balance cell size

- Normalized Document Frequency

$$ndf(p,c) = \frac{\log(1+df(p,c))}{\log(1+maxDF(c))} \quad (6)$$

Guarantee spread out!

- Combine:

$$rel(p,c) = ndf(p,c)\cdot ntf(p,c) \quad (7)$$

Experiments

- Data:
  - 4,785,990 news articles
  - Six dimensions
- Methods
  - **MCX**: based ratio to global background
  - **SegPhrase**: Using only integrity score
  - **MCX+Seg**: SegPhrase + MCX
  - **TF-IDF**: Distinctness by IDF, Popularity by TF
  - **RP (No INT)**: ablation without integrity
  - **RP (No POP)**: ablation without popularity
  - **RP (No DIS)**: ablation without distinctness
  - RP: our proposed method

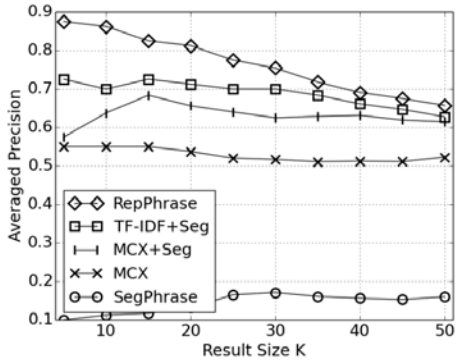


Figure 7: Phrase assignment accuracy comparison to baselines

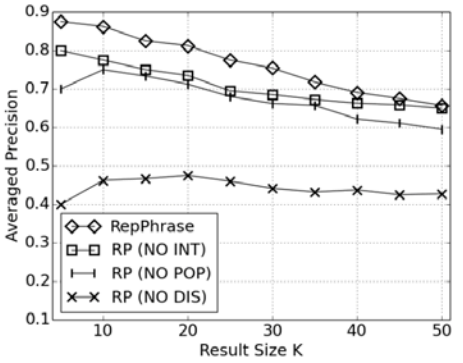
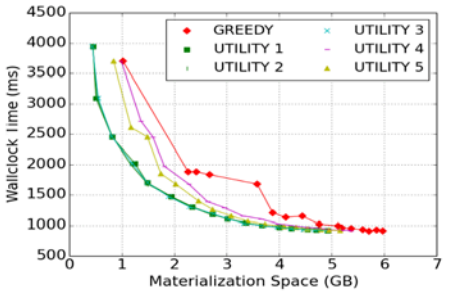
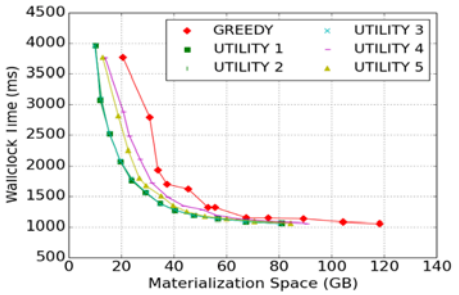


Figure 8: Phrase assignment accuracy comparison to ablations

Effectiveness



(a) Time-space balance of 4-Dim Cube



(b) Time-space balance of 6-Dim Cube

Efficiency

Effectiveness of CaseOLAP on Real-World Datasets

Distinct phrases on 2016 news data

Top-10 representative phrases for five example queries

⟨US, Gun Control⟩	⟨US, Immigration⟩	⟨US, Domestic Politics⟩	⟨US, Law and Crime⟩	⟨US, Military⟩
gun laws	immigration debate	gun laws	district attorney	sexual assault in the military
the national rifle association	border security	insurance plans	shot and killed	military prosecutors
gun rights	guest worker program	background check	federal court	armed services committee
background check	immigration legislation	health coverage	life in prison	armed forces
gun owners	undocumented immigrants	tax increases	death row	defense secretary
assault weapons ban	overhaul of the nation's immigration laws	the national rifle association	grand jury	military personnel
mass shootings	legal status	assault weapons ban	department of justice	sexually assaulted
high capacity magazines	path to citizenship	immigration debate	child abuse	fort meade
gun legislation	immigration status	the federal exchange	plea deal	private manning
gun control advocates	immigration reform	medicaid program	second degree murder	pentagon officials

PubMed Abstracts: Distinct relationships between subcategories of cardiovascular diseases and proteins

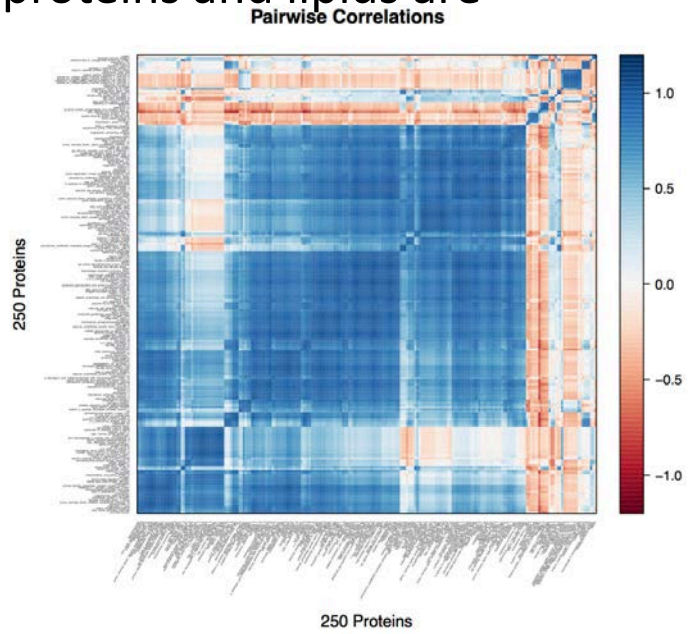
Table 2: Top representative phrases for 6 cardiac diseases

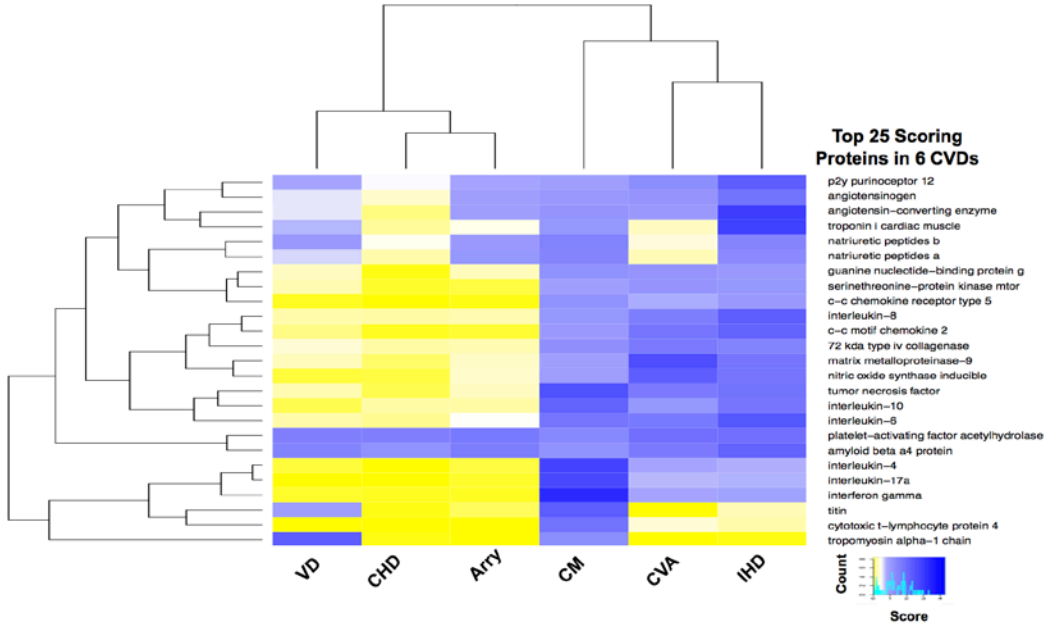
⟨Cerebrovascular Accident⟩	⟨Ischemic Heart Disease⟩	⟨Cardiomyopathy⟩	⟨Arrhythmia⟩	⟨Valve Dysfunction⟩	⟨Congenital Heart Disease⟩
alpha-galactosidase a	Cholesteryl ester transfer protein	Interferon gamma	Methionine synthase	Mineralocorticoid receptor	fibrillin-1
brain neurotrophic factor	apolipoprotein a-I	interleukin-4	ryanodine receptor 2	tropomyosin alpha-1 chain	plakophilin-2
tissue-type activator	integrin alpha-iib	interleukin-17a	potassium v.g. h member 2	elastin	tyrosine-protein type 11
apolipoprotein e	adiponectin	titin	inward rectifier channel 2	beta-2-glycoprotein 1	arachidonate 5-l-a protein
neurogenic l.n.h.p. 3	p2y purinoceptor 12	tumor necrosis factor	beta-2-glycoprotein 1	myosin-binding protein c	catechol o-methyltransferase

More on Real-World Case Study: BioData Analysis

- CASeOLAP found top-k representative proteins for each Cardiovascular Disease
  - Protein in IHD & CVA has similar pattern to reveal inflammatory function
  - Amyloid beta A4 appears to be consistent over 6 groups
  - Galactosidase had a high score of (5.903), suggesting that glycoproteins and lipids are highly relevant

Dataset: **CLINICAL BIOMARKER PAPERS**			
A collection of *PubMed* research papers about Cardiovascular Diseases from 1995-2016, with the top 250 highly relevant proteins to CVDs as target entities.			
# Documents	**# Dimensions**	**Avg. Word Count**	**Text Type**
500K	2	~2000	Academic
Dimension	**# Values**	**Description**	
Disease	6	Six main Cardiovascular disease groups with MeSH terms	
Year	12	Different years from 1995 to 2016	



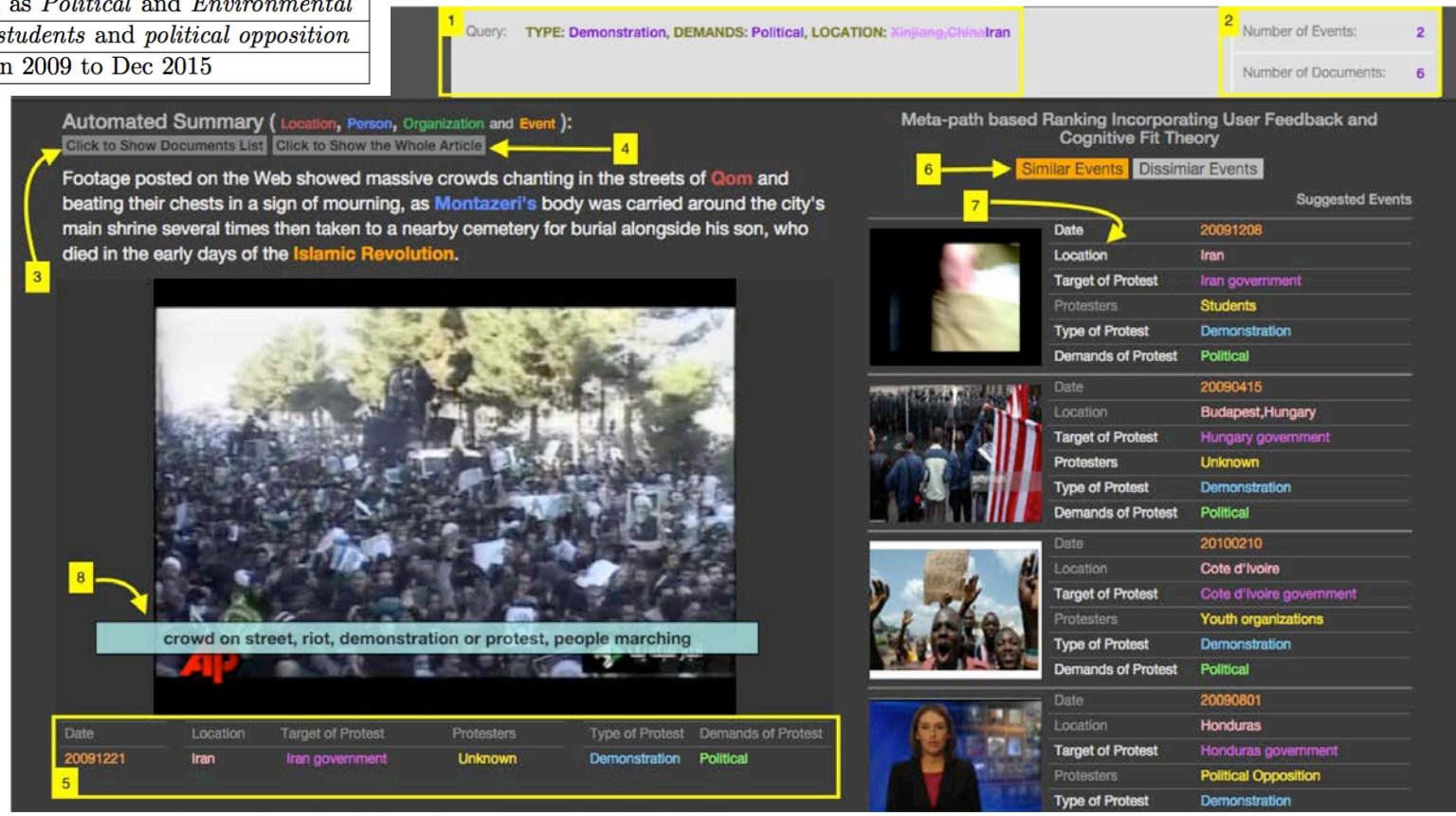


More on Real-World Cases: Protest News

Dataset: **PROTEST NEWS ARTICLES**			
A collection of Protest News Articles containing 59 protest incidents (January 2009 - December 2010) and 52 attack incidents (January 2014 - December 2015.)			
# Documents	**# Dimensions**	**Avg. Word Count**	**Text Type**
10K	6	~500	News

Dimension	# Values	Description
Incident	111	Individual protest and attach incidents
Location	20	Countries that the protests happened
Type of Protest	6	Six different types of pretest such as *Demonstration*
Demands of Protest	4	Four different types of pretest such as *Political* and *Environmental*
Protester	10	Different protester groups such as *students* and *political opposition*
Time	48	Different months spanning from Jan 2009 to Dec 2015

- CaseOLAP helps find top-k representative phrases and helps automatically generate representative video captions



Outline

- Learning Embeddings in Networks and Text
  - LINE: Large-scale Information Network Embedding
  - Biological Relationship Discovery with Network Embedding
- LAKI: Representing Documents via Latent Keyphrase Inference
- MetaPAD: Meta Pattern Discovery from Massive Text Corpora
- TextCube, EventCube and CaseOLAP
- Summary 

Summary

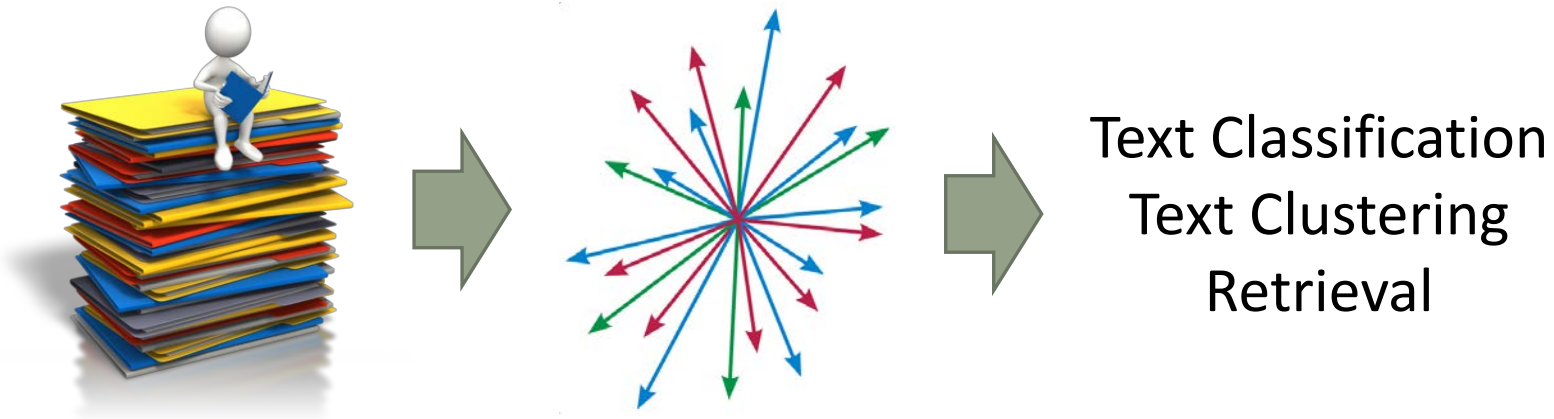
- A promising integrated approach for text analysis
  - Text mining + embedding + information network analysis
- Embedding for text analysis
  - LINE + PTE + information network approach (e.g., meta path-guided analysis)
- Representing documented by in-depth semantic analysis
  - LAKI and beyond
- MetaPAD: Meta Pattern Discovery from Massive Text Corpora
  - Facilitate information extraction from massive text using meta-patterns
- CaseOLAP: Distinctive analysis of documents in multi-dimensional space
- Many more to be studied

References

- JianTang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei, “LINE: Large-scale information network embedding”, WWW'15
- JianTang, Meng Qu, and Qiaozhu Mei, “PTE: Predictive text embedding through large-scale heterogeneous text networks”, KDD'15
- Jialu Liu, Xiang Ren, Jingbo Shang, Taylor Cassidy, Clare Voss and Jiawei Han, “Representing Documents via Latent Keyphrase Inference”, WWW'16
- Meng Jiang, Jingbo Shang, Taylor Cassidy, Xiang Ren, Lance M. Kaplan, Timothy P. Hanratty, and Jiawei Han, “MetaPAD: Meta Pa.ern Discovery from Massive Text Corpora”, submitted for publication, 2017
- Alon Halevy, Natalya Noy, Sunita Sarawagi, Steven Euijong Whang, and Xiao Yu, “Discovering structure in the universe of a.ribute names”, WWW’16
- Fangbo Tao, Honglei Zhuang, Chi Wang Yu, Qi Wang, Taylor Cassidy, Lance Kaplan, Clare Voss, Jiawei Han, “Multi-Dimensional, Phrase-Based Summarization in Text Cubes", Data Eng. Bulletin 39(3), Sept. 2016

PTE: Predictive Text Embedding

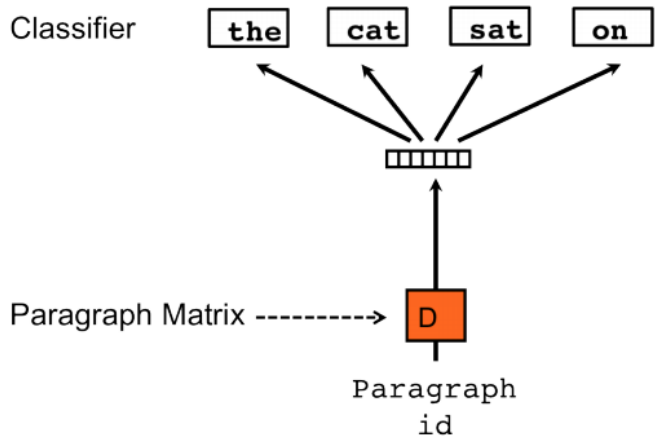
- Text Representation
  - Learning meaningful text representations is important for various machine learning tasks



Text Classification
Text Clustering
Retrieval

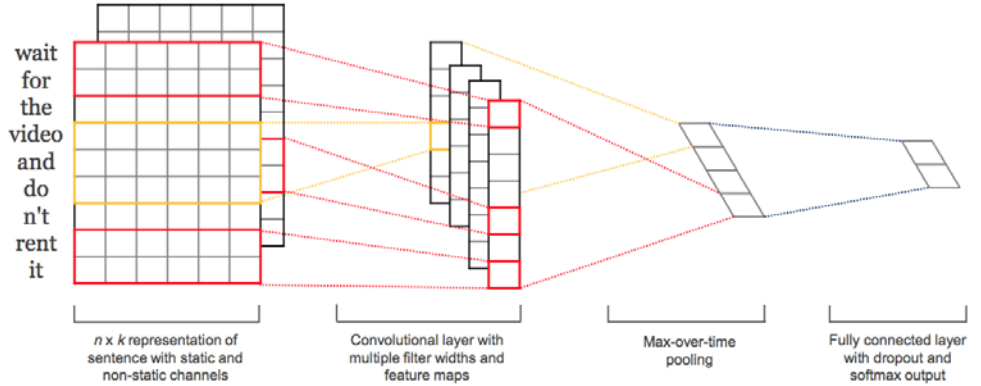
  - Bag of words:
    - Sparsity
    - Ignore the relatedness between different words

Recent Studies



- **Distributed Representations**
  - Embed text into a low-dimensional space
  - Word2vec, Paragraph Vector
- Strength:
  - Low-dimensional vectors; similar texts have similar vectors; efficient
- Weakness:
  - Totally unsupervised; Can't guide the training

- **Convolutional Neural Network**
  - Used for text classification
  - The hidden layer can be used for text representation
- Strength: High accuracy
- Weakness: Totally supervised;
  - Slow to train; Training is very tricky

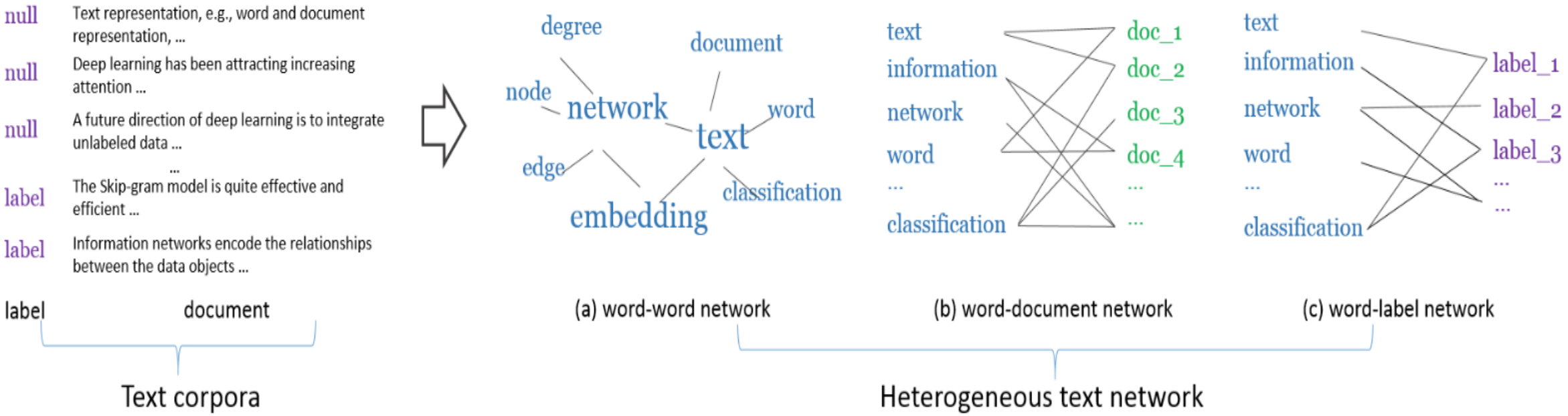


Effective Text Representations

- Desired Features for Effective Text Representation
  - Semi-supervised
    - Use labeled data to guide the training and boost the performance
    - Learn from massive unlabeled text data
  - Low-dimensional vector representations & Efficient
- How to realize such an effective representation?
  - A prerequisite of learning good text representations is to train effective word representations
    - Texts consist of words: To understand a text, we should first know the words well
  - Learn effective word representations
    - Distributional Hypothesis
      - words that occur in the same contexts tend to have similar meanings
    - Use multiple contexts to train word representations

The PTE Method

- Three different contexts:
  - Unsupervised: word document
  - Supervised: label
- Use Heterogeneous Information Networks to represent the co-occurrence relationship between words and contexts



Learn Word Embedding from Hetero. Info. Networks

- For a single context:
  - $O_c = \sum_{w,c} \hat{P}_{wc} log\sigma(\vec{w} \cdot \vec{c}) + K \sum_{w,c} \hat{P}_w \hat{P}_c log\sigma(-\vec{w} \cdot \vec{c})$
- For all three contexts:
  - $O_{pte} = O_{word} + O_{doc} + O_{label}$

Joint Training

Algorithm 1: Joint training.

Data: G_{ww}, G_{wd}, G_{wl}, number of samples T, number of negative samples K.
Result: word embeddings $\vec{w}$.
while $iter \leq T$ **do**
- sample an edge from E_{ww} and draw K negative edges, and update the word embeddings;
- sample an edge from E_{wd} and draw K negative edges, and update the word and document embeddings;
- sample an edge from E_{wl} and draw K negative edges, and update the word and label embeddings;

end

Pre-training + Fine-tuning

Algorithm 2: Pre-training + Fine-tuning.

Data: G_{ww}, G_{wd}, G_{wl}, number of samples T, number of negative samples K.
Result: word embeddings $\vec{w}$.
while $iter \leq T$ **do**
- sample an edge from E_{ww} and draw K negative edges, and update the word embeddings;
- sample an edge from E_{wd} and draw K negative edges, and update the word and document embeddings;

end
while $iter \leq T$ **do**
- sample an edge from E_{wl} and draw K negative edges, and update the word and label embeddings;

end

Infer Text Embedding

- Taking average of word embedding
  - For $d = [w_1, w_2, \dots, w_n]$ $\quad \vec{d} = \frac{1}{n}\sum_{k=1}^{n} \overrightarrow{w_k}$
  - Ignore word order
  - Efficient and effective in most cases
- Using Recursive NN or Recurrent NN
  - Consider word order
  - Too slow

Experiments: Setup and Results

- Datasets

Table 1: Statistics of the Data Sets

	Long Documents							Short Documents		
Name	20NG	WIKI	IMDB	CORPORATE	ECONOMICS	GOVERNMENT	MARKET	DBLP	MR	TWITTER
Train	11,314	1,911,617*	25,000	245,650	77,242	138,990	132,040	61,479	7,108	800,000
Test	7,532	21,000	25,000	122,827	38,623	69,496	66,020	20,000	3,554	400,000
|V|	89,039	913,881	71,381	141,740	65,254	139,960	64,049	22,270	17,376	405,994
#Avg.Length	305.77	672.56	231.65	102.23	145.10	169.07	119.83	9.51	22.02	14.36
#Labels	20	7	2	18	10	23	4	6	2	2

*In the WIKI data, only 42,000 documents are labeled.

- Task:
  - Text classification: Metric: Macro-F1 Micro-F1(accuracy)
- Baselines: BOW, Skip-Gram, PVDM, PVDBOW, CNN

Experimental Results

- Results on Long and short documents

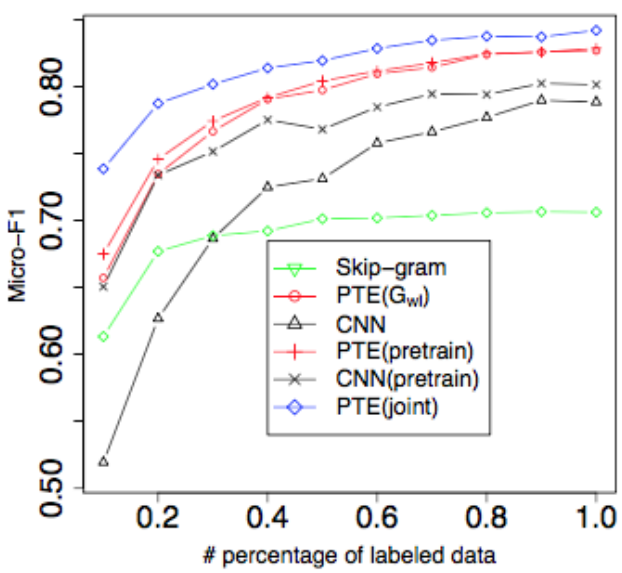
Table 2: Results of text classification on long documents.

		20NG		Wikipedia		IMDB	
Type	Algorithm	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1
Word	BOW	80.88	79.30	79.95	80.03	86.54	86.54
Unsupervised Embedding	Skip-gram	70.62	68.99	75.80	75.77	85.34	85.34
	PVDBOW	75.13	73.48	76.68	76.75	86.76	86.76
	PVDM	61.03	56.46	72.96	72.76	82.33	82.33
	LINE(G_{ww})	72.78	70.95	77.72	77.72	86.16	86.16
	LINE(G_{wd})	79.73	78.40	80.14	80.13	89.14	89.14
	LINE($G_{ww}+G_{wd}$)	78.74	77.39	79.91	79.94	89.07	89.07
Predictive Embedding	CNN	78.85	78.29	79.72	79.77	86.15	86.15
	CNN(pretrain)	80.15	79.43	79.25	79.32	89.00	89.00
	PTE(G_{wl})	82.70	81.97	79.00	79.02	85.98	85.98
	PTE($G_{ww}+G_{wl}$)	83.90	83.11	81.65	81.62	89.14	89.14
	PTE($G_{wd}+G_{wl}$)	**84.39**	**83.64**	82.29	82.27	89.76	89.76
	PTE(pretrain)	82.86	82.12	79.18	79.21	86.28	86.28
	PTE(joint)	84.20	83.39	**82.51**	**82.49**	**89.80**	**89.80**

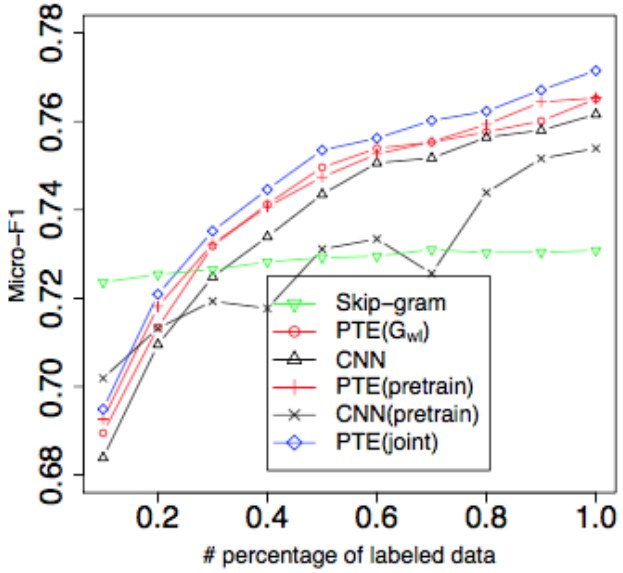
Table 4: Results of text classification on short documents.

		DBLP		MR		Twitter	
Type	Algorithm	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1
Word	BOW	75.28	71.59	71.90	71.90	75.27	75.27
Unsupervised Embedding	Skip-gram	73.08	68.92	67.05	67.05	73.02	73.00
	PVDBOW	67.19	62.46	67.78	67.78	71.29	71.18
	PVDM	37.11	34.38	58.22	58.17	70.75	70.73
	LINE(G_{ww})	73.98	69.92	71.07	71.06	73.19	73.18
	LINE(G_{wd})	71.50	67.23	69.25	69.24	73.19	73.19
	LINE($G_{ww}+G_{wd}$)	74.22	70.12	71.13	71.12	73.84	73.84
Predictive Embedding	CNN	76.16	73.08	72.71	72.69	**75.97**	**75.96**
	CNN(pretrain)	75.39	72.28	68.96	68.87	75.92	75.92
	PTE(G_{wl})	76.45	72.74	73.44	73.42	73.92	73.91
	PTE($G_{ww}+G_{wl}$)	76.80	73.28	72.93	72.92	74.93	74.92
	PTE($G_{wd}+G_{wl}$)	**77.46**	**74.03**	73.13	73.11	75.61	75.61
	PTE(pretrain)	76.53	72.94	73.27	73.24	73.79	73.79
	PTE(joint)	77.15	73.61	**73.58**	**73.57**	75.21	75.21

Performance & Visualization Results

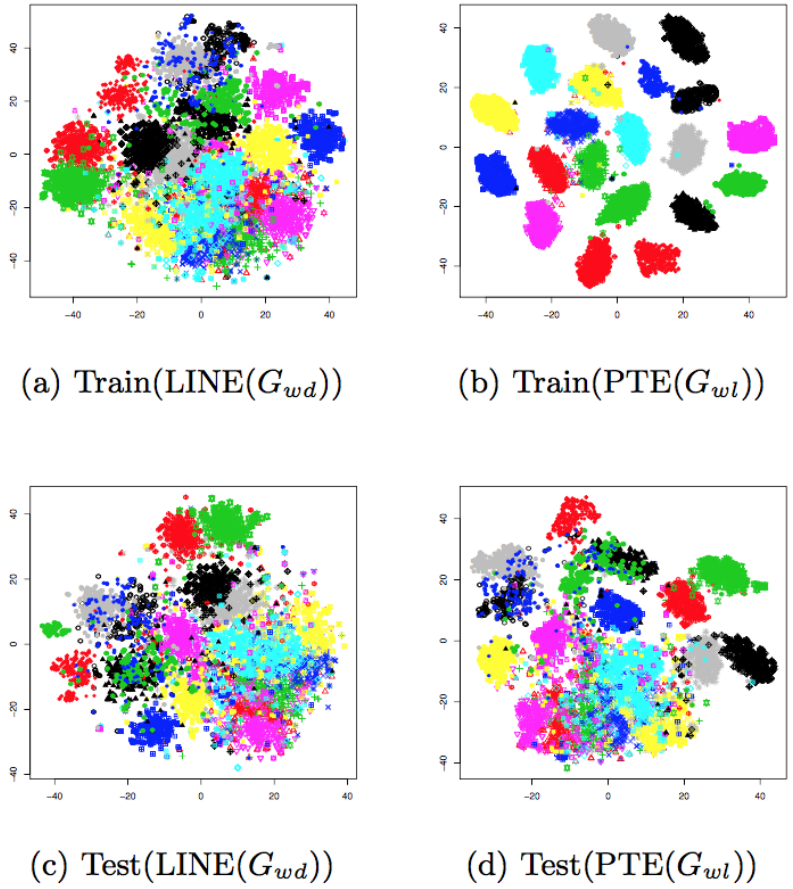


(a) 20NG



(b) DBLP

Performance comparison on percentage of labeled data



(a) Train(LINE(G_{wd}))

(b) Train(PTE(G_{wl}))

(c) Test(LINE(G_{wd}))

(d) Test(PTE(G_{wl}))

Visualization results