Learning Large-Scale Social Knowledge Graphs

Zhilin Yang, Jie Tang
Dept. Computer Science, Tsinghua University

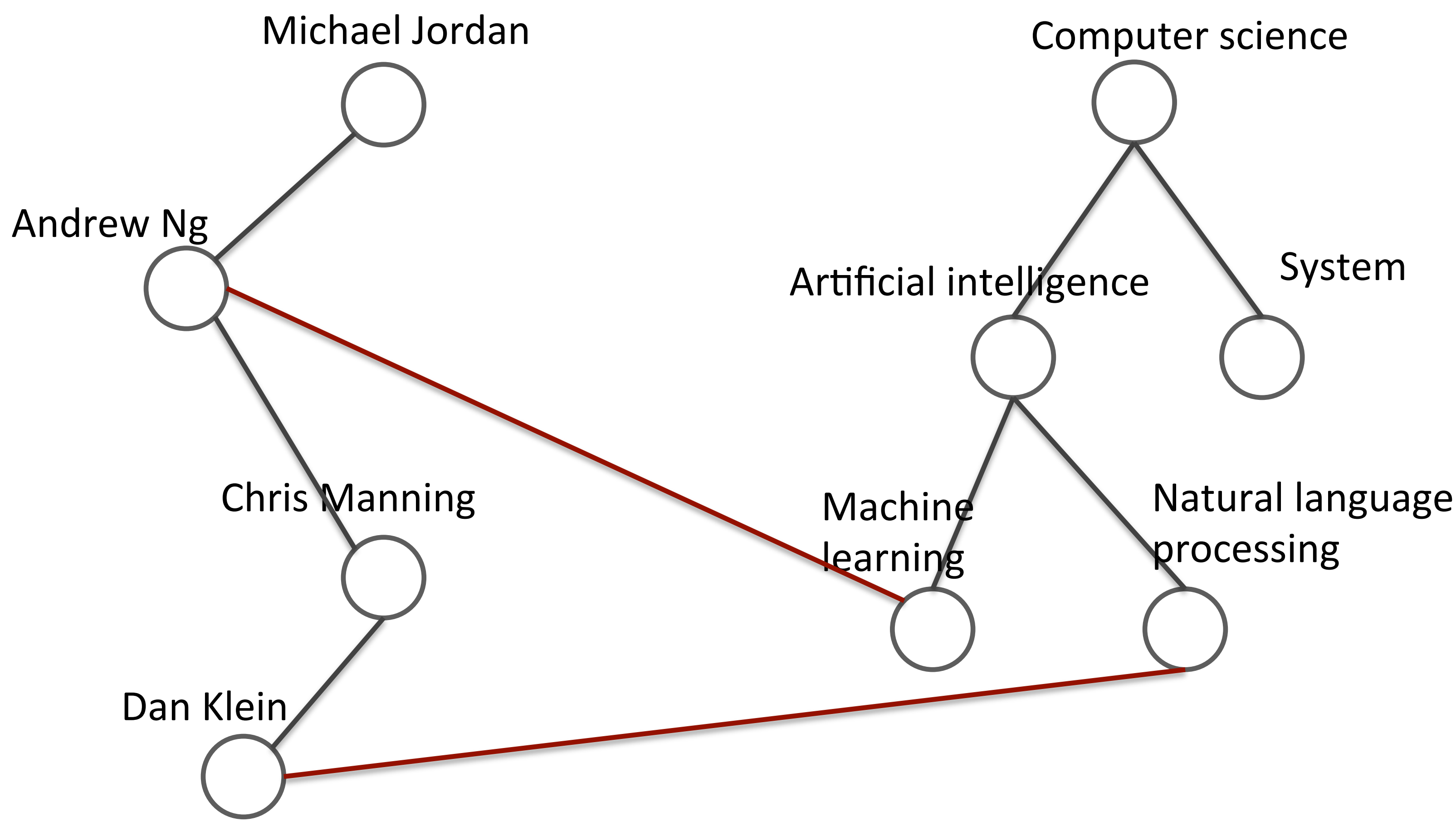
Large-scale social networks

- Facebook
  - 1.4 billion active users in Quarter 1, 2015
  - Tens of millions of posts per day
- AMiner
  - 39 million researchers
  - 79 million papers
- Large-Scale social networks are big information networks!

Large-scale collective knowledge

- Freebase
  - 44 million entities
  - 2.4 billion facts
- YAGO2
  - 10 million entities
  - 120 million facts
- Wikipedia
  - 35 million entities
  - 2 million categories

Bridge the gap



Social Network **Collective Knowledge**

Bridge the gap

- Social knowledge graph
- Why?
  - Better mine large volume of information
  - Better user understanding and recommendation
  - Better search

What we've done

- Propose an algorithm GenVector to learn large-scale social knowledge graph
  - Weakly supervision based on unsupervised techniques
  - Multi-source Bayesian embedding model
- Online deployment
  - Online service on AMiner.org
  - Online AB-test

Key features

- Large-scale
  - **38,049,189** researchers (AMiner)
  - **74,050,920** papers (AMiner)
  - **20,552,544,886** bytes corpus (Wikipedia full text)
  - **35,415,011** entities (Wikipedia)

Key features

- Large-scale
- Fast
  - Implementation optimization for a **60 times** speedup
  - From 3 hours per iteration to 3 minutes

Key features

- Large-scale
- Fast
- Accurate
  - Offline test: 4% to 15%+ better than state-of-the-arts
  - Online test: decrease the error rate by **67%**

Key features

- Large-scale
- Fast
- Accurate
- Novel
  - Bridge the gap between social networks and collective knowledge
  - Bridge the gap between topic models and word/network embedding

Key features

- Large-scale
- Fast
- Accurate
- Novel
- Real-world impact
  - Online deployment on AMiner
  - **183,876** visits ever since

Key features

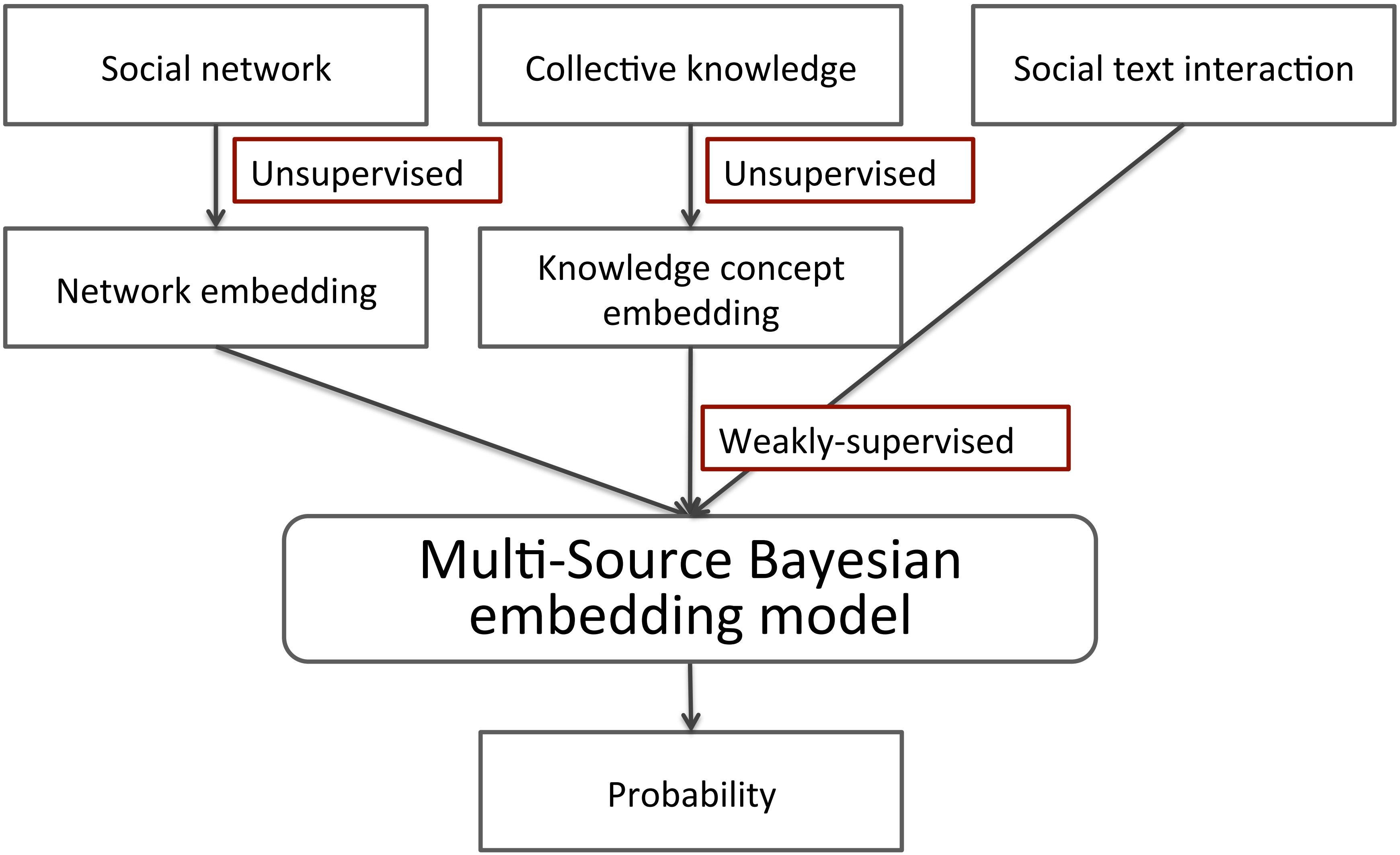
- Large-scale
- Fast
- Accurate
- Novel
- Real-world impact

How did we make it?

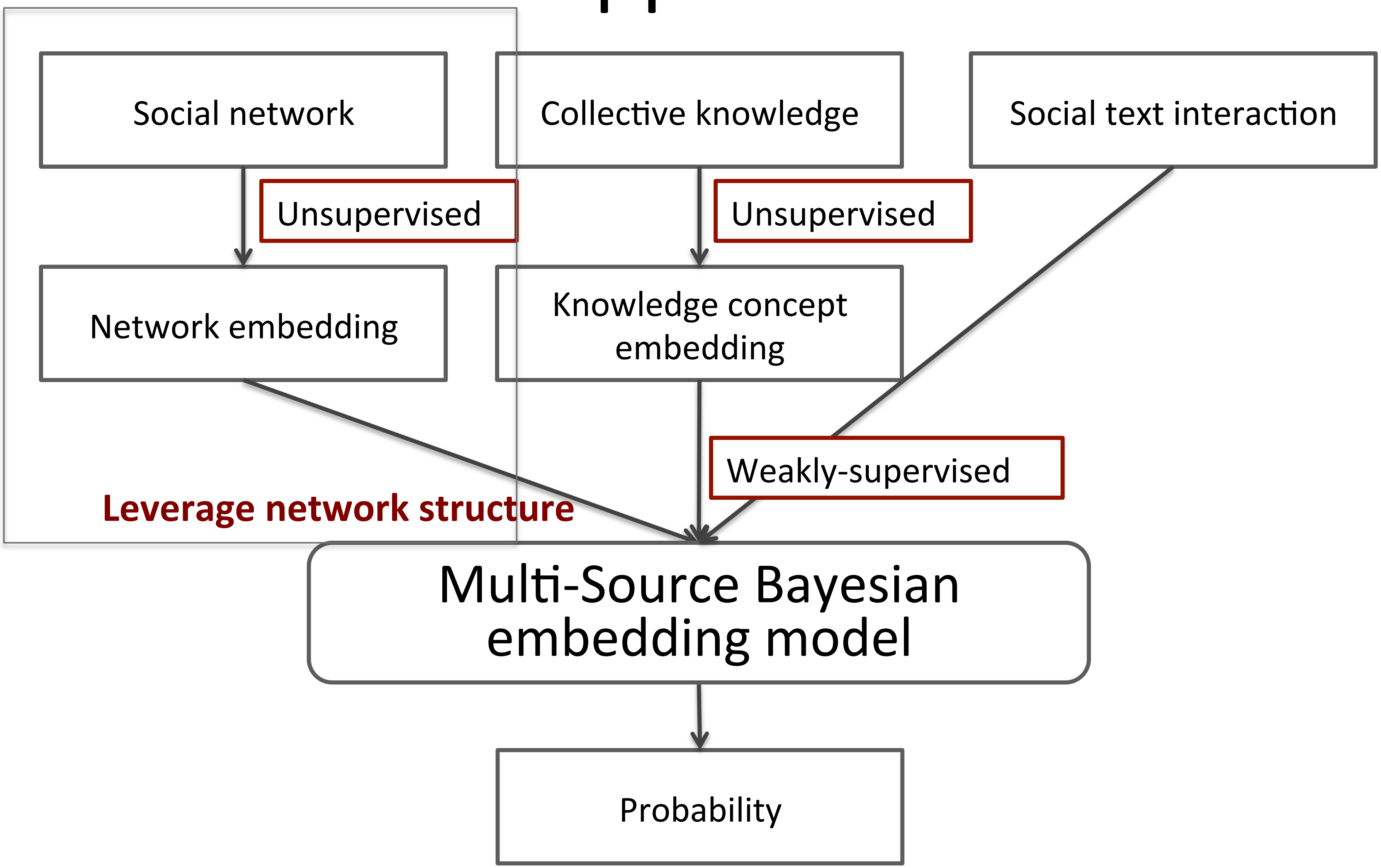
Problem formulation

- Input
  - A social network
  - A collective knowledge source
  - Social text interaction
- Output
  - For each social network vertex, output related knowledge concepts as a ranked list

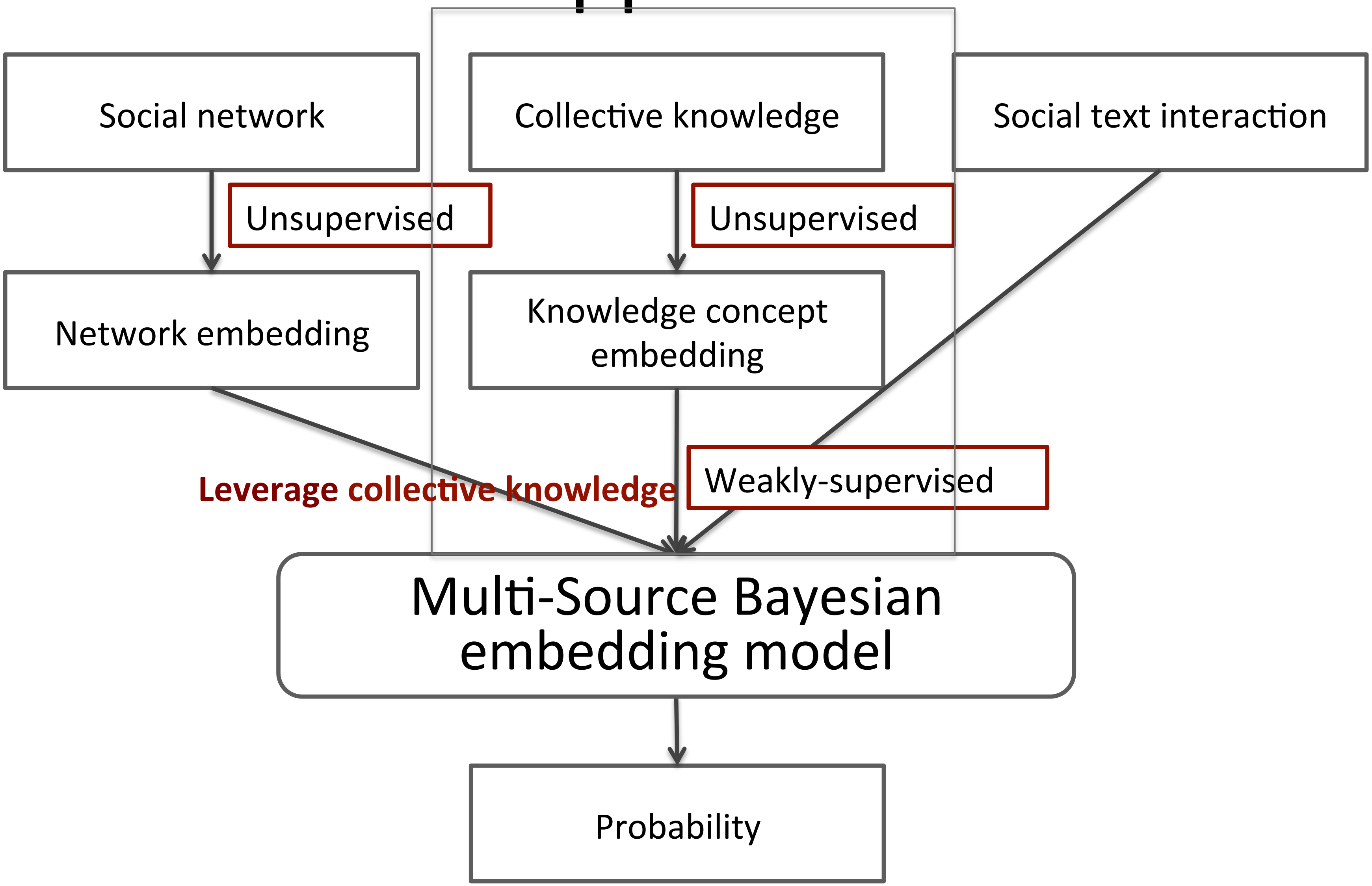
Approach


Social network
Collective knowledge
Social text interaction
Unsupervised
Unsupervised
Network embedding
Knowledge concept embedding
Weakly-supervised
Multi-Source Bayesian embedding model
Probability

Approach

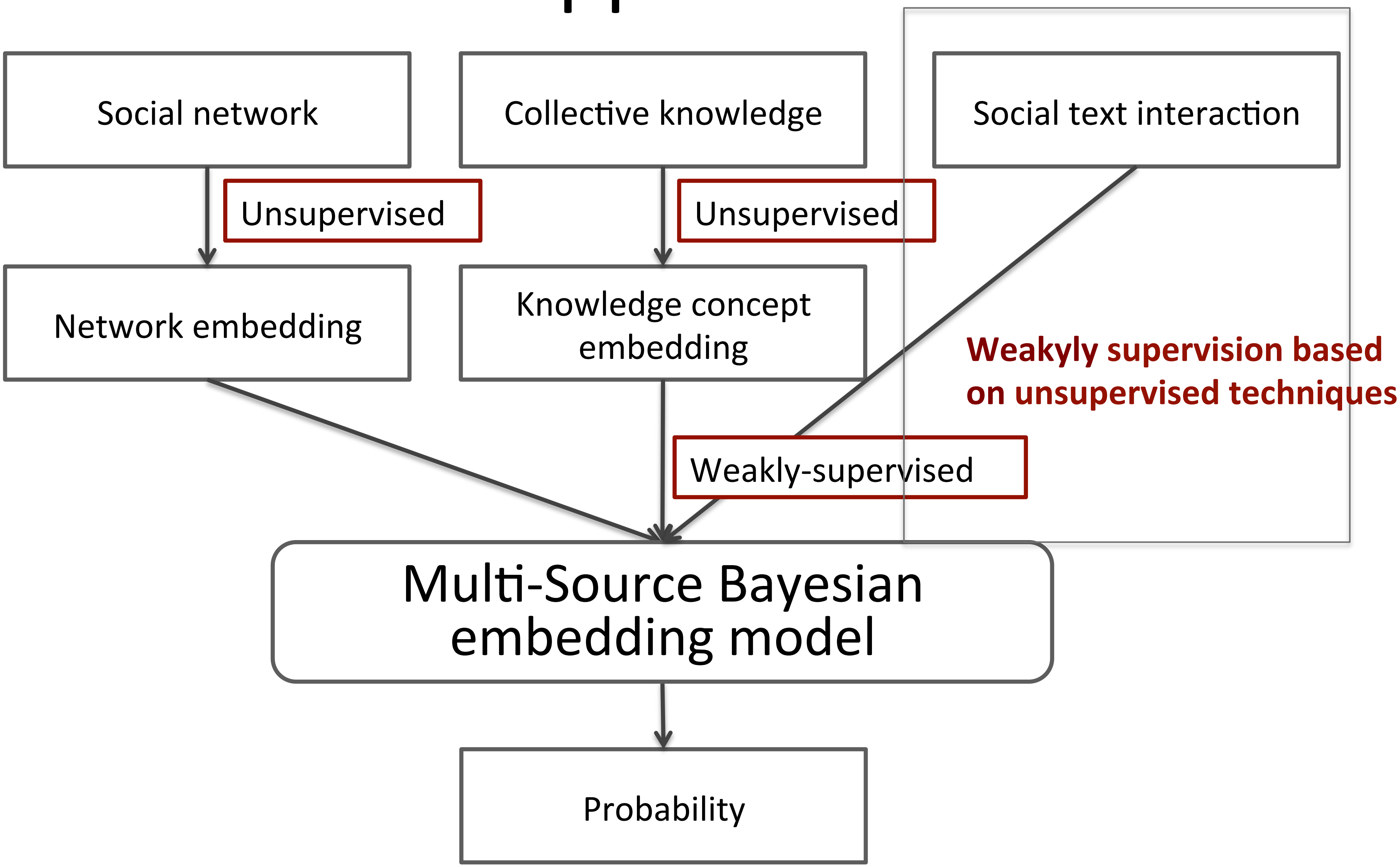

Social network
Collective knowledge
Social text interaction
Unsupervised
Unsupervised
Network embedding
Knowledge concept embedding
Weakly-supervised
Leverage network structure
Multi-Source Bayesian embedding model
Probability

Approach



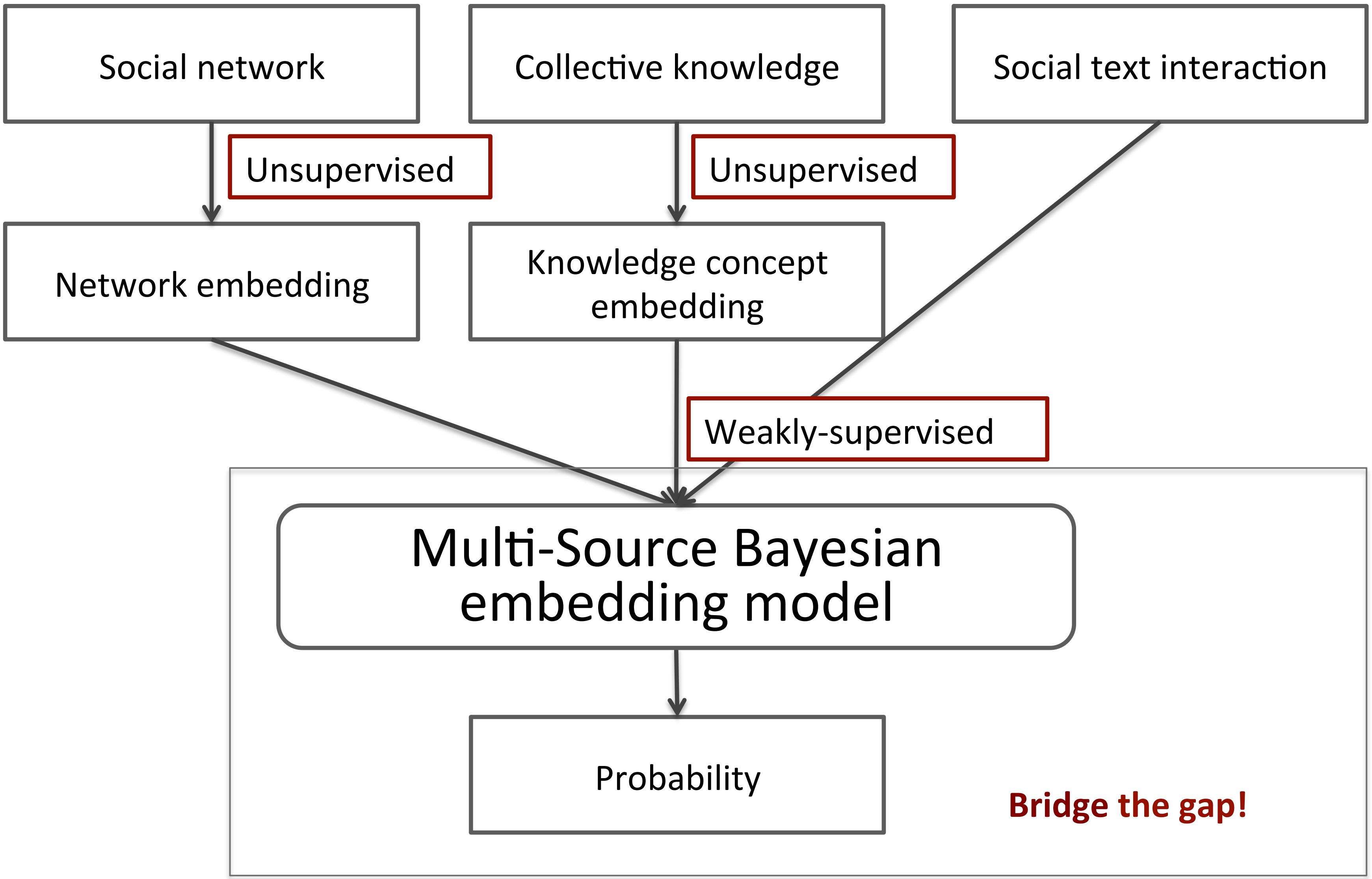
Social network
Collective knowledge
Social text interaction
Unsupervised
Unsupervised
Network embedding
Knowledge concept embedding
Weakly-supervised
Leverage collective knowledge
Multi-Source Bayesian embedding model
Probability

Approach

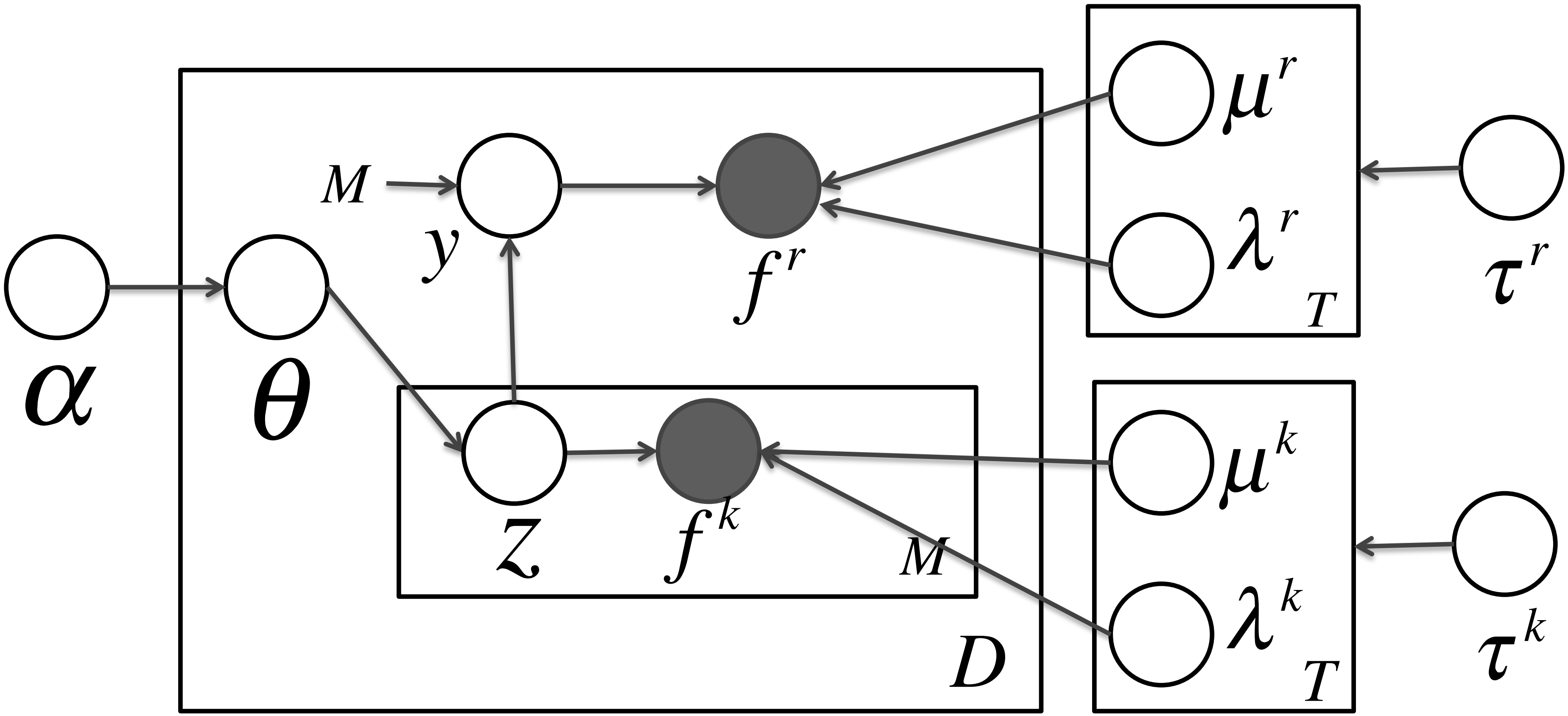


Social network
Collective knowledge
Social text interaction
Unsupervised
Unsupervised
Network embedding
Knowledge concept embedding
Weakyly supervision based on unsupervised techniques
Weakly-supervised
Multi-Source Bayesian embedding model
Probability

Approach

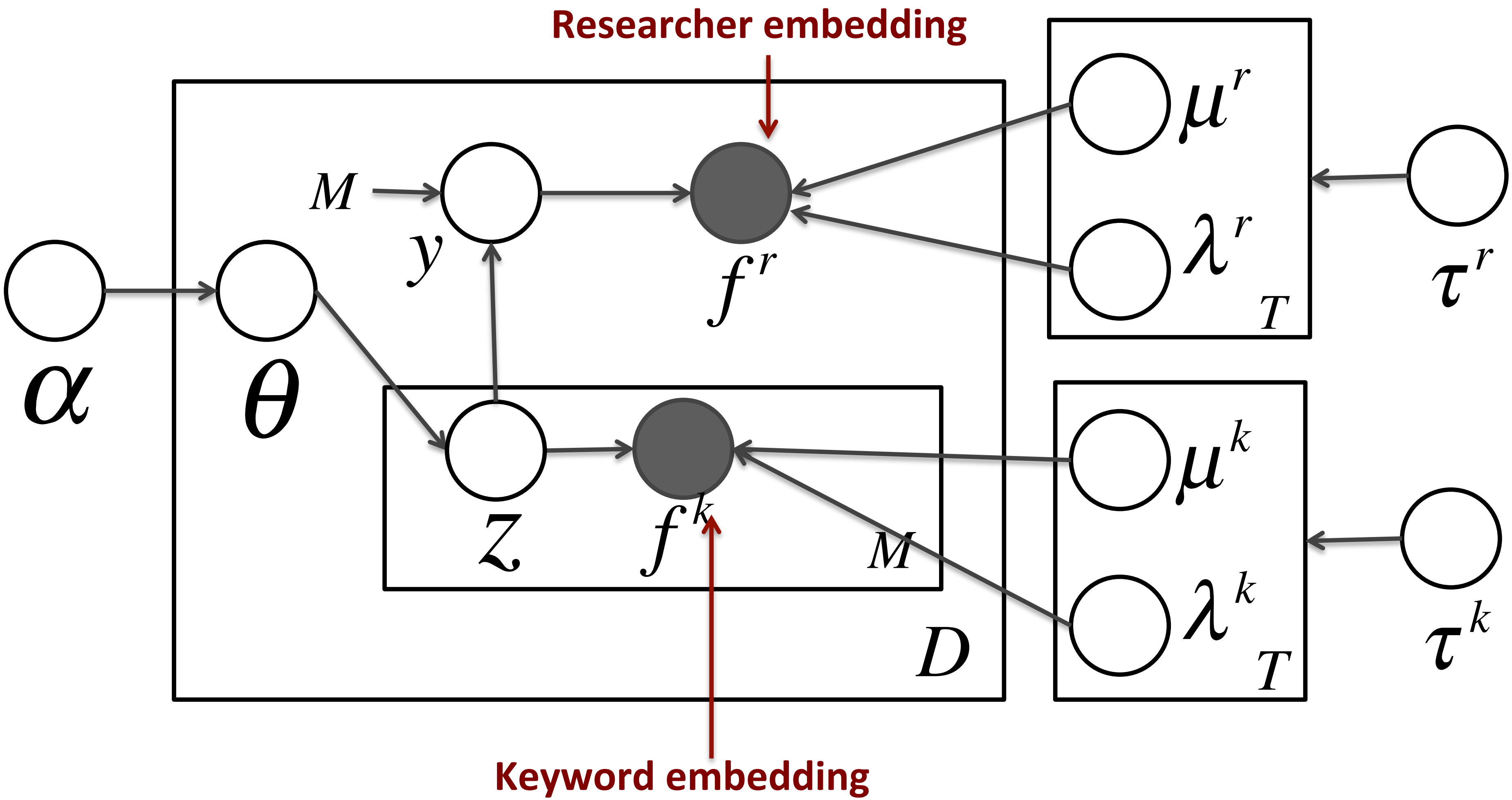

Social network
Collective knowledge
Social text interaction
Unsupervised
Unsupervised
Network embedding
Knowledge concept embedding
Weakly-supervised
Multi-Source Bayesian embedding model
Probability
Bridge the gap!

Multi-source Bayesian embeddings



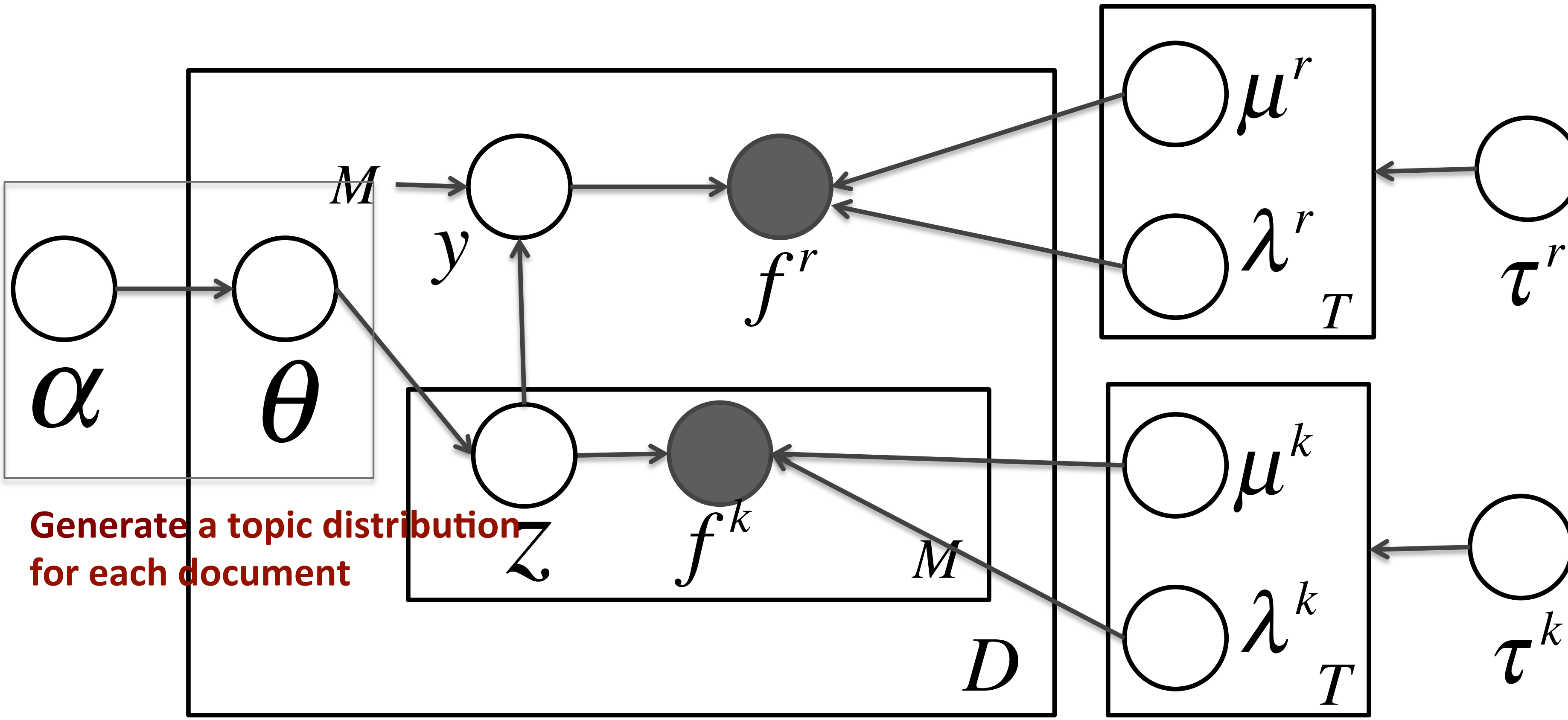
Number of documents: D, number of topics: T, dimension of embedding: E

Multi-source Bayesian embeddings



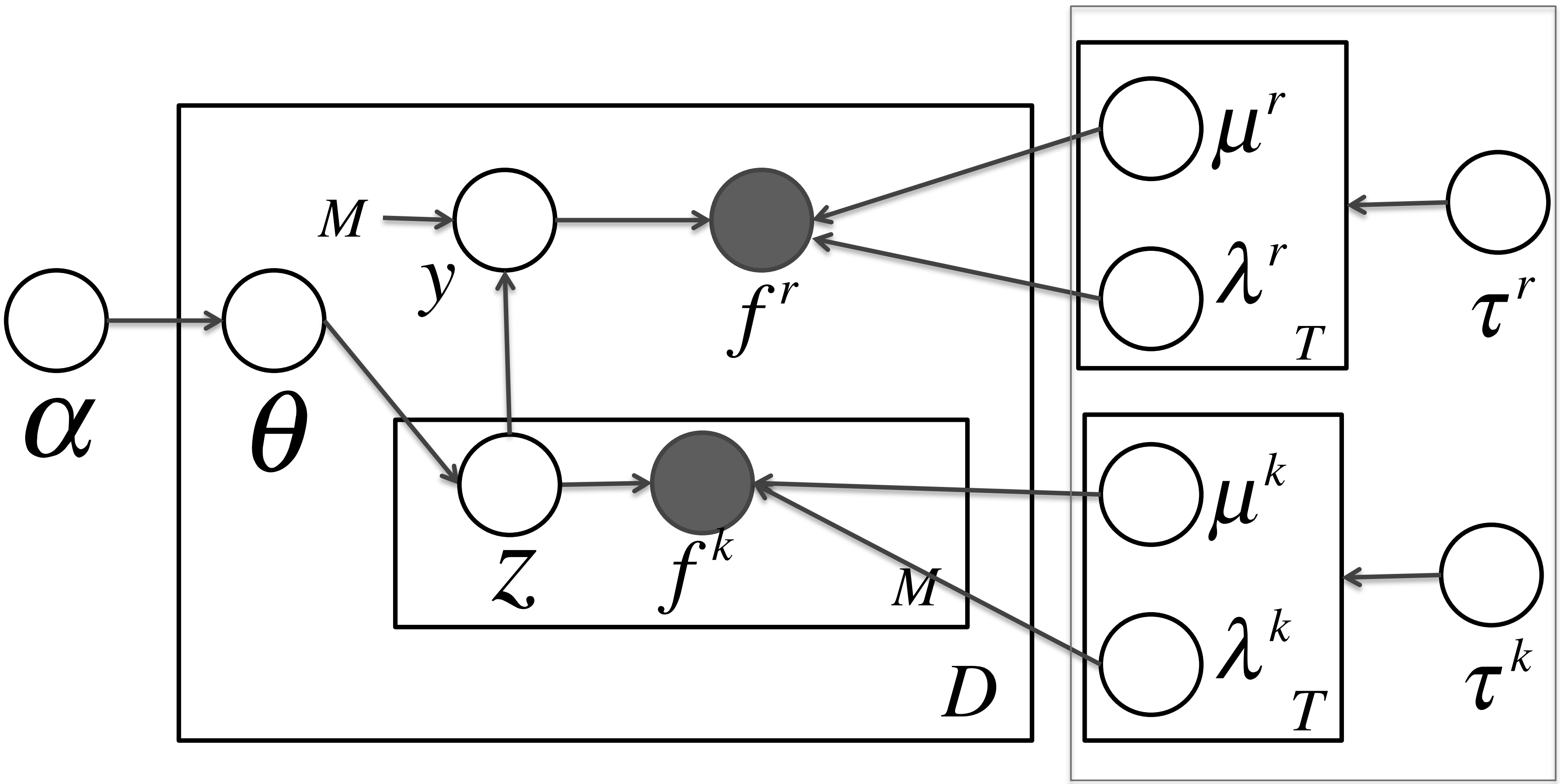
Number of documents: D, number of topics: T, dimension of embedding: E

Multi-source Bayesian embeddings



Number of documents: D, number of topics: T, dimension of embedding: E

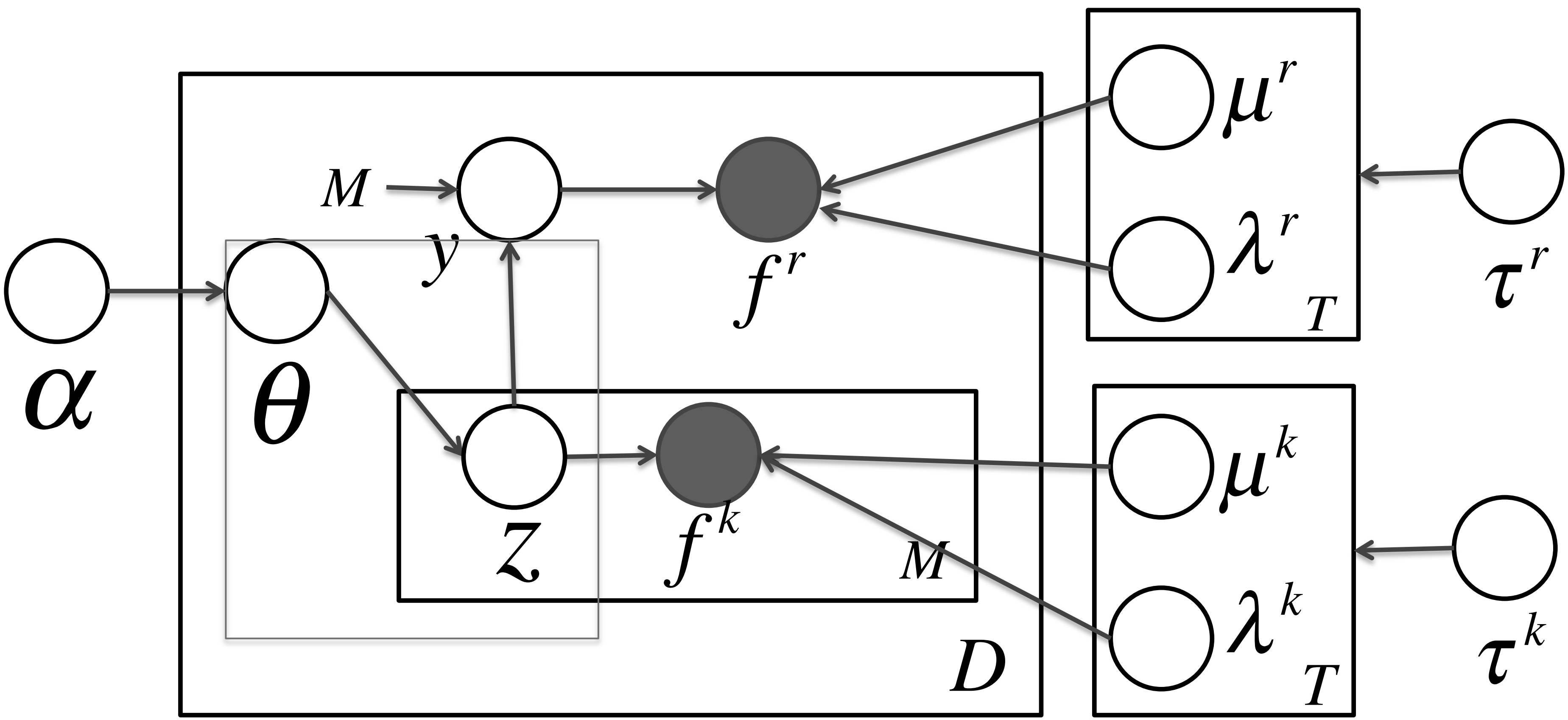
Multi-source Bayesian embeddings



Generate Gaussian distribution for each topic

Number of documents: D, number of topics: T, dimension of embedding: E

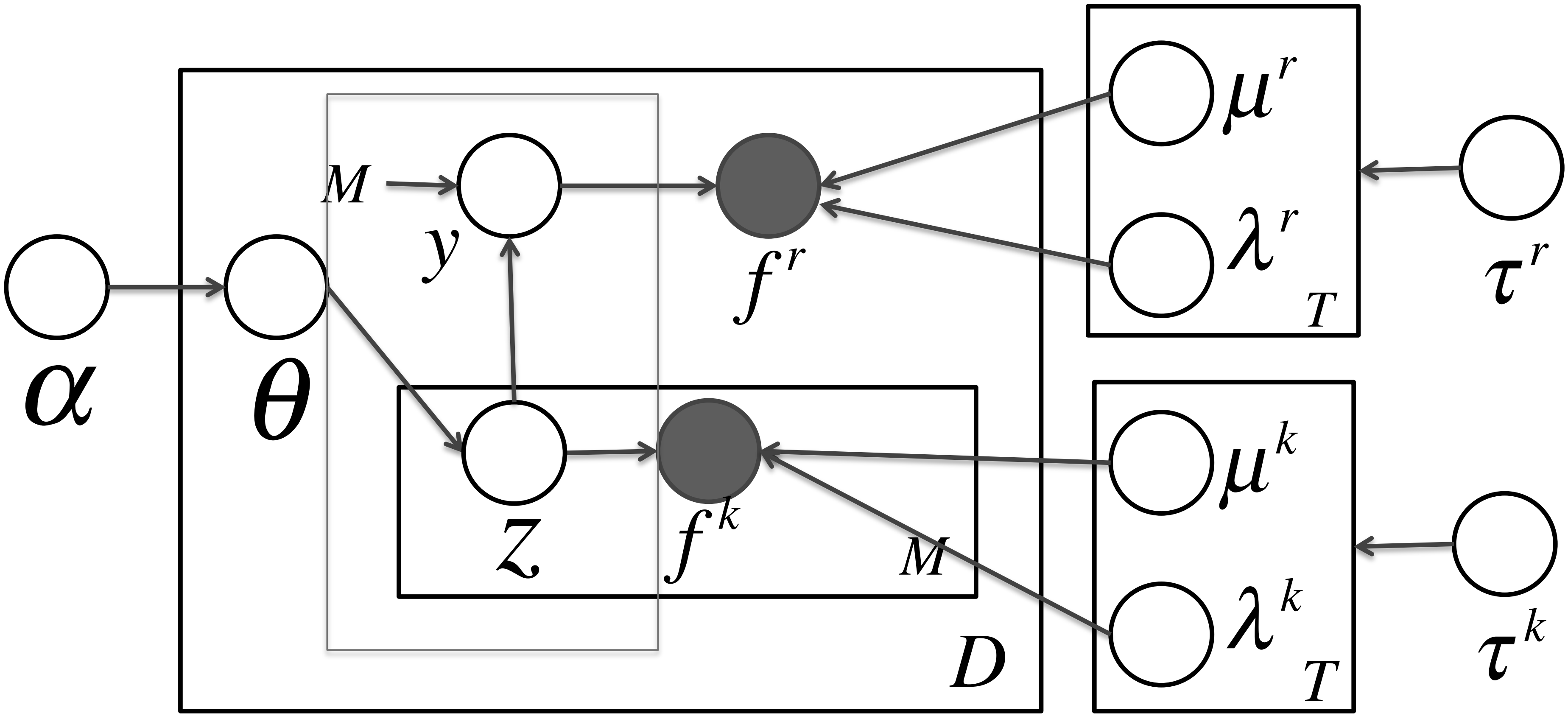
Multi-source Bayesian embeddings



Generate the topic for each word

Number of documents: D, number of topics: T, dimension of embedding: E

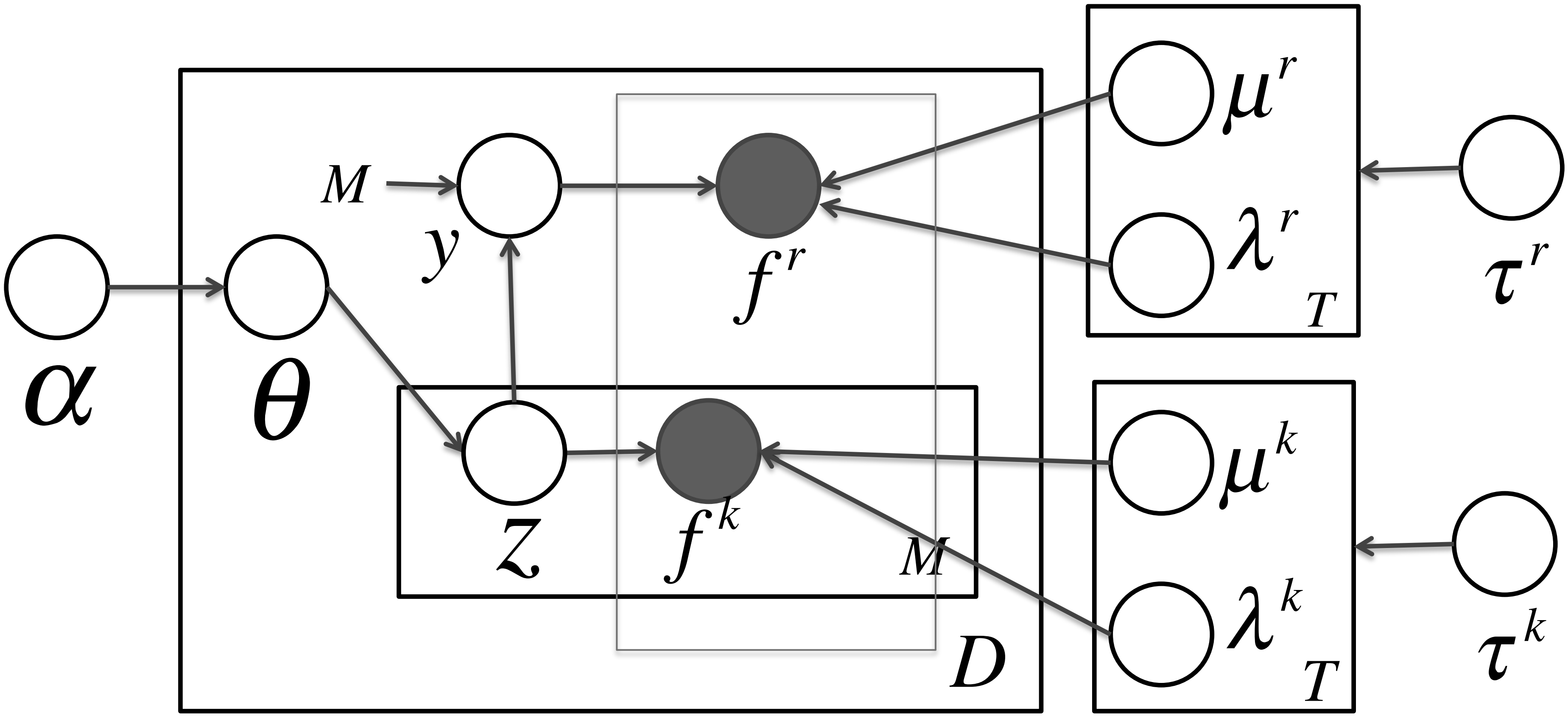
Multi-source Bayesian embeddings



Generate the topic for each user

Number of documents: D, number of topics: T, dimension of embedding: E

Multi-source Bayesian embeddings



Generate embeddings for keywords and users

Number of documents: D, number of topics: T, dimension of embedding: E

Inference

- Collapsed Gibbs sampling
- The joint probability

$$
\begin{aligned}
p(\theta,\mu^r,\lambda^r,\mu^k,\lambda^k,z,y,f^r,f^k;\alpha,\tau^r,\tau^k) \quad = \quad & p(\theta;\alpha)p(\mu^r,\lambda^r;\tau^r)p(\mu^k,\lambda^k;\tau^k) \\
& p(z|\theta)p(f^k|z,\mu^k,\lambda^k)p(f^r|y,\mu^r,\lambda^r)p(y|z)
\end{aligned}
$$

Inference

Dirichlet distribution

$$p(\theta_d; \alpha) = \frac{1}{\Delta(\alpha)} \prod_{t=1}^{T} \theta_{dt}^{\alpha_t - 1}$$

Normal Gamma distribution

$$p(\mu_{te}^r, \lambda_{te}^r; \tau_e^r = \{\mu_0, \lambda_0, \alpha_0, \beta_0\}) = \frac{\beta_0^{\alpha_0} \sqrt{\lambda_0}}{\Gamma(\alpha_0) \sqrt{2\pi}} \lambda_{te}^{r\ \alpha_0 - 1/2} e^{\beta_0 \lambda_{te}^r} e^{-\frac{\lambda_0 \lambda_{te}^r (\mu_{te}^r - \mu_0)^2}{2}}$$

$$p(\mu_{te}^k, \lambda_{te}^k; \tau_e^k = \{\mu_0, \lambda_0, \alpha_0, \beta_0\}) = \frac{\beta_0^{\alpha_0} \sqrt{\lambda_0}}{\Gamma(\alpha_0) \sqrt{2\pi}} \lambda_{te}^{k\ \alpha_0 - 1/2} e^{\beta_0 \lambda_{te}^k} e^{-\frac{\lambda_0 \lambda_{te}^k (\mu_{te}^k - \mu_0)^2}{2}}$$

Inference

Generating topics

$$p(z_{dm}|\theta_d) = \theta_{d z_{dm}}$$

$$p(y_d|z_d) = \frac{\sum_{m=1}^{M_d} \mathbb{I}(z_{dm} = y_d) + l}{M_d + Tl}$$

Inference

Generating embeddings

$$p(f_{dm}^k | z_{dm}, \mu^k, \lambda^k) = \frac{1}{\sqrt{2\pi}} \sqrt{\lambda^k} e^{-\frac{\lambda^k}{2}(f_{dm}^k - \mu^k)^2}$$

$$p(f_{dm}^r | z_{dm}, \mu^r, \lambda^r) = \frac{1}{\sqrt{2\pi}} \sqrt{\lambda^r} e^{-\frac{\lambda^r}{2}(f_{dm}^r - \mu^r)^2}$$

Inference

Full Conditional

$$p(y_d = t | y_{-d}, z, f^r, f^k) \propto (n_d^t + l) \prod_{e=1}^{E^r} G'(f^r, y, t, e, \tau^r, d)$$

$$p(z_{dm} = t | z_{-dm}, y, f^r, f^k) \propto (n_d^{y_d} + l)(n_d^t + \alpha_t) \prod_{e=1}^{E^k} G'(f^k, z, t, e, \tau^k, dm)$$

$$G'(f, y, t, e, \tau, d) = \frac{\Gamma(\alpha_n)}{\Gamma(\alpha_{n'})} \frac{\beta_{n'}^{\alpha_{n'}}}{\beta_n^{\alpha_n}} \left(\frac{\kappa_{n'}}{\kappa_n}\right)^{\frac{1}{2}} \frac{(2\pi)^{-n/2}}{(2\pi)^{-n'/2}}$$

Parameter update

$$\theta_d^t = \frac{n_d^t + \alpha_t}{\sum_{t=1}^{T}(n_d^t + \alpha_t)}$$

$$\mu_t^k = \frac{\kappa_0\mu_0 + n\bar{x}}{\kappa_0 + n}$$

$$\lambda_t^k = \alpha_n\beta_n^{-1} = \frac{\alpha_0 + n/2}{\beta_0 + \frac{1}{2}\sum_i(x_i - \bar{x})^2 + \frac{\kappa_0 n(\bar{x}-\mu_0)^2}{2(\kappa_0+n)}}$$

Embedding update

$$\frac{\partial L}{\partial f_{de}^{r}} = \sum_{t=1}^{T} -\lambda_{te}^{r}(f_{de}^{r} - \mu_{te}^{r})$$

$$\frac{\partial L}{\partial f_{we}^{k}} = \sum_{t=1}^{T} n_{w}^{t}(-\lambda_{te}^{k})(f_{we}^{k} - \mu_{te}^{k})$$

Learning framework

- Initialize
- Burn-in
  - Sample topics
- Sampling
  - Sample topics
  - Update parameters
  - Update embeddings

Experiments

- Comparison methods
  - GenVector: our method
  - GenVector-E: without embeddings
  - GenVector-M: without the model
  - GenVector-R: use weakly-supervision score only
  - AM-base: AMiner previous method
  - CountKG: sort by counts after KG matching
  - Author-topic: Author-topic model
  - NTN: Neural tensor network

Experiments: homepage matching

Methods	Precision@5
GenVector	**77.9402%**
GenVector-E	77.8548%
GenVector-M	65.5608%
GenVector-R	72.8549%
AM-base	73.8189%
CountKB	54.4832%
Author-topic	74.4397%
NTN	65.8911%

Experiments: LinkedIn skill maching

Methods	Precision@5
GenVector	**26.8468%**
GenVector-E	26.5765%
GenVector-M	24.6695%
GenVector-R	26.3063%
AM-base	24.5195%
CountKB	25.4954%
Author-topic	26.4864%
NTN	24.3243%

Experiments: human labeling bad cases

Methods	Precision@5
GenVector	**98.8%**
GenVector-R	**99.6%**
AM-base	81.2%
Author-topic	98.4%
NTN	92.8%

Online deployment

Upload

Jiawei Han (韩家炜)

Follow

Department of Computer Science, University of Illinois at Urbana-Champaign

Professor

(217) 333-6903

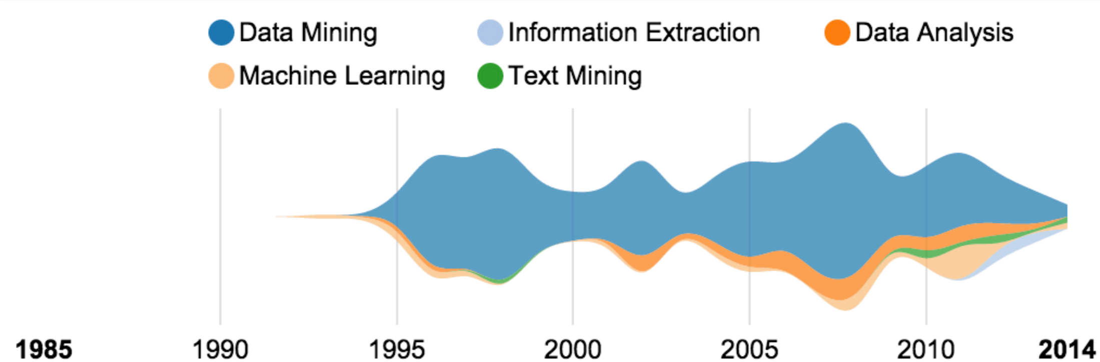
hanj@cs.uiuc.edu

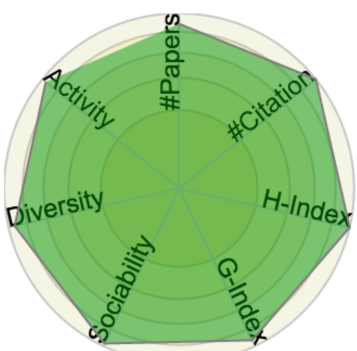
http://www.cs.uiuc.edu/~hanj/

External Links

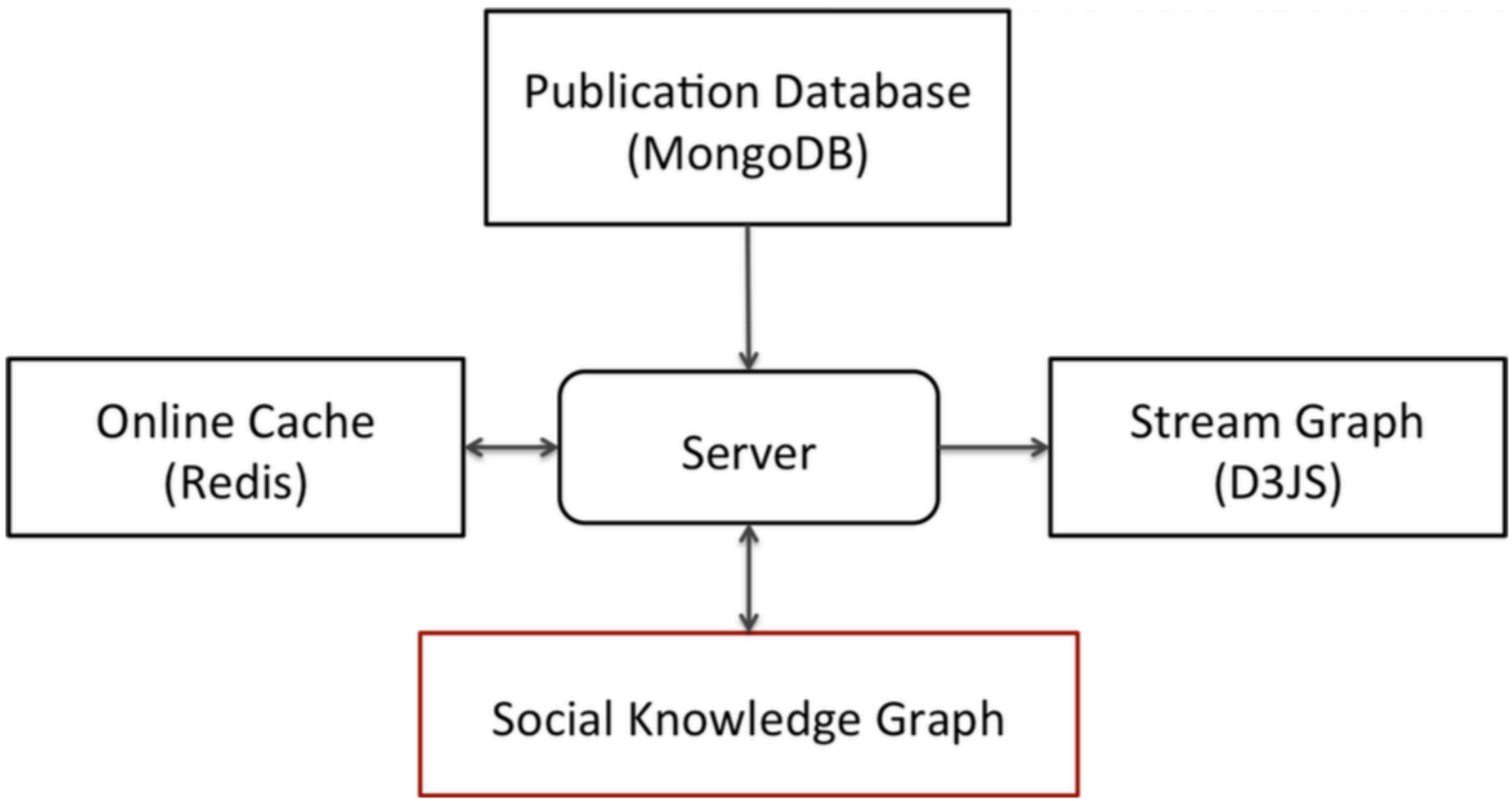
Update

Research Interests





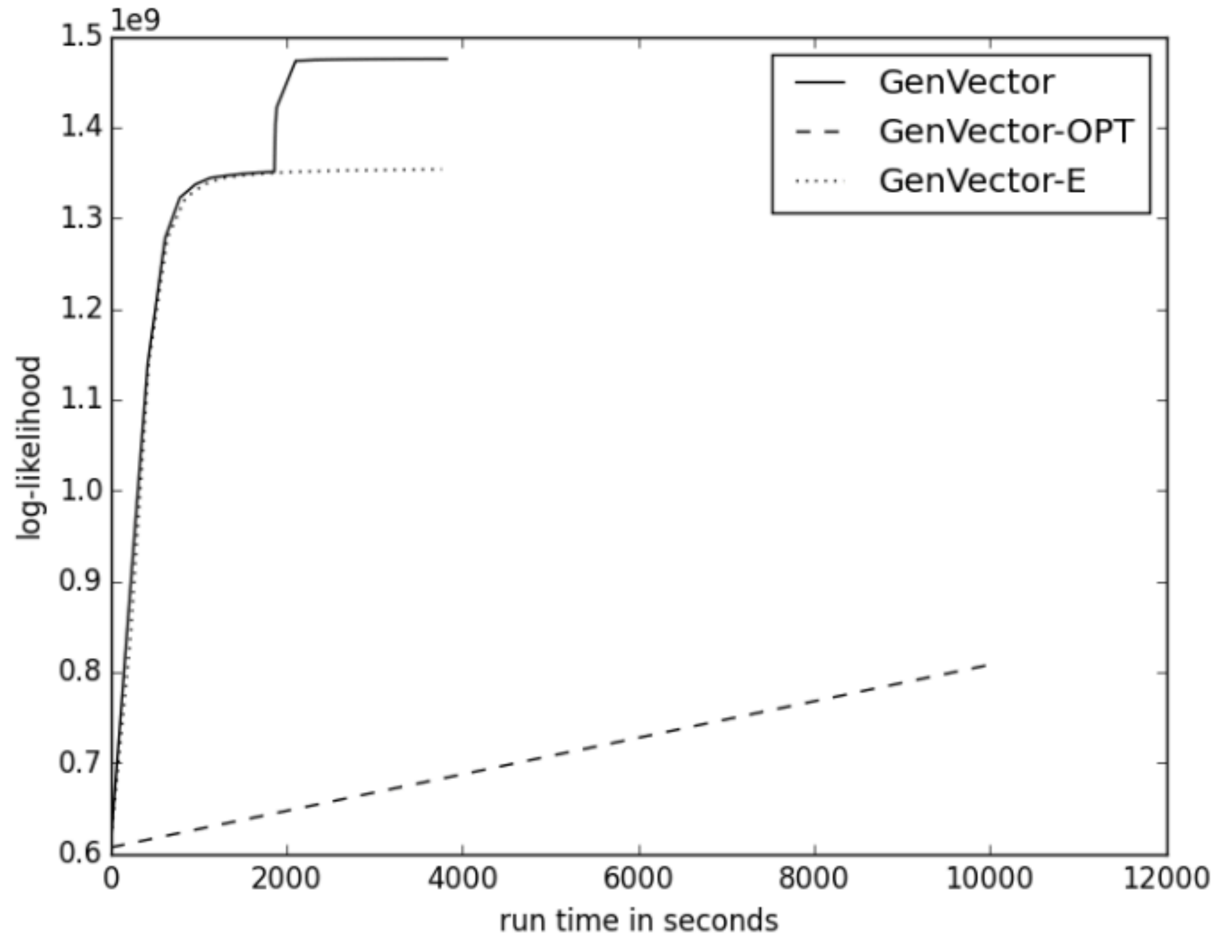
Online deployment



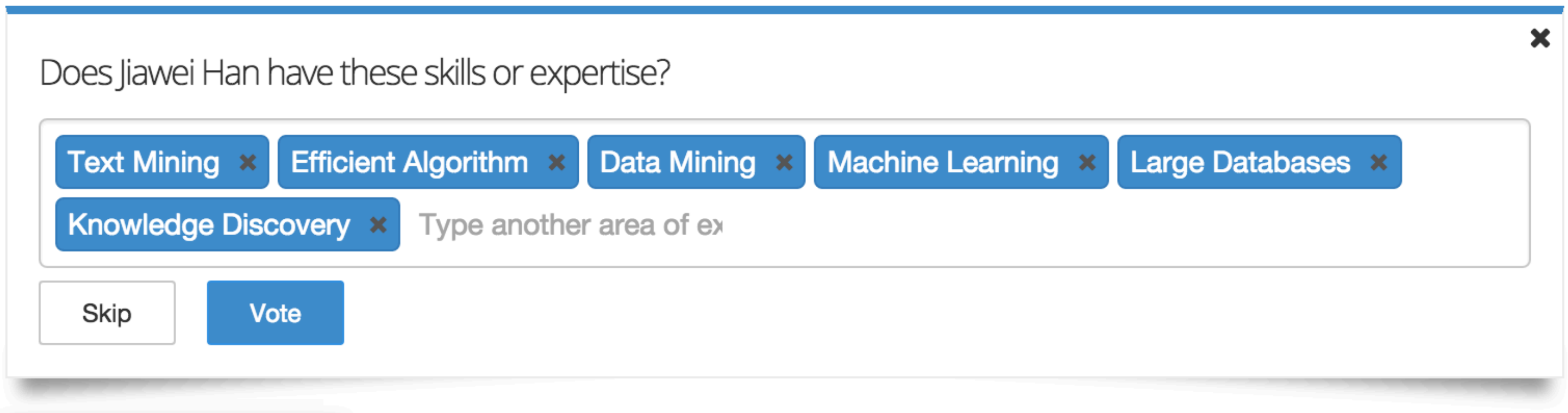
Implementation optimization

- Faster computation of G'()
- Faster computation of log, exp and pow
- Local variables instead of in-array access
- Multi-thread parallelization

Run time and convergence



Online AB-test



Leverage collective intelligence

-- evaluate the methods

-- leverage user feedback to improve the model

Online AB-test

Methods	Precision@10
GenVector	**96.67%**
AM-base	90.00%

Case study: Andrew Ng

GenVector	AM-base
Unsupervised learning	Challenging problem
Feature learning	Reinforcement learning
Bayesian networks	Autonomous helicopter
Reinforcement learning	Autonomous helicopter flight
Dimensionality reduction	Near-optimal planning

Case study: Dan Klein

GenVector	**AM-base**
Language models	Machine translation
Markov models	Word alignment
Probabilistic models	Bleu score
Natural language	Best result
Coreference resolution	Language model

Case study: Xiaoou Tang

GenVector	AM-base
Feature extraction	Face recognition
Image segmentation	Face image
Image matching	Novel approach
Image classification	Line drawing
Face recognition	Discriminant analysis

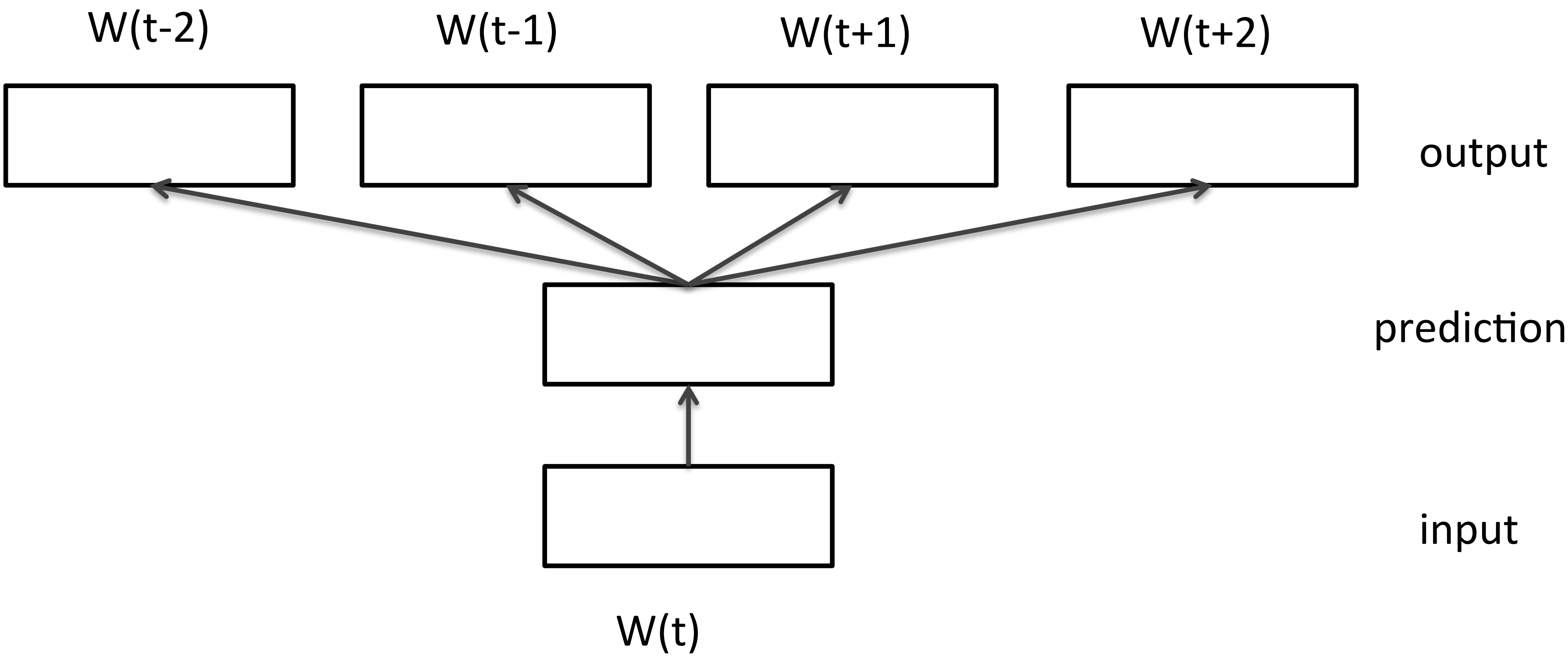
Take-away

- Large-scale
  - Link **38,049,189** researchers to **35,415,011** knowledge concepts
- Fast
  - **60** times speed up
- Accurate
  - Decrease the error rate by **67%** online
- Novel
  - Bridge social networks and collective knowledge
  - bridge topic models and network/word embedding
- Real-world impact
  - Online service with **183,876** visits

Appendix

Learning keyword embeddings

- Skip-gram



Learning keyword embeddings

- Skip-gram
  - Use the current keyword to predict the context
  - Objective function

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t)$$

$$p(w_O|w_I) = \frac{\exp\left({v'_{w_O}}^\top v_{w_I}\right)}{\sum_{w=1}^{W}\exp\left({v'_w}^\top v_{w_I}\right)}$$

Learning keyword embeddings

- Scan through all titles and abstracts
  - Extract n-grams according to Wikipedia concepts
- Replace all extracted n-grams in the Wikipedia corpus as a token
  - E.g., machine learning -> machine_learning
- Train a skip-gram model on the processed corpus

Learning network embeddings

- DeepWalk
  - Generate a random walk sequence from each node
  - Train a skip-gram model on the random walk sequence

Weakly supervision

- Given a researcher, extract all the keywords in his papers' titles, denoted as k1, k2, …, kn.
- Let ci be the count of the keyword ki in the author's papers' titles.
- Compute a score for each keyword ki

$$s_i = \sum_j c_j \cos_{i,j}$$

- Select top-k keywords as weakly-supervised information