TigerGraph | Graph Gurus Episode 28

In-Database Machine Learning Solution for Real-Time Recommendations

Today's Host

David Ronald

Director of Product Marketing

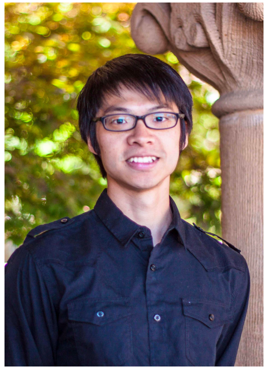
- 18+ years in tech industry
- Prior work in artificial intelligence, natural linguistic programming and telecommunications technology
- BSc in Applied Physics from Strathclyde University, MSc in Optoelectronic & Laser Devices from St Andrews

TigerGraph

Today's Presenters

Mingxi Wu
VP of Engineering

- 19+ years in data management industry & research
- BS in Computer Science from Fudan University
- MS & Ph.D in Computer Science from University of Florida



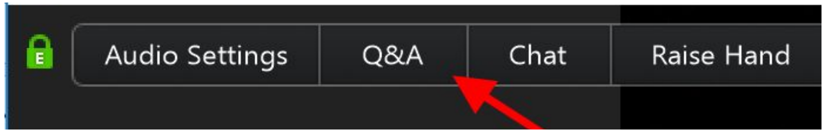
Changran Liu
Solution Architect

- BS in Mechanical Engineering, Tsinghua University
- MS & PhD in Mechanical Engineering, Stanford University
- PhD minor in Philosophy focused on applications of mathematical logic in artificial intelligence

TigerGraph

Some Housekeeping Items

- Although your phone is muted we do want to answer your questions - **submit your questions at any time** using the Q&A tab in the menu



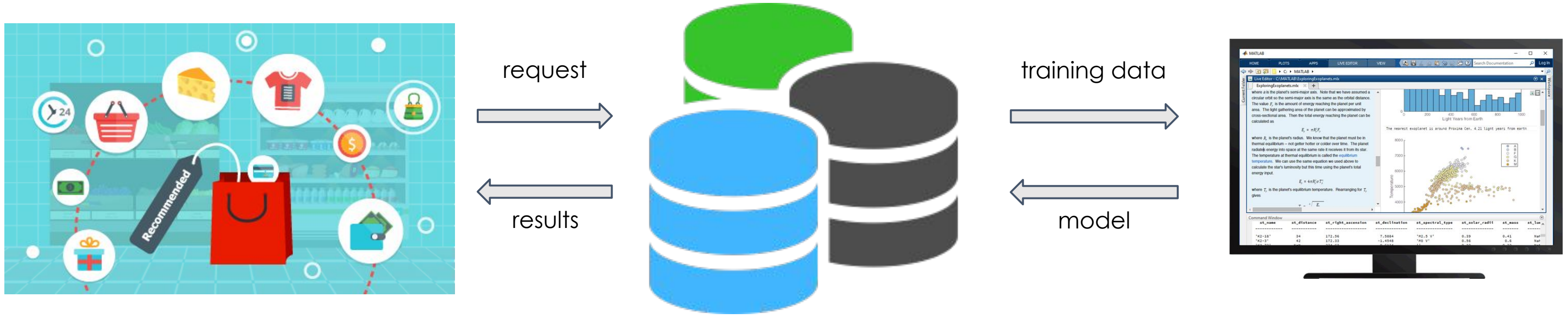
- The webinar is being recorded and will uploaded to our website shortly (https://www.tigergraph.com/webinars/) and the URL will be emailed you

- If you have issues with Zoom please contact the panelists via chat

Outline

- Why Do ML in Graph Database
- Recommendation Systems
- Demo
- Latent factor model (model based)
    - Intuition
    - Implementation

Current Situation



Applications:

- recommendation
- fraud detection
- ...

Database:

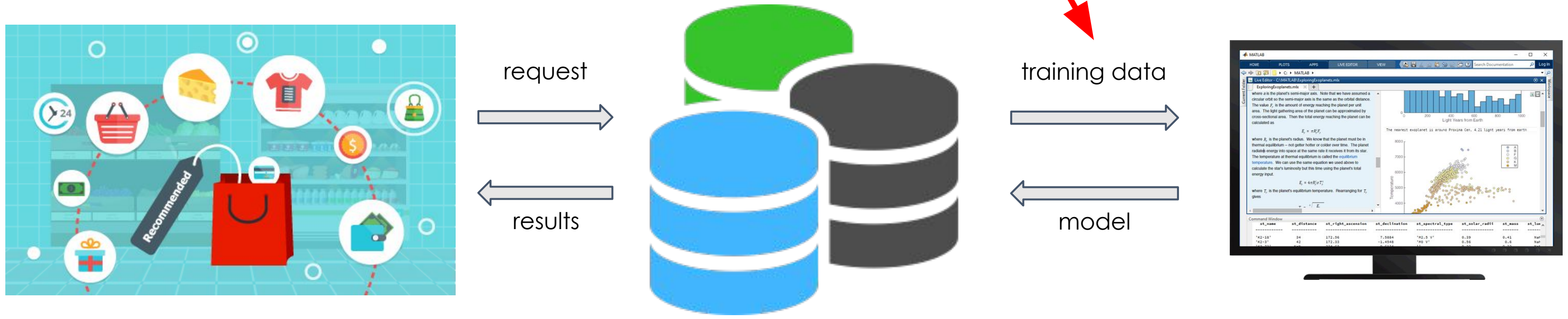
- data storage
- data update
- preprocess data

Machine learning platform

- model training
- model validation

Current Situation

The whole training set needs to be transferred



Applications:

- recommendation
- fraud detection
- ...

Database:

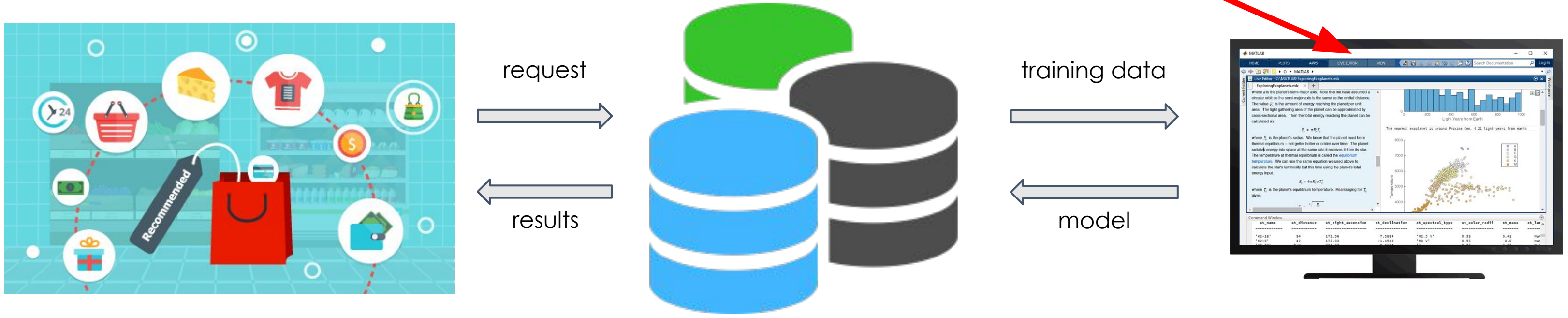
- data storage
- data update
- preprocess data

Machine learning platform

- model training
- model validation

Current Situation

Data is stale when it's used for training



Applications:

- recommendation
- fraud detection
- ...

Database:

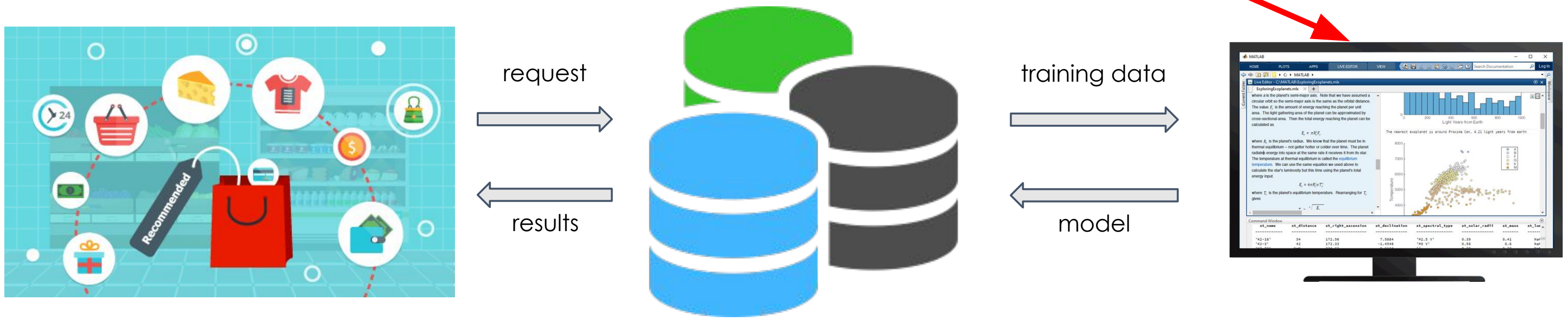
- data storage
- data update
- preprocess data

Machine learning platform

- model training
- model validation

Current Situation

Learning platform is not scaled-out

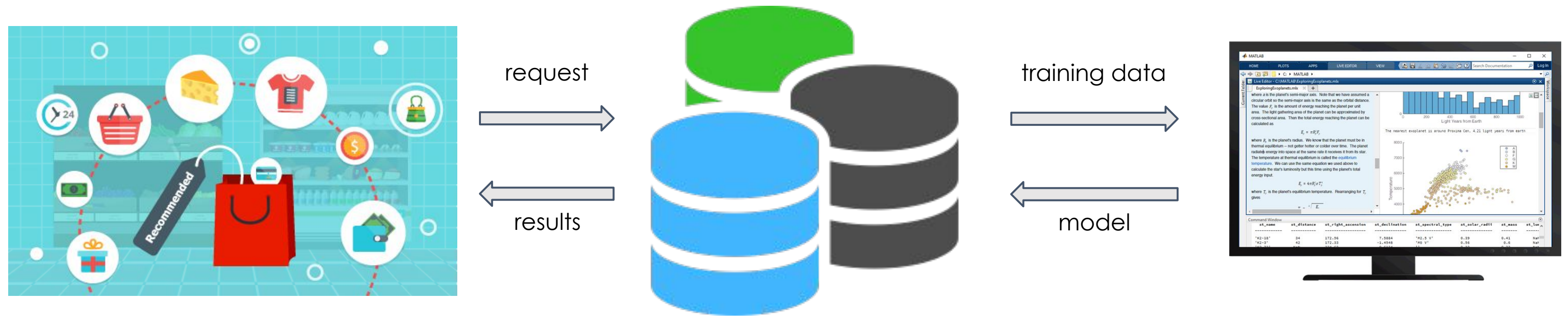


Applications:
- recommendation
- fraud detection
- ...

Database:
- data storage
- data update
- preprocess data

Machine learning platform
- model training
- model validation

The Challenge For In-database ML



Applications:

- recommendation
- fraud detection
- ...

Database:

- data storage
- data update
- preprocess data

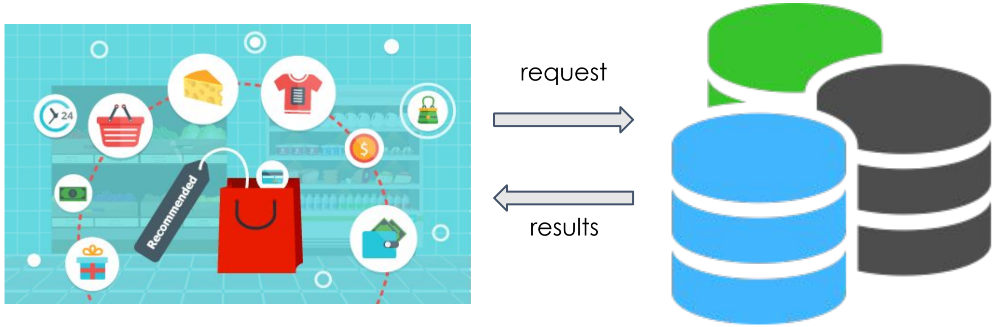
Machine learning platform

- model training
- model validation

- SQL is declarative, not good for iterative algorithms
- Relational model prevents users get some useful features that spanning multiple hops.
- Many databases are not real-time mutable, so data is stale.

TigerGraph

Solution: In Graph Database ML with GSQL



Applications:

- recommendation
- fraud detection
- ...

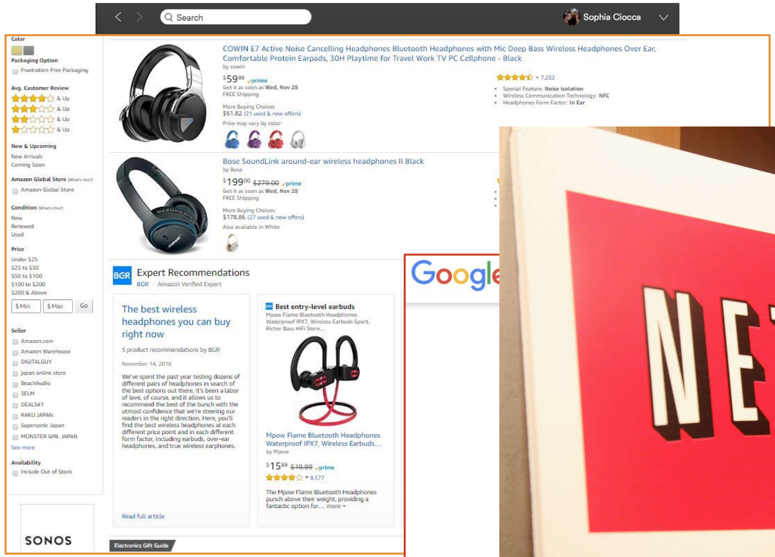
In-situ ML in TigerGraph Database:

- Native graph storage and PG model
- Coded once, auto scale-out & scale-up
- Support real-time update
- GSQL Turing-complete language
  - preprocess data
  - model training: flow-control, accumulator, pattern match
  - model validation

Why Do ML in a Graph Database? (cont.)

- Data management capability. Graph database has unique advantage over other database (such as relational database) in managing explosive data elements due to
  - Natural modeling. Graph model is object oriented modeling with relationship(edge) as the first class citizen.
  - Flexibility. Expand and shrink data model will not break existing query workload
- Compute capability. Declarative query language GSQL assists data scientist write ML algorithm at high level
  - **Flow control**: WHILE, FOREACH, IF-THEN
  - **Accumulator**: provide runtime state variable at vertex and global level
  - **Pattern Match**: declaratively specify what data set to include/exclude.
- In-situ machine learning of the data habitat reduces the overhead of exporting data, and inherently avoids the data stale problem
- Excellent scale capability by TigerGraph MPP architecture: leave the scale-up and scale-out engineering challenge to the graph database engine.

Recommendation Systems



Search

Sophia Ciocca

Color

Packaging Option

Frustration-Free Packaging

Avg. Customer Review

& Up

& Up

& Up

& Up

New & Upcoming

New Arrivals

Coming Soon

Amazon Global Store (What's this?)

Amazon Global Store

Condition (What's this?)

New

Renewed

Used

Price

Under $25

$25 to $50

$50 to $100

$100 to $200

$200 & Above

$ Min $ Max Go

Seller

Amazon.com

Amazon Warehouse

DIGITALGUY

japan online store

BeachAudio

3ELM

DEALS#1

RAKU JAPAN

Supersonic Japan

MONSTER GIRL JAPAN

See more

Availability

Include Out of Stock

SONOS

COWIN E7 Active Noise Cancelling Headphones Bluetooth Headphones with Mic Deep Bass Wireless Headphones Over Ear, Comfortable Protein Earpads, 30H Playtime for Travel Work TV PC Cellphone - Black

by cowin

$59.99 prime

Get it as soon as Wed, Nov 28

FREE Shipping

More Buying Choices

$51.82 (21 used & new offers)

Price may vary by color

7,252

- Special Feature: Noise Isolation
- Wireless Communication Technology: NFC
- Headphones Form Factor: In Ear

Bose SoundLink around-ear wireless headphones II Black

by Bose

$199.00 ~~$279.00~~ prime

Get it as soon as Wed, Nov 28

FREE Shipping

More Buying Choices

$178.86 (27 used & new offers)

Also available in White

BGR Expert Recommendations

BGR · Amazon Verified Expert

The best wireless headphones you can buy right now

5 product recommendations by BGR

November 14, 2018

We've spent the past year testing dozens of different pairs of headphones in search of the best options out there. It's been a labor of love, of course, and it allows us to recommend the best of the bunch with the utmost confidence that we're steering our readers in the right direction. Here, you'll find the best wireless headphones at each different price point and in each different form factor, including earbuds, over-ear headphones, and true wireless earphones.

Read full article

Best entry-level earbuds

Mpow Flame Bluetooth Headphones Waterproof IPX7, Wireless Earbuds Sport, Richer Bass HiFi Stere...

Mpow Flame Bluetooth Headphones Waterproof IPX7, Wireless Earbuds...

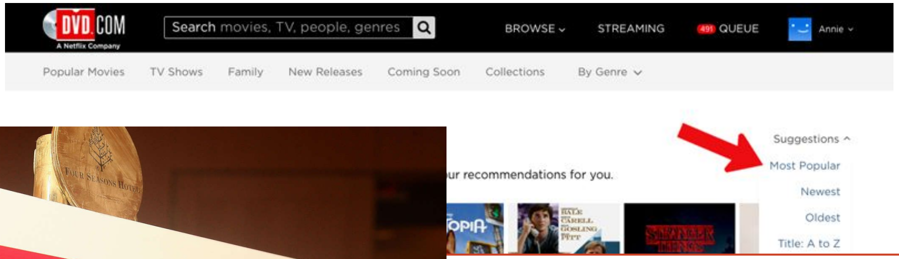
by Mpow

$15.89 ~~$19.99~~ prime

9,577

The Mpow Flame Bluetooth Headphones punch above their weight, providing a fantastic option for... more

Electronics Gift Guide



DVD.COM A Netflix Company

Search movies, TV, people, genres

BROWSE STREAMING 491 QUEUE Annie

Popular Movies TV Shows Family New Releases Coming Soon Collections By Genre

ur recommendations for you.

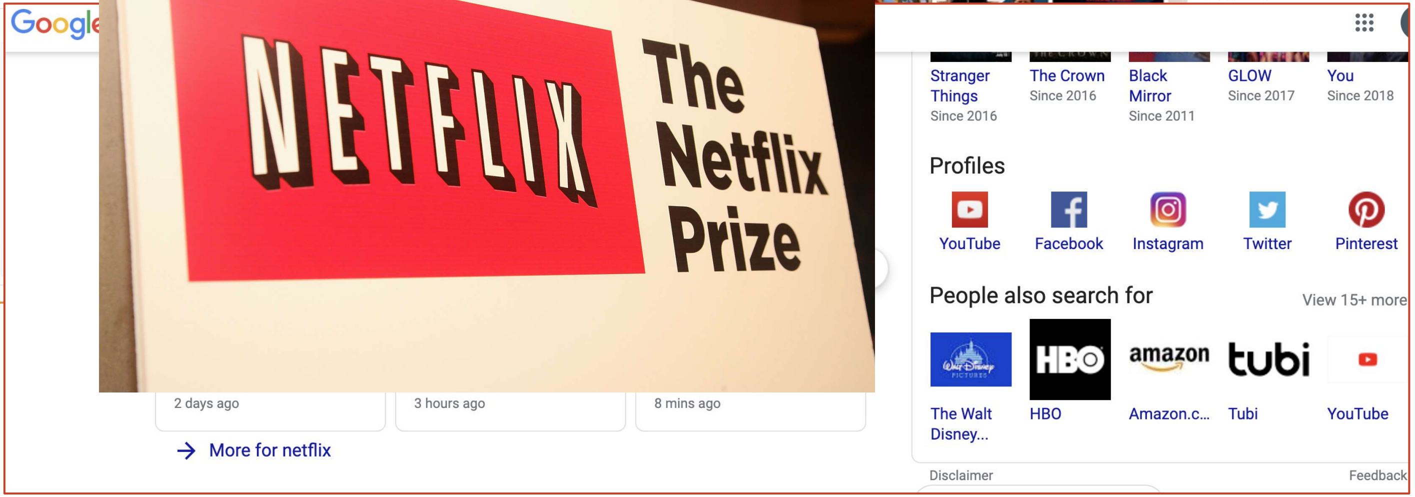
Suggestions

Most Popular

Newest

Oldest

Title: A to Z



Google

2 days ago

3 hours ago

8 mins ago

More for netflix

Stranger Things	The Crown	Black Mirror	GLOW	You
Since 2016	Since 2016	Since 2011	Since 2017	Since 2018

Profiles

YouTube Facebook Instagram Twitter Pinterest

People also search for

View 15+ more

The Walt Disney... HBO Amazon.c... Tubi YouTube

Disclaimer

Feedback

Movie Recommendation

movie features

MARVEL'S THE AVENGERS

PG13, 2 hr.22 min.
Action & Adventure , Science Fiction & Fantasy
Directed By: Joss Whedon
In Theaters: May 4, 2012 Wide
On DVD: Sep 25, 2012
Walt Disney Pictures

The Avengers: Trailer 1
1 minute 55 seconds
Added: Apr 24, 2018

The Avengers: Trailer 2
2 minutes 22 seconds
Added: Apr 24, 2018

VIEW ALL VIDEOS (2)

ottentomatoes.com/m/marvels_the_avengers/

MARVEL'S THE AVENGERS REVIEWS

All Critics | Top Critics | **All Audience**

users

ratings

NEXT →

Danny D

★☆☆☆☆ 5d ago

How many movies did it take to come up with this mundane plot ?

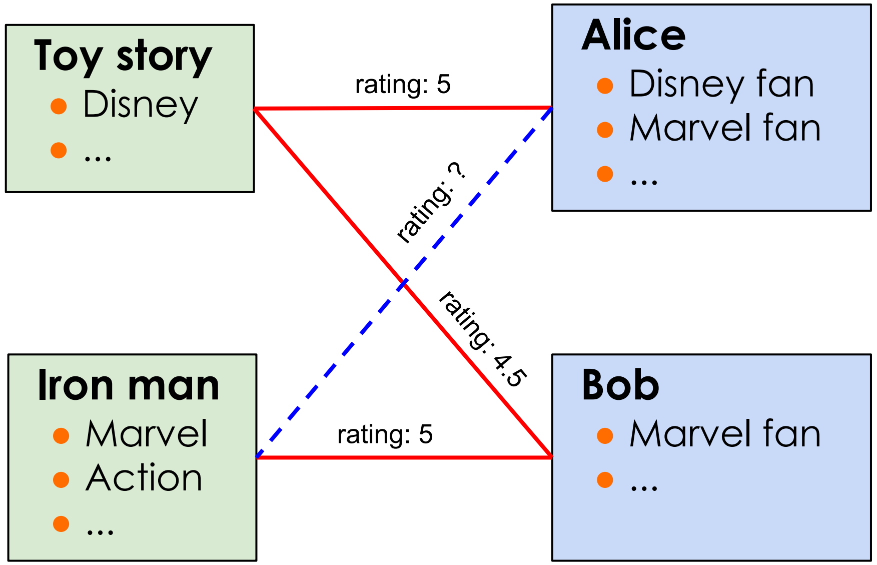
Benjamin C

Martyn K

Goals:

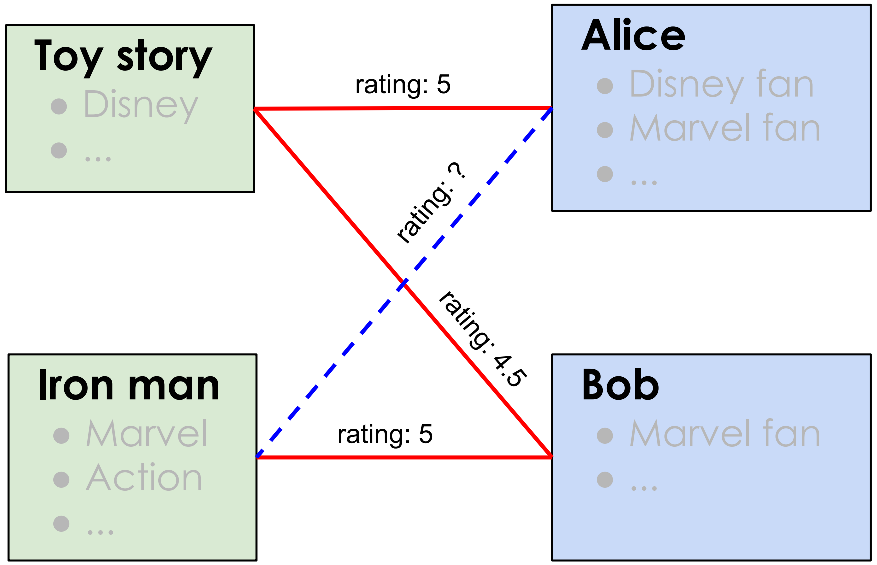
- Predict users' ratings for movies they haven't seen, based on previous ratings
- Recommend movies to users based on rating prediction

User-Rate-Movie Graph



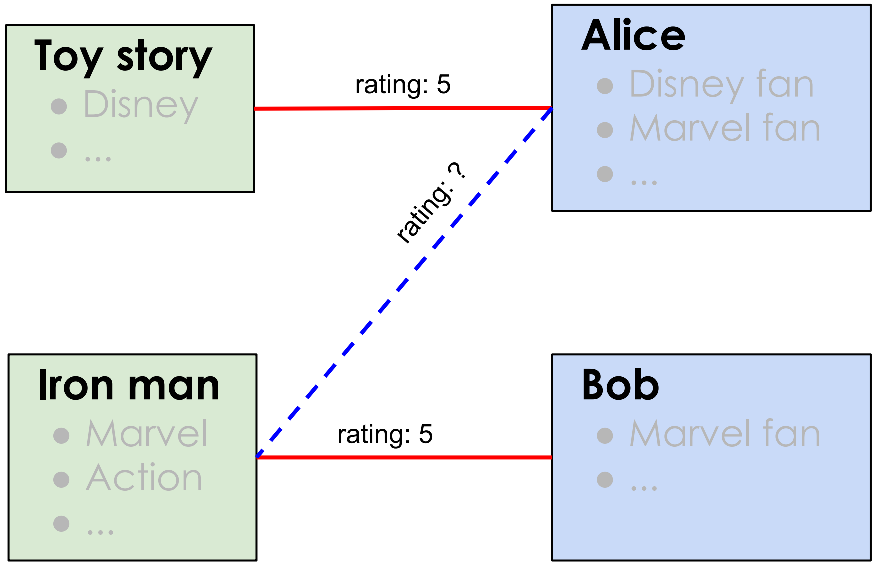
- Content based method

User-Rate-Movie Graph



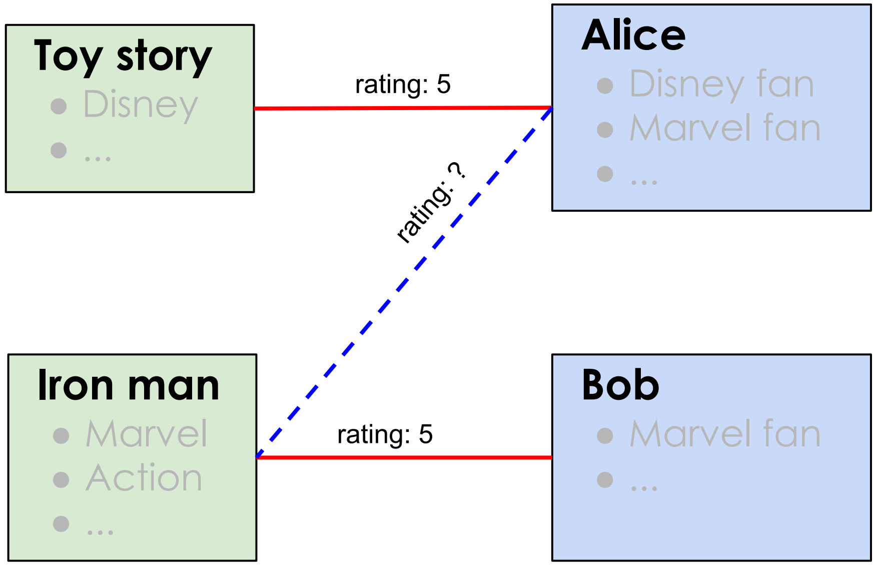
- Content based method
- K-nearest neighbors

User-Rate-Movie Graph



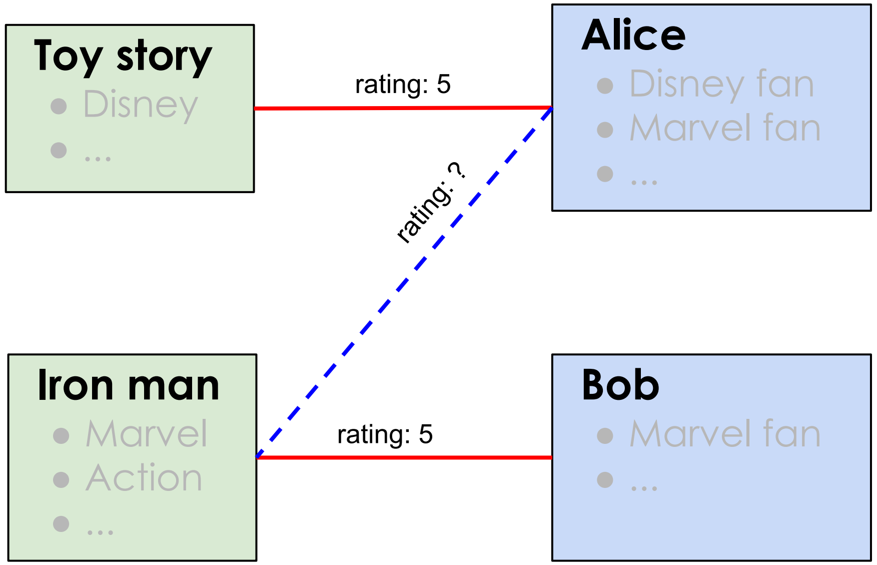
- Content based method
- K-nearest neighbors
- Latent factor (model-based)

User-Rate-Movie Graph



- Content based method
- K-nearest neighbors
- Latent factor (model-based)
- Hybrid method
- ...

User-Rate-Movie Graph



- Content based method
- K-nearest neighbors
- **Latent factor (model-based)**
- Hybrid method
- ...

Outline

- Why Do ML in Graph Database
- Recommendation Systems
- **Demo**
- Latent factor model (model based)
  - Intuition
  - Implementation

TigerGraph

Training

- splitData: tag training and validation data, and persist the tag on the training data
- Initialization: initialize the latent factor vectors
- traing_validation: solve the latent factor vectors by gradient descent using tagged training graph data. The trained latent factors are persist to user and movies vertices as their attributes.
- recommend: output top 10 movies for a given users based on the recommendation model trained in previous query.

Demo

MovieLens Data

- Dataset of 100k ratings and 40k tags that 1k users gave to 17k movies
- Each rating is a quadruplet of the form `<user, movie, rating, date>`
- Each movie is tagged with multiple different terms
- The user and movie fields are integer IDs, while grades are from 0 to 5 stars
- https://grouplens.org/datasets/movielens/

Root Mean Square Error (RMSE)

$$\sqrt{\frac{1}{M}\sum_{i,j:\,r(i,j)=1}^{M}\left(\hat{y}^{(i,j)} - y^{(i,j)}\right)^2}$$

Results

TF-IDF method
(content based)
RMSE: **0.91239**

+

Latent factor model
(model based)
RMSE: **0.96869**

=

hybrid model
RMSE: **0.90368**

Root Mean Square Error (RMSE) = $\sqrt{\frac{1}{M}\sum_{i,j:\,r(i,j)=1}^{M}(\hat{y}^{(i,j)} - y^{(i,j)})^2}$

Outline

- Why Do ML in Graph Database
- Recommendation Systems
- Demo
- **Latent factor model (model based)**
  - Intuition
  - Implementation

TigerGraph

Movie Rating Prediction (Latent factors model)

$\theta^{(1)} = [5, 0]$ $\theta^{(2)} = [5, 0]$ $\theta^{(3)} = [0, 5]$ $\theta^{(4)} = [0, 5]$

	Movie	Alice		Bob	Carol	Dave
$x^{(1)} = [0.9, 0]$	Love at last	5	4.5	5	0	0
$x^{(2)} = [1, 0.1]$	Romance forever	5	5	?	?	0
$x^{(3)} = [0.9, 0]$	Cute puppies of love	?	4.5	4	0	?
$x^{(4)} = [0.1, 1]$	Toy story	?	0.5	?	?	5
$x^{(5)} = [0.1, 1]$	Sword vs. karate	0	0.5	0	5	?
$x^{(6)} = [0, 0.9]$	Nonstop car chases	0	0	0	5	4

- Each movie has a latent factor vector: $\theta^{(j)}$
- Each user has a latent factor vector: $x^{(i)}$
- Predict the user j's rating to movie i by: $(\theta^{(j)})^T x^{(i)}$

Movie Rating Prediction (Latent factors model)

$\theta^{(1)} = [5, 0]$ $\theta^{(2)} = [5, 0]$ $\theta^{(3)} = [0, 5]$ $\theta^{(4)} = [0, 5]$

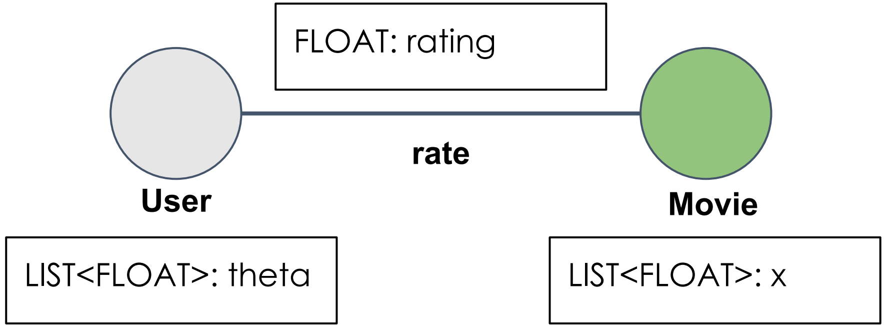
romance

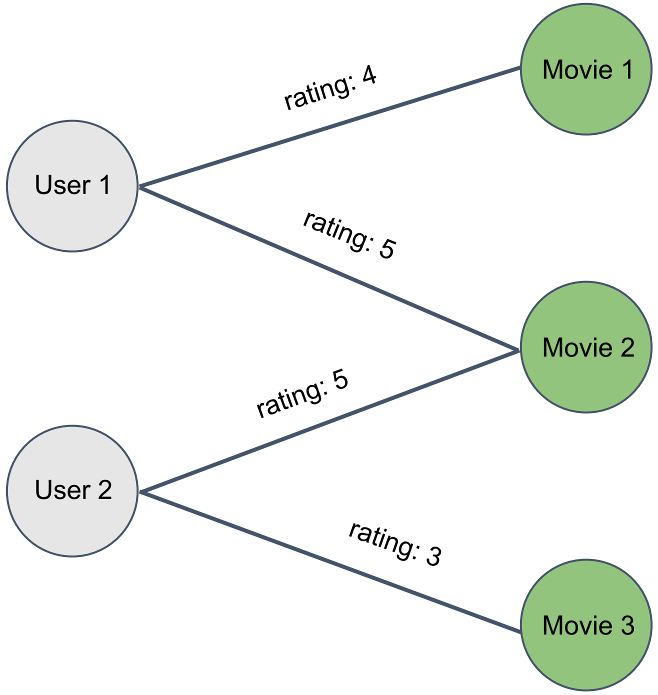
action

	Movie	Alice		Bob	Carol	Dave
$x^{(1)} = [0.9, 0]$	Love at last	5	4.5	5	0	0
$x^{(2)} = [1, 0.1]$	Romance forever	5	5	?	?	0
$x^{(3)} = [0.9, 0]$	Cute puppies of love	?	4.5	4	0	?
$x^{(4)} = [0.1, 1]$	Toy story	?	0.5	?	?	5
$x^{(5)} = [0.1, 1]$	Sword vs. karate	0	0.5	0	5	?
$x^{(6)} = [0, 0.9]$	Nonstop car chases	0	0	0	5	4

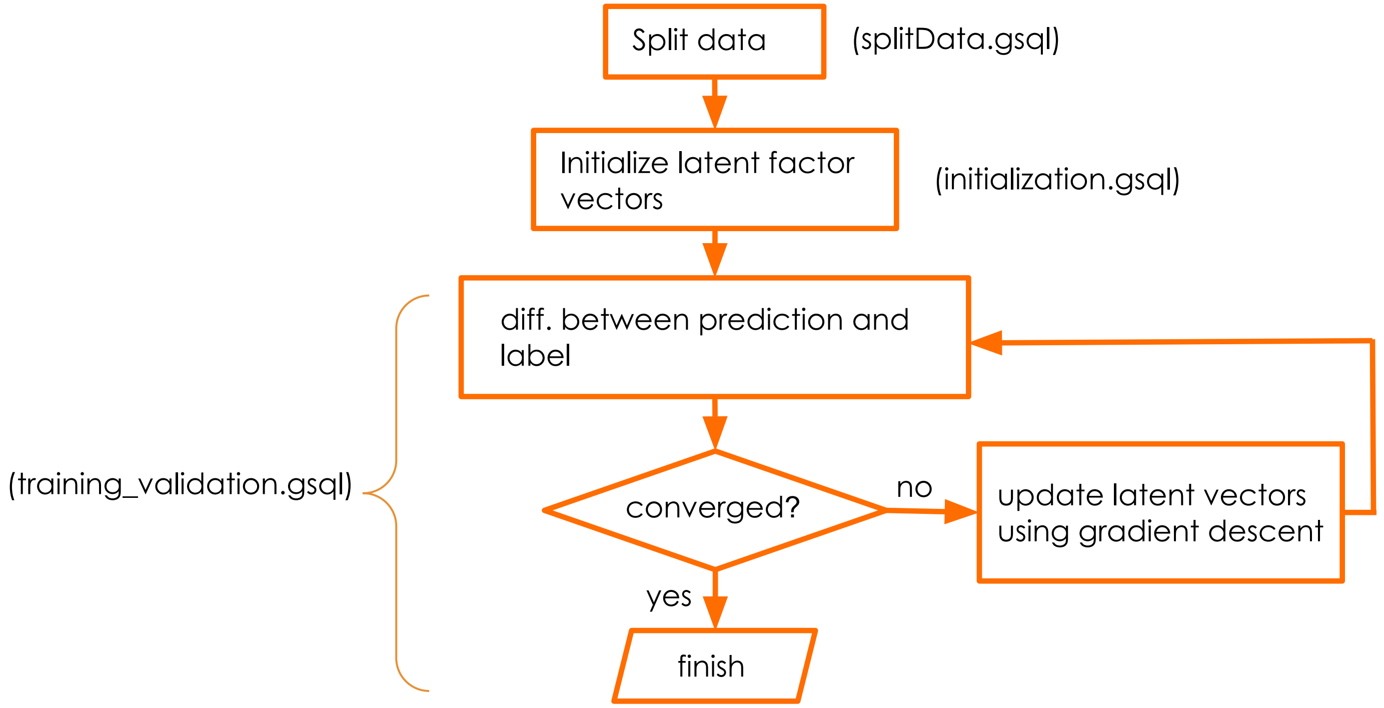
- Each movie has a latent factor vector: $\boldsymbol{\theta}^{(\mathbf{j})}$
- Each user has a latent factor vector: $\boldsymbol{x}^{(\mathbf{i})}$
- Predict the user j's rating to movie i by: $(\boldsymbol{\theta}^{(\mathbf{j})})^{\mathbf{T}}\boldsymbol{x}^{(\mathbf{i})}$

Schema and Graph



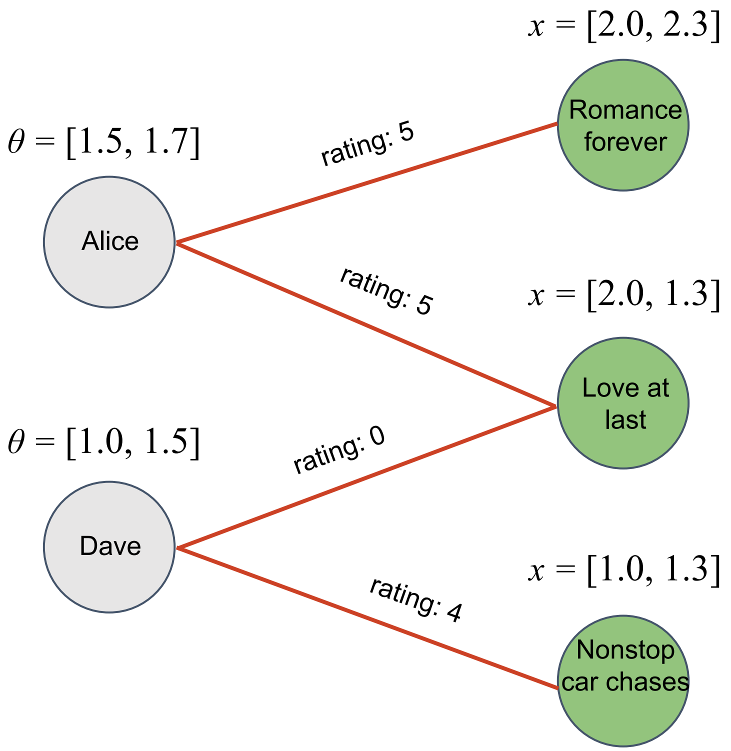


Training


Split data
(splitData.gsql)
Initialize latent factor vectors
(initialization.gsql)
diff. between prediction and label
(training_validation.gsql)
converged?
no
update latent vectors using gradient descent
yes
finish

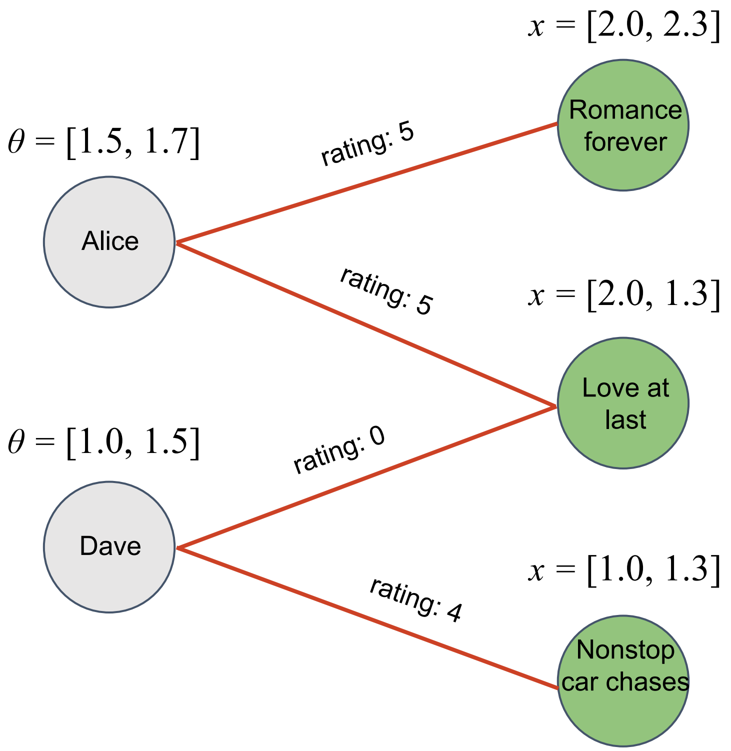
GSQL Training Block

```
USERs = SELECT s FROM USERs:s -(rate:e)-> MOVIE:t
        ACCUM
          DOUBLE prediction = dotProduct(s.@theta,t.@x),
          DOUBLE delta = prediction-e.rating,
          s.@Gradient += product(t.@x,delta),
          t.@Gradient += product(s.@theta,delta)
        POST-ACCUM
          s.@theta += product(s.@Gradient,-alpha),
          t.@x += product(t.@Gradient,-alpha);
```



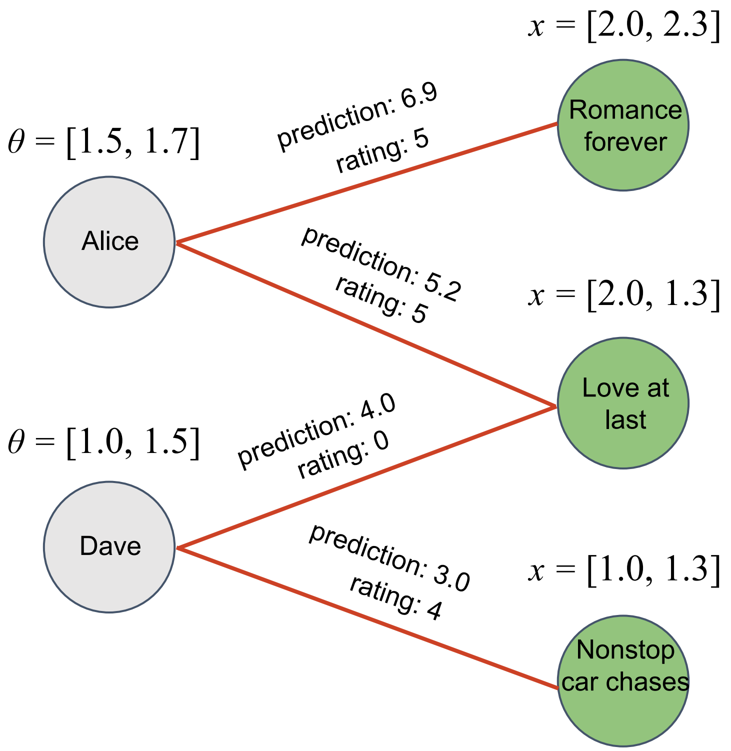
GSQL Training Block

```
USERs = SELECT s FROM USERs:s -(rate:e)-> MOVIE:t
          ACCUM
            DOUBLE prediction = dotProduct(s.@theta,t.@x),
            DOUBLE delta = prediction-e.rating,
            s.@Gradient += product(t.@x,delta),
            t.@Gradient += product(s.@theta,delta)
          POST-ACCUM
            s.@theta += product(s.@Gradient,-alpha),
            t.@x += product(t.@Gradient,-alpha);
```



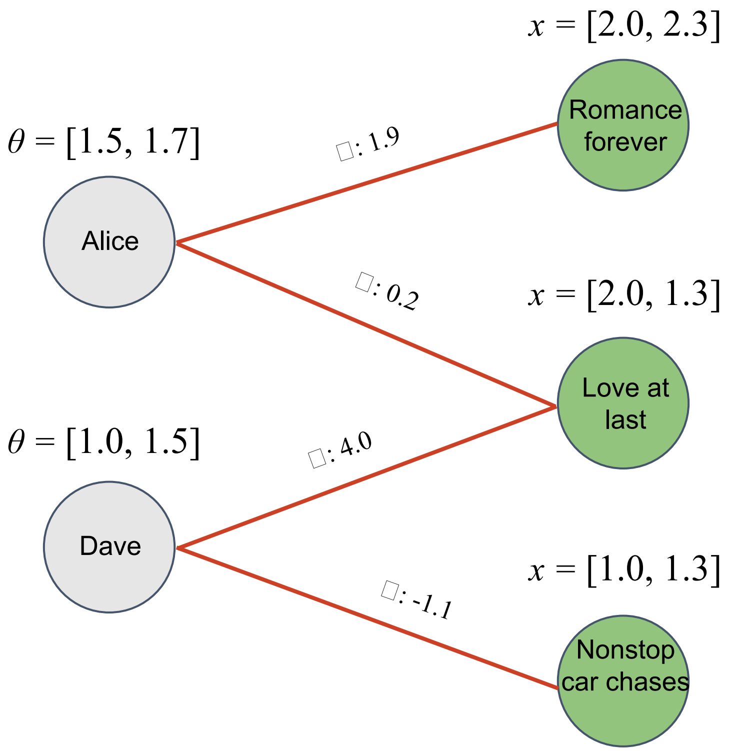
GSQL Training Block

```
USERs = SELECT s FROM USERs:s -(rate:e)-> MOVIE:t
      ACCUM
        DOUBLE prediction = dotProduct(s.@theta,t.@x),
        DOUBLE delta = prediction-e.rating,
        s.@Gradient += product(t.@x,delta),
        t.@Gradient += product(s.@theta,delta)
      POST-ACCUM
        s.@theta += product(s.@Gradient,-alpha),
        t.@x += product(t.@Gradient,-alpha);
```



GSQL Training Block

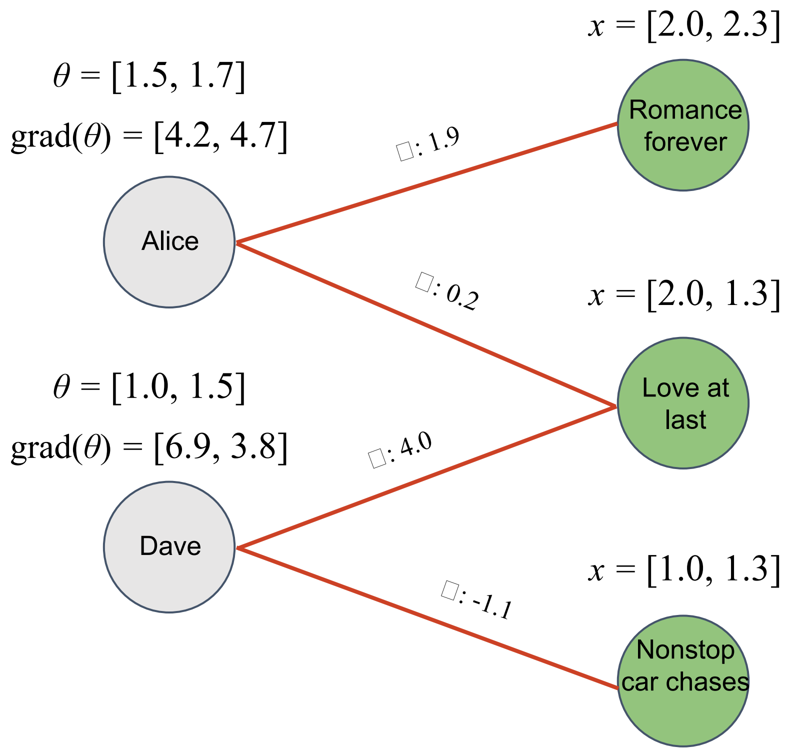
```
USERs = SELECT s FROM USERs:s -(rate:e)-> MOVIE:t
        ACCUM
          DOUBLE prediction = dotProduct(s.@theta,t.@x),
          DOUBLE delta = prediction-e.rating,
          s.@Gradient += product(t.@x,delta),
          t.@Gradient += product(s.@theta,delta)
        POST-ACCUM
          s.@theta += product(s.@Gradient,-alpha),
          t.@x += product(t.@Gradient,-alpha);
```



TigerGraph

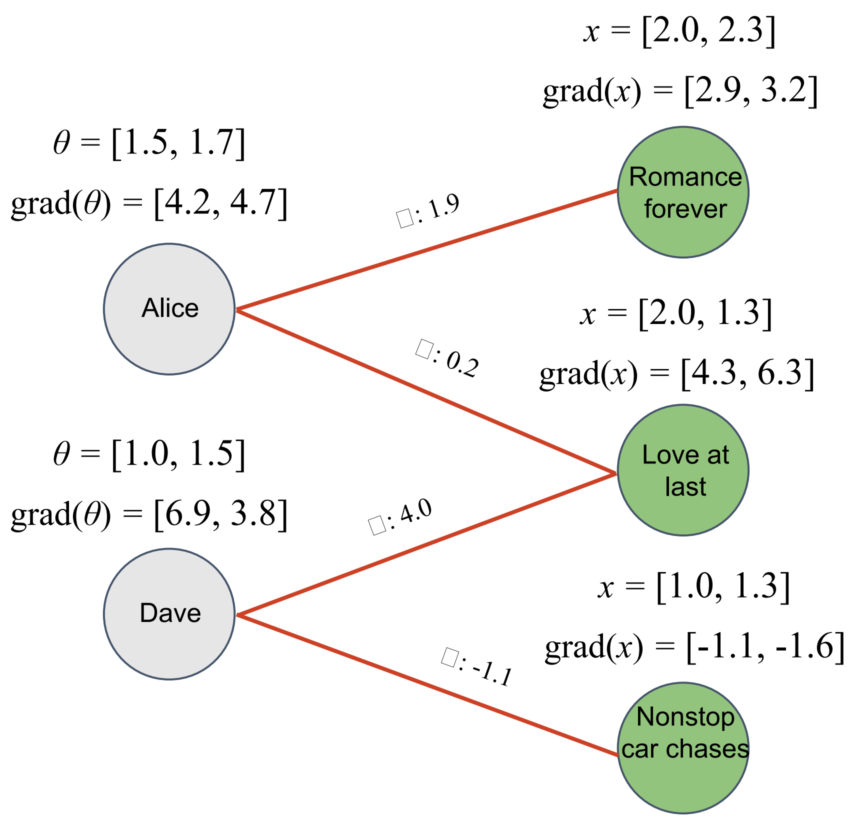
GSQL Training Block

```
USERs = SELECT s FROM USERs:s -(rate:e)-> MOVIE:t
     ACCUM
       DOUBLE prediction = dotProduct(s.@theta,t.@x),
       DOUBLE delta = prediction-e.rating,
       s.@Gradient += product(t.@x,delta),
       t.@Gradient += product(s.@theta,delta)
     POST-ACCUM
       s.@theta += product(s.@Gradient,-alpha),
       t.@x += product(t.@Gradient,-alpha);
```



GSQL Training Block

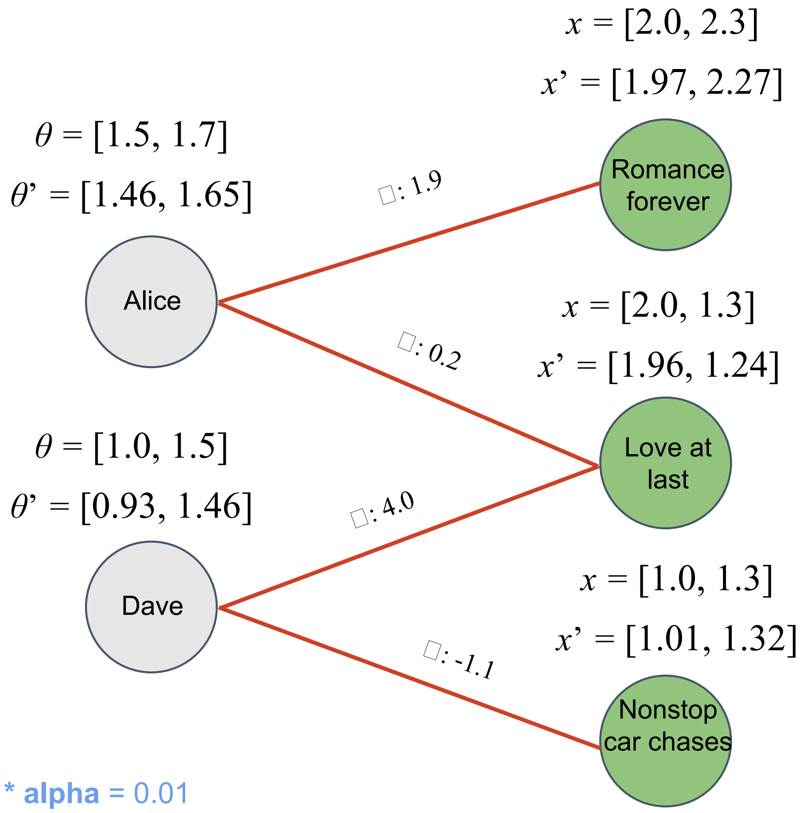
```
USERs = SELECT s FROM USERs:s -(rate:e)-> MOVIE:t
        ACCUM
          DOUBLE prediction = dotProduct(s.@theta,t.@x),
          DOUBLE delta = prediction-e.rating,
          s.@Gradient += product(t.@x,delta),
          t.@Gradient += product(s.@theta,delta)
        POST-ACCUM
          s.@theta += product(s.@Gradient,-alpha),
          t.@x += product(t.@Gradient,-alpha);
```



TigerGraph

GSQL Training Block

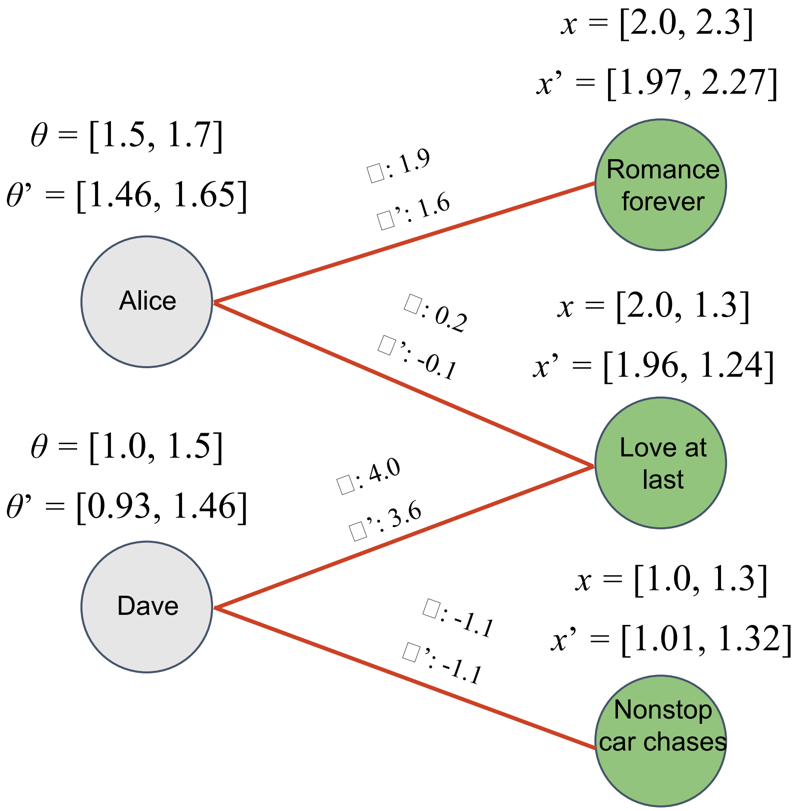
```
USERs = SELECT s FROM USERs:s -(rate:e)-> MOVIE:t
     ACCUM
       DOUBLE prediction = dotProduct(s.@theta,t.@x),
       DOUBLE delta = prediction-e.rating,
       s.@Gradient += product(t.@x,delta),
       t.@Gradient += product(s.@theta,delta)
     POST-ACCUM
       s.@theta += product(s.@Gradient,-alpha),
       t.@x += product(t.@Gradient,-alpha);
```



* **alpha** = 0.01

GSQL Training Block

```
USERs = SELECT s FROM USERs:s -(rate:e)-> MOVIE:t
          ACCUM
            DOUBLE prediction = dotProduct(s.@theta,t.@x),
            DOUBLE delta = prediction-e.rating,
            s.@Gradient += product(t.@x,delta),
            t.@Gradient += product(s.@theta,delta)
          POST-ACCUM
            s.@theta += product(s.@Gradient,-alpha),
            t.@x += product(t.@Gradient,-alpha);
```



Outline

- Why Do ML in Graph Database
- Recommendation Systems
- In-database Learning
- Latent factor model (model based)
- **TF-IDF method (content based)**
- Hybrid method

TF-IDF (Term Frequency-Inverse Document Frequency)

Movie features

term	Love forever	Saw
action	1	1
horror	0	1
romance	1	0

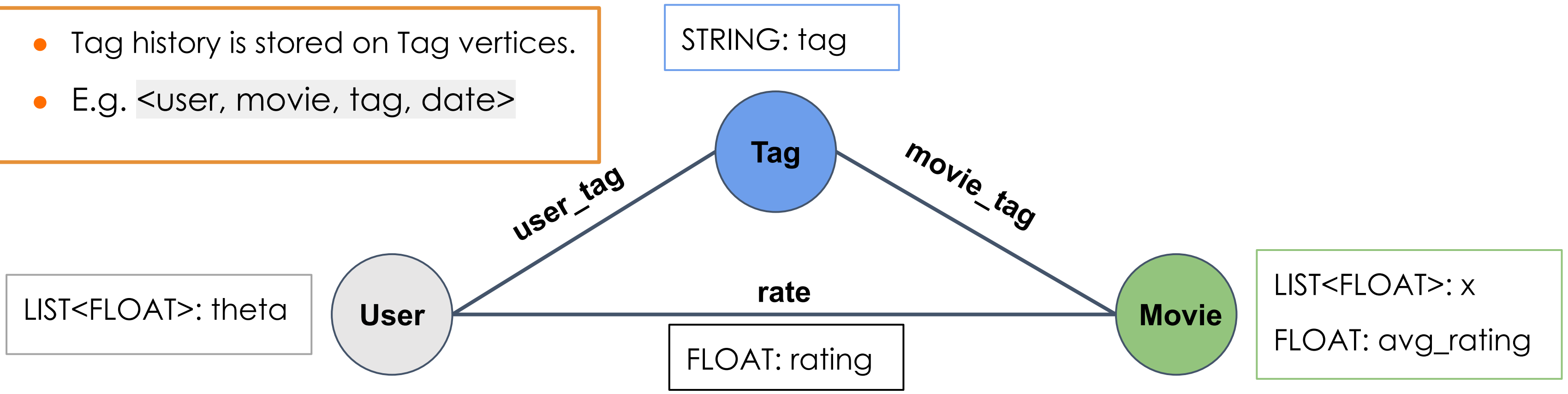
User profiles

term	Alice	Jack
action	1	0
horror	1	-1
romance	0	1

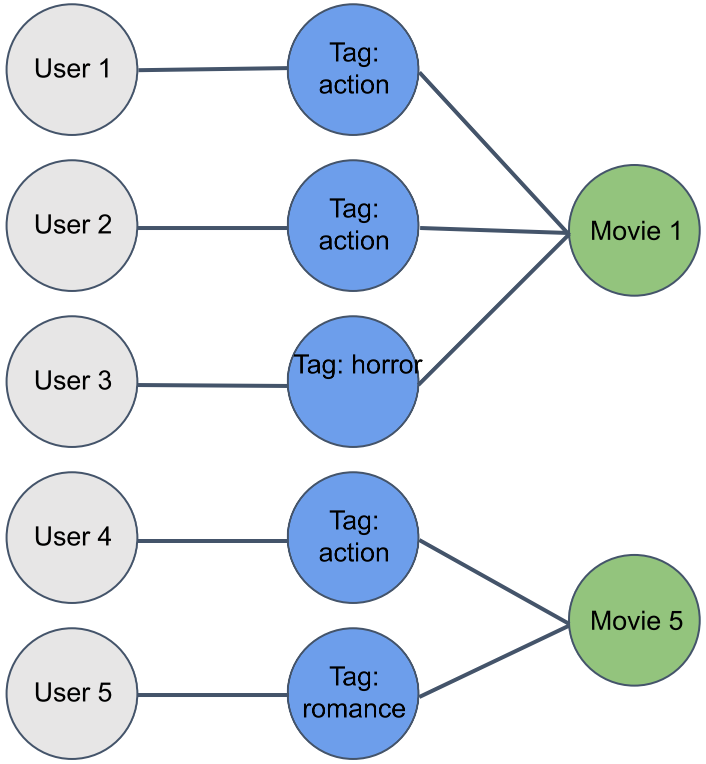
- TF-IDF of each tag for each movie are computed from tag history.
- Movie features are determined from the TF-IDF of its tags.
- User profiles are constructed from the features of the movies rated by the user.

Schema

- Tag history is stored on Tag vertices.
- E.g. <user, movie, tag, date>



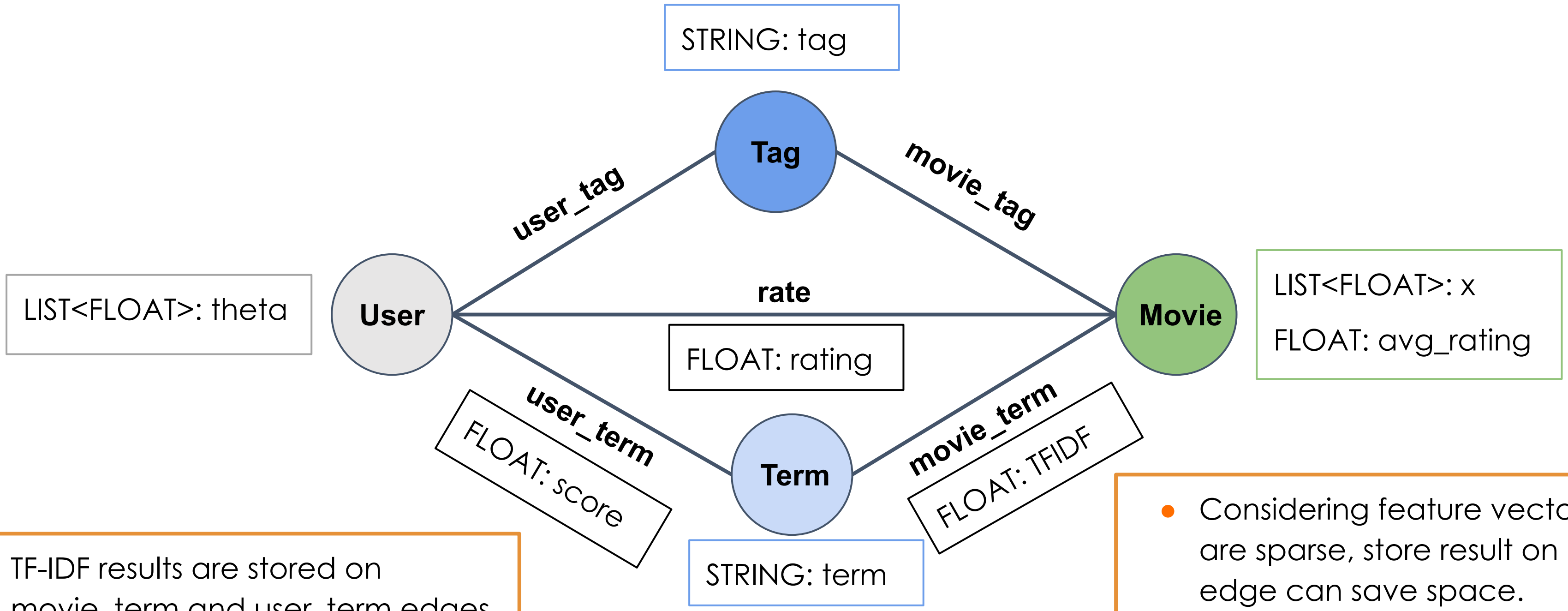
TF-IDF (Term Frequency-Inverse Document Frequency)



term	term frequency	
	Movie 1	Movie 2
action	2	1
horror	1	0
romance	0	1

term	**inverse document frequency = log(N_m/$N_{m,t}$)** N_m: number of movies $N_{m,t}$: number of movies tagged with term)
action	log(2/2) = 0
horror	log(2/1) = 0.3
romance	log(2/1) = 0.3

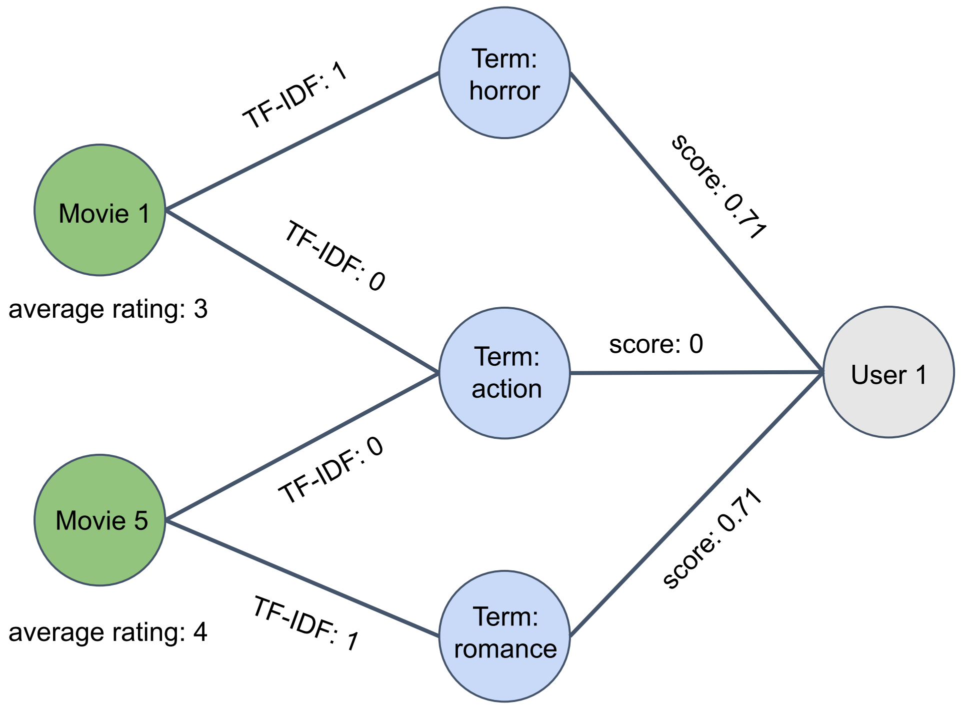
Schema



- TF-IDF results are stored on movie_term and user_term edges.

- Considering feature vectors are sparse, store result on edge can save space.
- Does not need an index table.

TF-IDF



Movie 1
average rating: 3
Movie 5
average rating: 4
TF-IDF: 1
TF-IDF: 0
TF-IDF: 0
TF-IDF: 1
Term: horror
Term: action
Term: romance
score: 0.71
score: 0
score: 0.71
User 1

Outline

- Why Do ML in Graph Database
- Recommendation Systems
- In-database Learning
- Latent factor model (model based)
- TF-IDF method (content based)
- **Hybrid method**

TigerGraph

Hybrid Model

- User j has a predicted rating to movie i based on content: $\mathbf{CB\_prediction}_{i,j}$
- Compute latent factors for
  - each user: $\theta^{(j)}$
  - each movie: $x^{(i)}$
- so that the user j's rating to movie i can be predicted by: $(\theta^{(j)})^T x^{(i)} + \mathbf{CB\_prediction}_{i,j}$

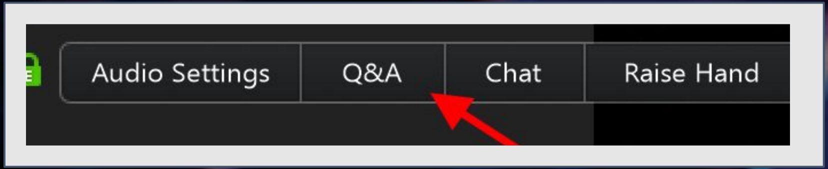
```
USERs = SELECT s FROM USERs:s -(rate:e)-> MOVIE:t
        ACCUM
          DOUBLE prediction = dotProduct(s.@theta,t.@x),
          DOUBLE delta = prediction+e.CB_prediction-e.rating,
          s.@Gradient += product(t.@x,delta),
          t.@Gradient += product(s.@theta,delta)
        POST-ACCUM
          s.@theta += product(s.@Gradient,-alpha),
          t.@x += product(t.@Gradient,-alpha);
```

Summary

- User-rate-item relation can be represented as a graph
- The Latent factor model can be trained in TigerGraph database
- The hybrid recommendation model can be conveniently implemented using TigerGraph
- The solution for recommendation system can easily be adapted for link prediction or entity resolution problems.

Q&A

Please submit your questions via the Q&A tab in Zoom



More Questions?

Join our Developer Forum

https://groups.google.com/a/opengsql.org/forum/#!forum/gsql-users

Sign up for our Developer Office Hours (every Thursday at 11 AM PST)

https://info.tigergraph.com/officehours

Additional Resources

Start Free at TigerGraph Cloud Today

https://www.tigergraph.com/cloud/

Test Drive Online Demo

https://www.tigergraph.com/demo

Download the Developer Edition

https://www.tigergraph.com/download/

Guru Scripts

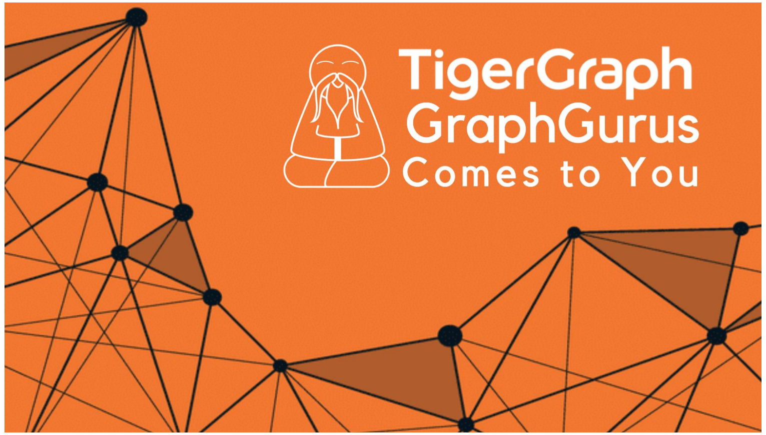
https://github.com/tigergraph/ecosys/tree/master/guru_scripts

Upcoming Graph Guru Events

Virtual Healthcare Roundtable
Transforming Healthcare with Graph Database and Analytics

Virtual Healthcare Roundtable: Transforming Healthcare with Graph Database and Analytics

https://info.tigergraph.com/healthcare-roundtable



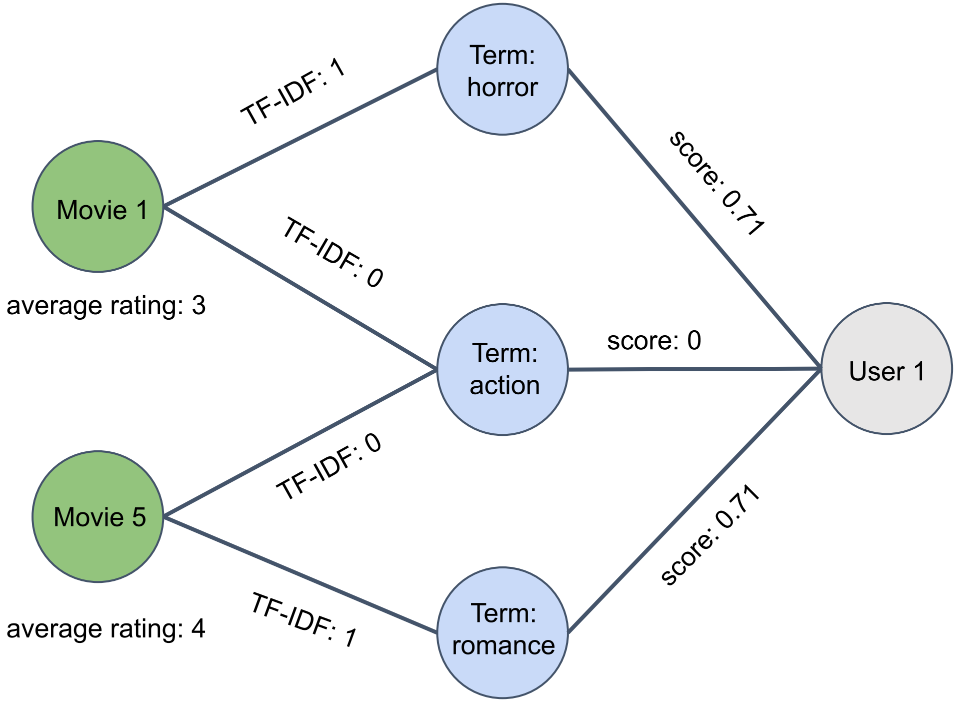
Coming to **Seattle**, **San Francisco**, **Atlanta** and more. View the full list of events, or request your own, here:

https://www.tigergraph.com/graphguruscomestoyou/

Thank You

Extra Slides

Recommendation Systems



- KNN & cosine similarity
- TF-IDF & cosine similarity
- Bayesian classifier
- Latent vectors
- Hybrid method
- ...

Recommendation Systems

Content Filtering

- Based on user/item attributes
- Difficult to interpret attributes

Memory Based

- TF-IDF & cosine similarity

Model Based

- Neural networks
- Bayesian classifiers

Collaborative Filtering

- Based on user behaviors
- Sparse data
- Cold start

Memory Based

- KNN & cosine similarity
- does not work well for sparse data in predicting score.

Model Based

- Latent factor model
- Training model

- Can we have a hybrid model?

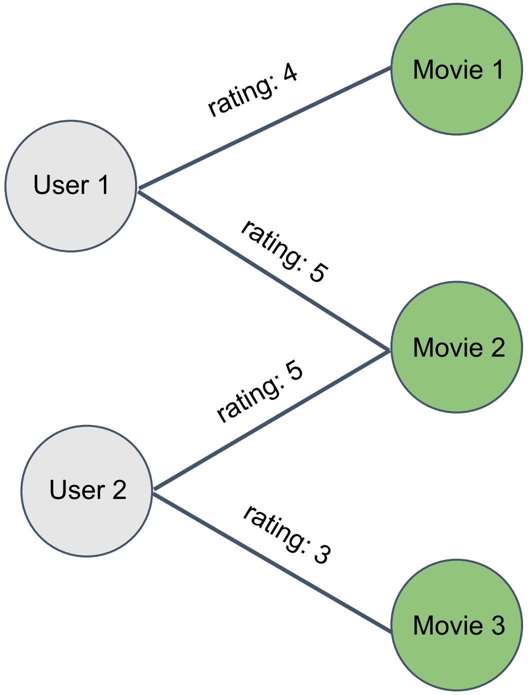
	Content Filtering • Based on user/item attributes • Difficult to interpret attributes	**Collaborative Filtering** • Based on user behaviors • Sparse data • Cold start
Memory Based • Need to query data history to make prediction • does not work well for sparse data in predicting score.	• TF-IDF & cosine similarity	• KNN & cosine similarity
Model Based • Prediction is based on trained model • Training model	• Neural networks • Bayesian classifiers	• Latent factor model

Cost Function

$$J\left(x^{(1)}, \dots, x^{(n_m)}, \theta^{(1)}, \dots, \theta^{(n_u)}\right) = \underbrace{\frac{1}{2} \sum_{(i,j): r(i,j)=1}^{M} \left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)}\right)^2}_{\text{RMSE}} + \underbrace{\frac{\lambda}{2} \sum_{j=1}^{n_u} \sum_{k=1}^{n} \left(\theta_k^{(j)}\right)^2 + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^{n} \left(x_k^{(i)}\right)^2}_{\text{regularization}}$$

$$\frac{\partial J}{\partial \theta_k^{(j)}} = \sum_{i: r(i,j)=1}^{M} \left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)}\right) x_k^{(i)} + \lambda \theta_k^{(j)}$$

$$\frac{\partial J}{\partial x_k^{(i)}} = \sum_{j: r(i,j)=1}^{M} \left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)}\right) \theta_k^{(j)} + \lambda x_k^{(i)}$$



Cost Function

$$J\left(x^{(1)},\ldots,x^{(n_m)},\theta^{(1)},\ldots,\theta^{(n_u)}\right)=\underbrace{\frac{1}{2}\sum_{(i,j):r(i,j)=1}^{M}\left(\sum_{k=1}^{n}\theta_k^{(j)}x_k^{(i)}-y^{(i,j)}\right)^2}_{\text{RMSE}}+\underbrace{\frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}\left(\theta_k^{(j)}\right)^2+\frac{\lambda}{2}\sum_{i=1}^{n_m}\sum_{k=1}^{n}\left(x_k^{(i)}\right)^2}_{\text{regularization}}$$

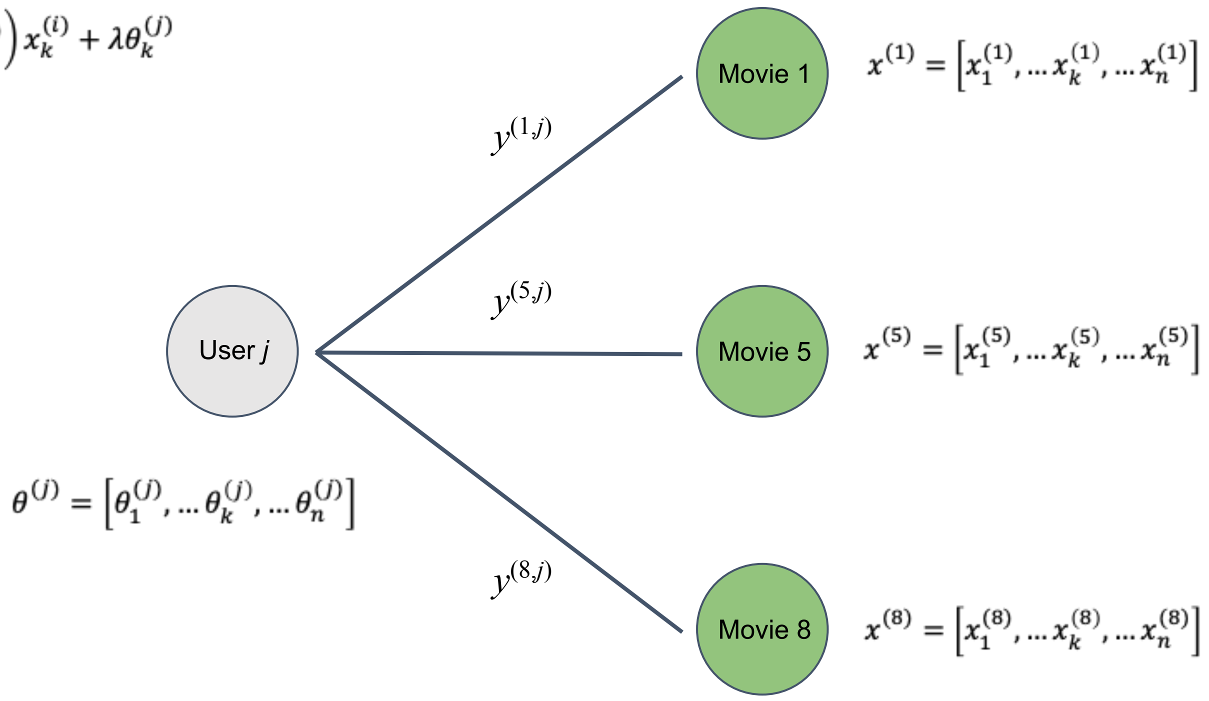
$$\frac{\partial J}{\partial \theta_k^{(j)}}=\sum_{i:r(i,j)=1}^{M}\left(\left(\theta^{(j)}\right)^T x^{(i)}-y^{(i,j)}\right)x_k^{(i)}+\lambda\theta_k^{(j)}$$

$$\frac{\partial J}{\partial x_k^{(i)}}=\sum_{j:r(i,j)=1}^{M}\left(\left(\theta^{(j)}\right)^T x^{(i)}-y^{(i,j)}\right)\theta_k^{(j)}+\lambda x_k^{(i)}$$

Gradient Descent

$$\frac{\partial J}{\partial \theta_k^{(j)}} = \sum_{i:r(i,j)=1}^{M} \left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)} \right) x_k^{(i)} + \lambda \theta_k^{(j)}$$

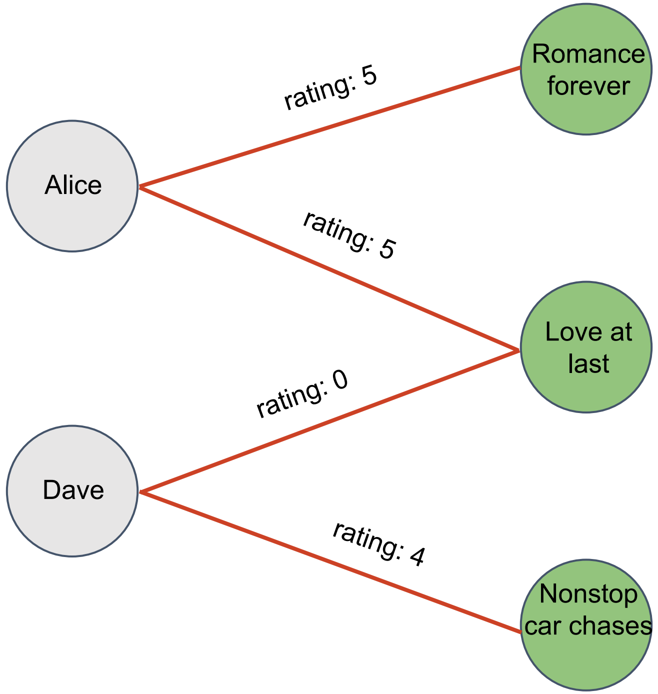
$$\theta_k^{(j)} = \theta_k^{(j)} - \alpha \frac{\partial J}{\partial \theta_k^{(j)}}$$



User *j*
Movie 1
Movie 5
Movie 8
$y^{(1,j)}$
$y^{(5,j)}$
$y^{(8,j)}$
$x^{(1)} = \left[x_1^{(1)}, \dots x_k^{(1)}, \dots x_n^{(1)}\right]$
$x^{(5)} = \left[x_1^{(5)}, \dots x_k^{(5)}, \dots x_n^{(5)}\right]$
$x^{(8)} = \left[x_1^{(8)}, \dots x_k^{(8)}, \dots x_n^{(8)}\right]$
$\theta^{(j)} = \left[\theta_1^{(j)}, \dots \theta_k^{(j)}, \dots \theta_n^{(j)}\right]$

GSQL Training Block

```
USERs = SELECT s FROM USERs:s -(rate:e)-> MOVIE:t
        ACCUM
          DOUBLE prediction = dotProduct(s.@theta,t.@x),
          DOUBLE delta = prediction-e.rating,
          s.@Gradient += product(t.@x,delta),
          t.@Gradient += product(s.@theta,delta)
        POST-ACCUM
          s.@theta += product(s.@Gradient,-alpha),
          t.@x += product(t.@Gradient,-alpha);
```

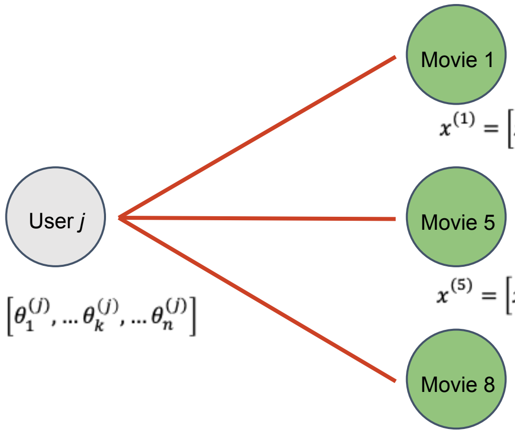


GSQL Training Block

```
USERs = SELECT s FROM USERs:s -(rate:e)-> MOVIE:t
        ACCUM
          DOUBLE delta = dotProduct_ArrayAccum_ArrayAccum(s.@theta,t.@x),
          delta = delta-e.rating,
          s.@Gradient += product_ArrayAccum_const(t.@x,delta),
          t.@Gradient += product_ArrayAccum_const(s.@theta,delta)
        POST-ACCUM
          s.@Gradient += product_ArrayAccum_const(s.@theta,lambda),
          s.@theta += product_ArrayAccum_const(s.@Gradient,-alpha),
          t.@Gradient += product_ArrayAccum_const(t.@x,lambda),
          t.@x += product_ArrayAccum_const(t.@Gradient,-alpha);
```

$$\frac{\partial J}{\partial \theta_k^{(j)}} = \sum_{i:r(i,j)=1}^{M} \left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)} \right) x_k^{(i)} + \lambda \theta_k^{(j)}$$

$$\theta_k^{(j)} = \theta_k^{(j)} - \alpha \frac{\partial J}{\partial \theta_k^{(j)}}$$



$$x^{(1)} = \left[x_1^{(1)}, \dots x_k^{(1)}, \dots x_n^{(1)}\right]$$

$$\theta^{(j)} = \left[\theta_1^{(j)}, \dots \theta_k^{(j)}, \dots \theta_n^{(j)}\right]$$

$$x^{(5)} = \left[x_1^{(5)}, \dots x_k^{(5)}, \dots x_n^{(5)}\right]$$

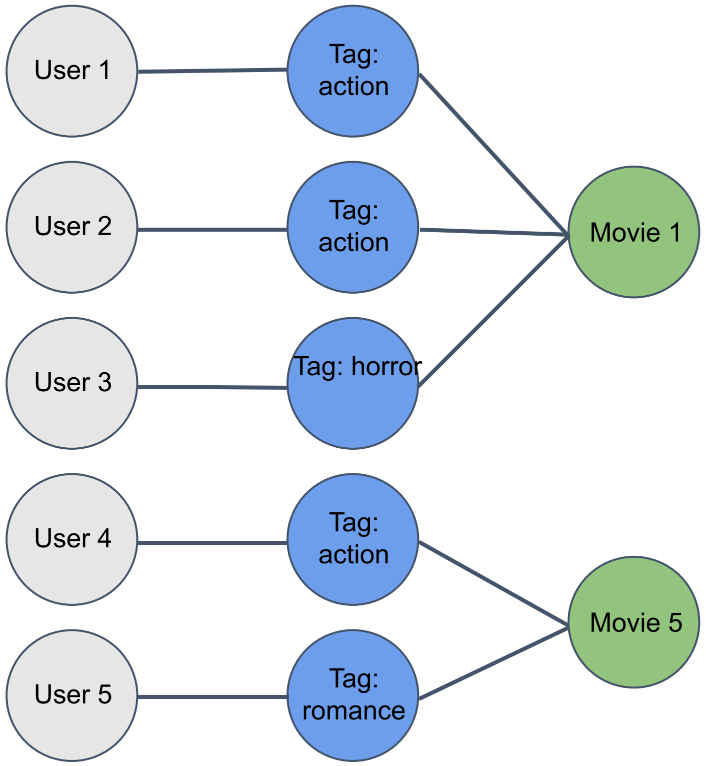
$$x^{(8)} = \left[x_1^{(8)}, \dots x_k^{(8)}, \dots x_n^{(8)}\right]$$

TigerGraph

```
"MOVIEs": [
  {
    "v_id": "318",
    "v_type": "MOVIE",
    "attributes": {
      "MOVIEs.name": "\"Shawshank Redemption",
      "MOVIEs.@rating_prediction": 3.52554,
      "MOVIEs.@rating_label": 4,
      "MOVIEs.avg_rating": 0
    }
  },
  {
    "v_id": "858",
    "v_type": "MOVIE",
    "attributes": {
      "MOVIEs.name": "\"Godfather",
      "MOVIEs.@rating_prediction": 3.36161,
      "MOVIEs.@rating_label": -1.797693134862315 7e+308,
      "MOVIEs.avg_rating": 0
    }
  },
  {
    "v_id": "50",
    "v_type": "MOVIE",
    "attributes": {
      "MOVIEs.name": "\"Usual Suspects",
      "MOVIEs.@rating_prediction": 3.32001,
      "MOVIEs.@rating_label": 3.5,
      "MOVIEs.avg_rating": 0
    }
```

TigerGraph

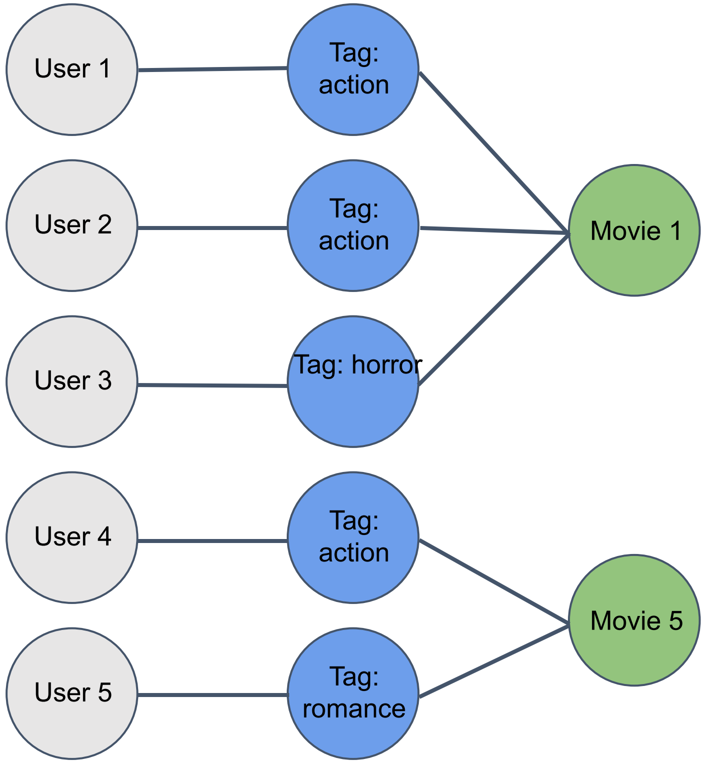
TF-IDF (Term Frequency-Inverse Document Frequency)



term	term frequency	
	Movie 1	**Movie 2**
action	2	1
horror	1	0
romance	0	1

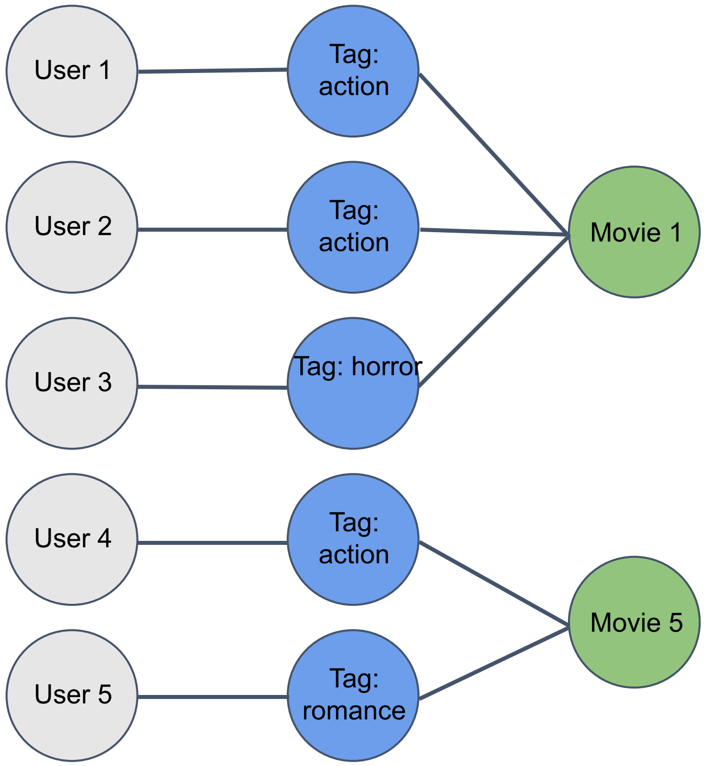
term	inverse document frequency = $\log(N_m/N_{m,t})$ N_m: number of movies $N_{m,t}$: number of movies tagged with term)
action	log(2/2) = 0
horror	log(2/1) = 0.3
romance	log(2/1) = 0.3

TF-IDF (Term Frequency-Inverse Document Frequency)



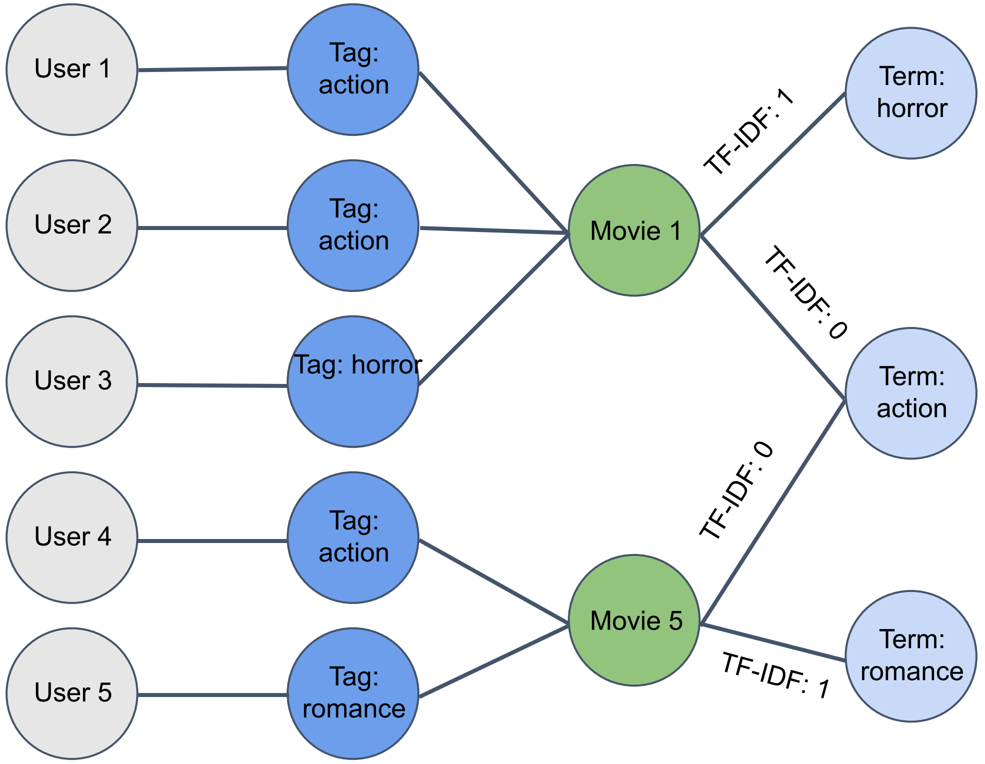
term	TF-IDF	
	Movie 1	Movie 2
action	2x0 = 0	1x0 = 0
horror	1x0.3 = 0.3	0x0.3 = 0
romance	0x0.3 = 0	1x0.3 = 0.3

TF-IDF (Term Frequency-Inverse Document Frequency)

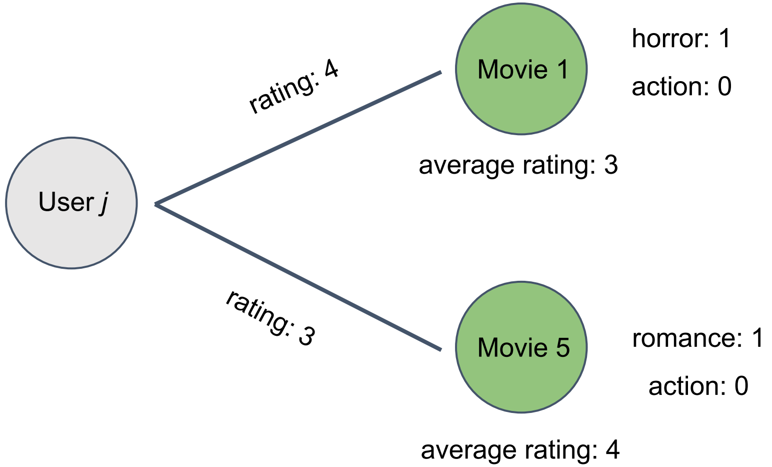


term	**TF-IDF**	
	Movie 1	**Movie 2**
action	0	0
horror	1	0
romance	0	1

TF-IDF

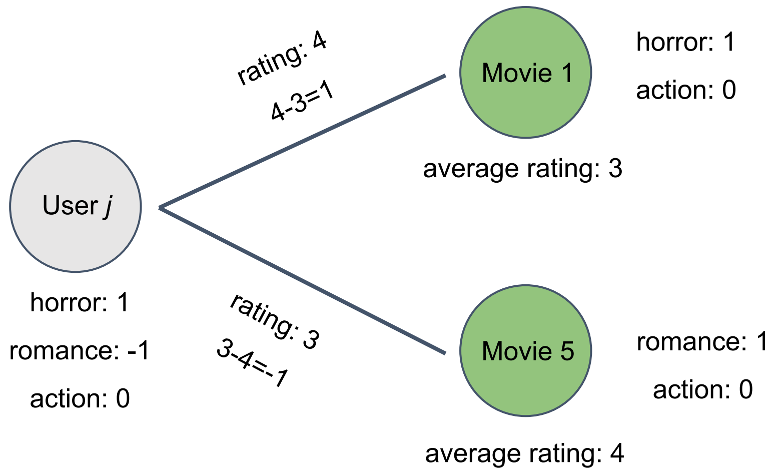

User 1
User 2
User 3
User 4
User 5
Tag: action
Tag: action
Tag: horror
Tag: action
Tag: romance
Movie 1
Movie 5
TF-IDF: 1
TF-IDF: 0
TF-IDF: 0
TF-IDF: 1
Term: horror
Term: action
Term: romance

TF-IDF



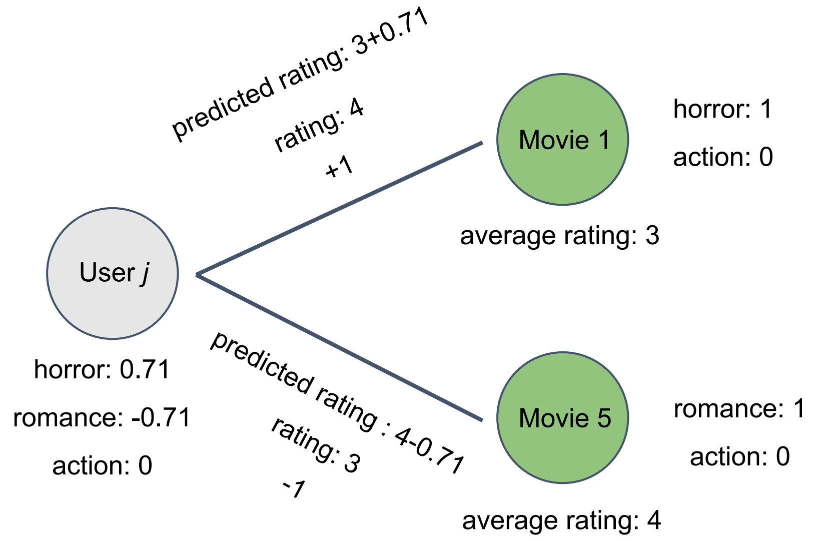
- For each user, difference between his/her rating to a movie and the average rating of the movie is computed
- The user profile vector is computed as the sum of the feature vectors of the movie he/she rated weighted by the difference above.
- The predicted rating is computed as the product of user's and movie's vectors plus the average rating.

TF-IDF



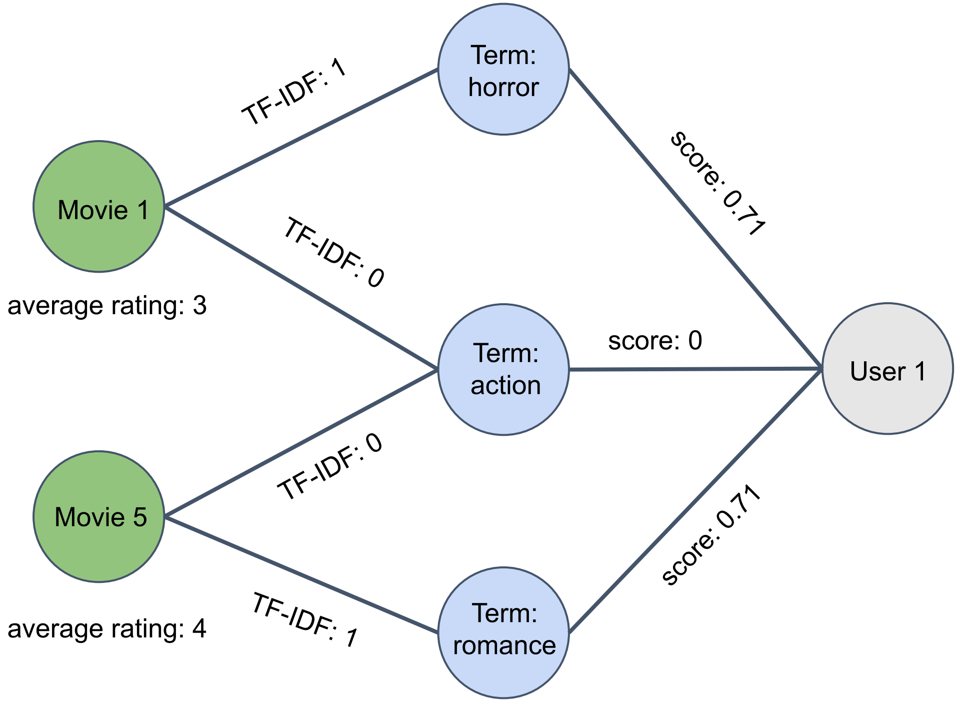
- For each user, difference between his/her rating to a movie and the average rating of the movie is computed
- The user profile vector is computed as the sum of the feature vectors of the movie he/she rated weighted by the difference above.
- The predicted rating is computed as the product of user's and movie's vectors plus the average rating.

TF-IDF



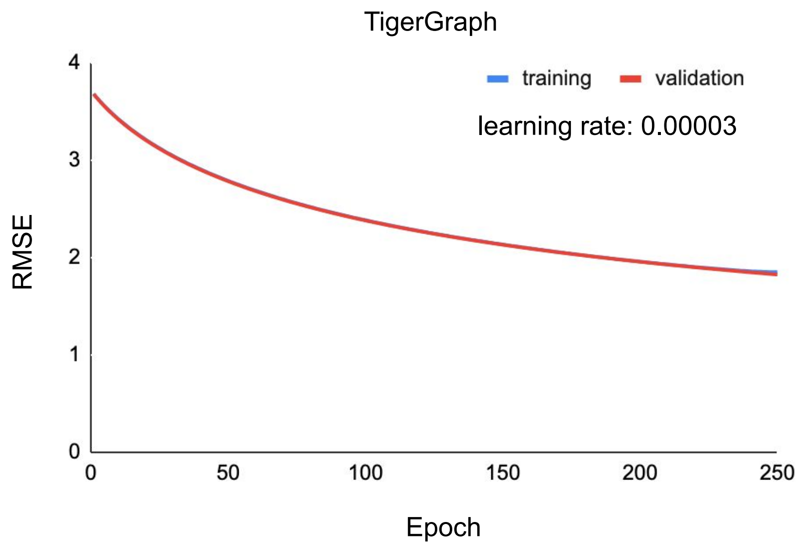
- For each user, difference between his/her rating to a movie and the average rating of the movie is computed
- The user profile vector is computed as the sum of the feature vectors of the movie he/she rated weighted by the difference above.
- The predicted rating is computed as the product of user's and movie's vectors plus the average rating.

TF-IDF

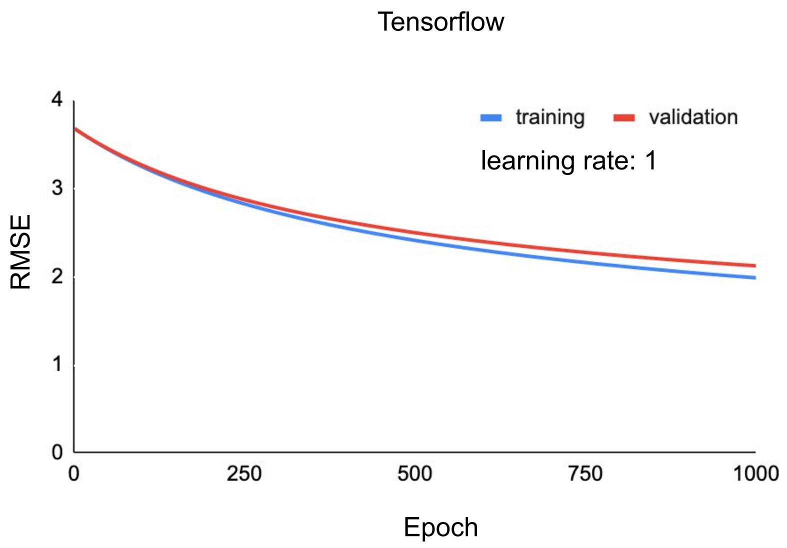

TF-IDF: 1
TF-IDF: 0
TF-IDF: 0
TF-IDF: 1
Term: horror
Term: action
Term: romance
Movie 1
average rating: 3
Movie 5
average rating: 4
score: 0.71
score: 0
score: 0.71
User 1

TigerGraph vs. Tensorflow

- 20,000,263 ratings from 138,493 users and 27,278 movies
- 70% training, 30% validation



Running time: 6.3 s/epoch



Running time: 1.5 s/epoch

In-Database Training

- Pros:
  - Distributed model training & storage for both model and data
  - No need to export data
  - Continuous training over evolving data
  - Easy to build hybrid models {content-based + collaborative filtering (memory based + model-based)}
- Cons:
  - Longer training time (~ 4x)

MovieLens Data

- MovieLens provided a *data* set of 20m ratings and 465k tags that 138k users gave to 27k movies
- Each rating is a quadruplet of the form: `<user, movie, rating, date>`
- Each tag is a quadruplet of the form: `<user, movie, tag, date>`
- The user and movie fields are integer IDs, while grades are from 0.5 to 5.0 stars

Next

- Compare performance
  - Training time
  - Loading time
  - Memory cost
  - CPU
- On different data source:
  - MovieLens
  - Netflix prize
  - Amazon
  - ...
- With python, C++, matlab...
  - Stochastic GD
  - Alternative GD
  - https://github.com/gbolmier/funk-svd

- Hybrid model
  - accuracy
- Segment size
- All Vertex Mode
- Pointer Model:

Movie rating data

- Netflix Prize problem (https://www.kaggle.com/netflix-inc/netflix-prize-data)
  - (user, movie, date, rating)
  - minimize the RMSE (root mean squared error) when predicting the ratings on the test dataset.
- MovieLens (https://grouplens.org/datasets/movielens/)
  - 5-star rating and free-text tagging activity ()
  - MovieLens 20M movie ratings: 20 million ratings and 465,000 tag applications applied to 27,000 movies by 138493 users between January 09, 1995 and March 31, 2015. All selected users had rated at least 20 movies.
-

Netflix Prize Problem

- Netflix provided a *training data* set of 100,480,507 ratings that 480,189 users gave to 17,770 movies
- Each training rating is a quadruplet of the form: `<user, movie, rating, date>`
- The user and movie fields are integer IDs, while grades are from 1 to 5 (integral) stars
- The goal is to minimize the RMSE (root mean squared error) when predicting the ratings on the *test dataset*.

Model-Based Collaborative Filtering

	user latent features				movie latent features	
	$\theta^{(1)} = [5, 0]$	$\theta^{(2)} = [5, 0]$	$\theta^{(3)} = [0, 5]$	$\theta^{(4)} = [0, 5]$		
Movie	**Alice (1)**	**Bob (2)**	**Carol (3)**	**Dave (4)**	x_1 **(romance)**	x_2 **(action)**
Love at last	5	5	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	4	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9

TigerGraph