Artificial Intelligence (CS303)

Lecture 13: Knowledge Graph

Hints for this lecture

- **A less rigorous (but more practical) way to represent and utilize knowledge.**

Outline of this lecture

- **Overview of Knowledge Graph (KG).**

- **How to construct KG?**

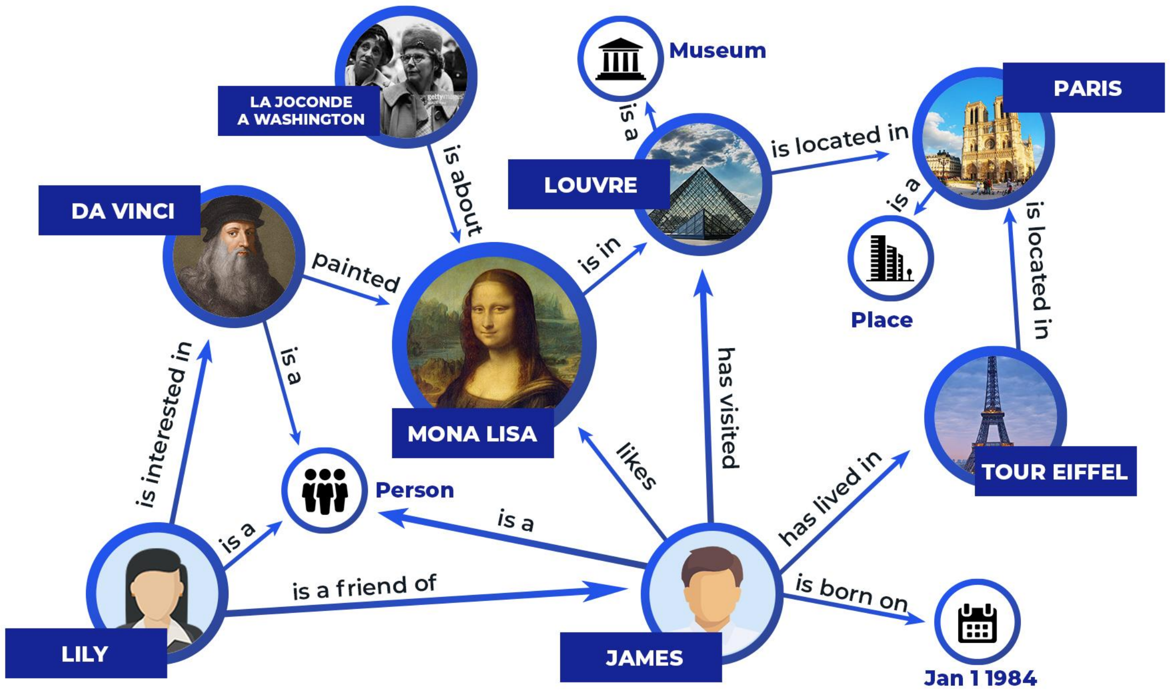
I. Overview of knowledge graph

What is Knowledge Graph?

- To make a knowledge base (KB) of practical significance, we need to:
  - Set a proper boundary for “knowledge”, which means:
    - bound the scope of the KB (and thus its representation)
    - bound the utility (application) of the KB
- In 2012, Google first proposed the concept of the knowledge graph (KG).
- The idea of KG stems from Semantic Network.
  - Knowledge Graph: ***Large-scale*** semantic network

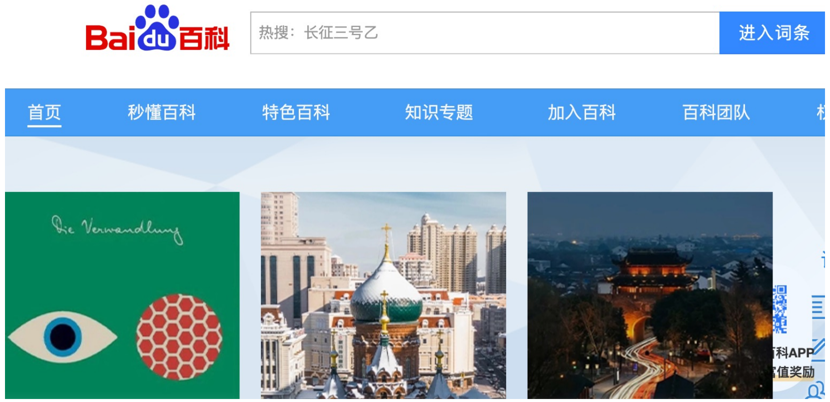
What is Knowledge Graph?

- SN/KG uses vertexes and edges to represent knowledge graphically.
  - Vertexes: entities and concepts
  - Edges: relations and properties



Why Knowledge Graph

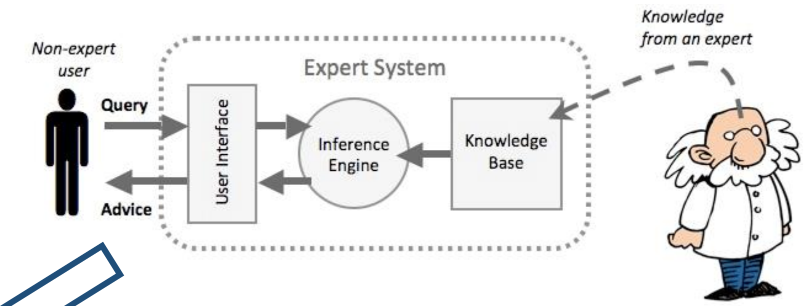
- Most human knowledge is expressed in natural language.
- It is more intuitive for human is to remember relationship between entities.
- In some (but significant) applications, we care more about entities, but not knowledge.
    - Recommender System
    - Q&A System
    - Information Retrieval

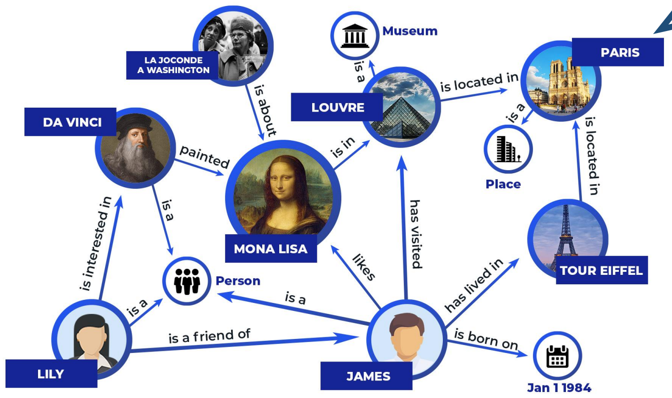


Why Knowledge Graph

Represent KB with Logic

- What is commonsense knowledge?
- Hard (even for human) to write the KB
- hard to express hidden knowledge.





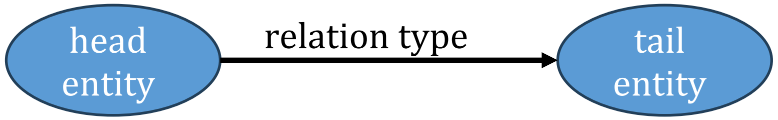
Represent KB with KG

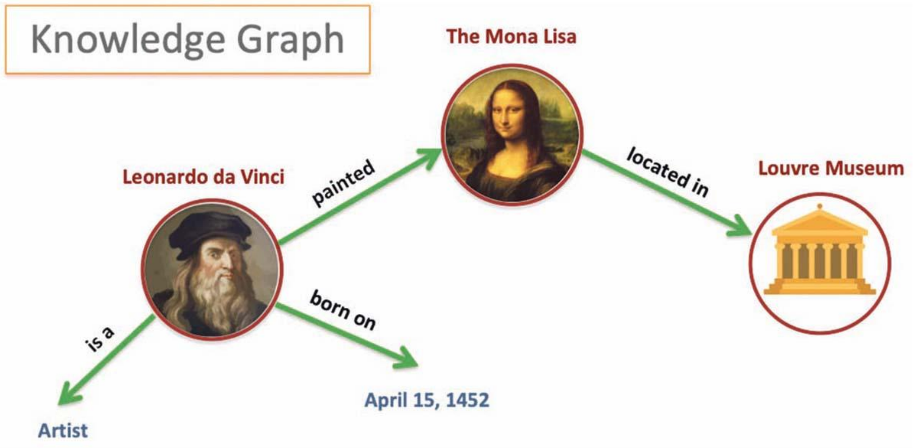
- Narrow down the definition of “commonsense knowledge”
- Can represent hidden knowledge.
- Applications in multi-domains.

II. How to construct Knowledge Graph？

KG as a Heterogeneous Directed Graph

- Heterogeneous directed graphs.
  - The KG can be represented as a graph $\mathcal{G} = (V, E)$, V is vertex set, E is the edge set.
  - V is also the entities set. E is also the relation set.
- RDF：Resource Description Framework, an XML Document standard from W3C
  - use relation triplet $< \text{head entity}, \text{relation type}, \text{tail entity} >$ to describe a relation.
    - Head entity: the subject of this relation
    - Relation type: the category of this relation
    - Tail entity: the object of this relation





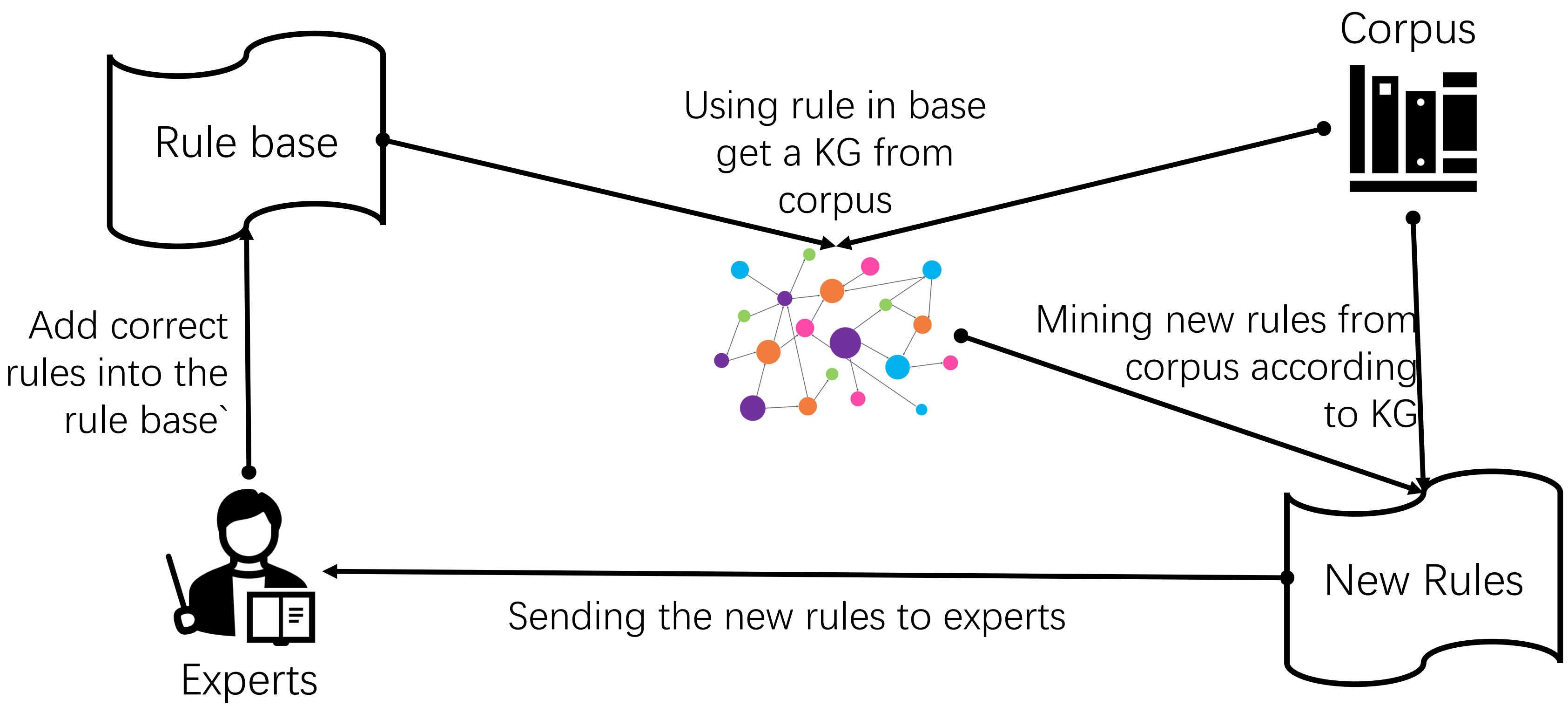
Construct a KG

- Suppose we have a lot of document written in natural language (text).
- Having experts manually identify all the entities of interest and annotate the relations between them.
  - Easier than writing a sentence in propositional/first-order logic
  - But still non-trivial

- (Semi-)Automate the construction of KG from documents?
  - Entity Recognition (Vertex set of the graph?)
  - Relation Extraction (Edge set of the graph?)
  - Automatic Annotation

The General (Semi-)Automatic Viewpoint

- Human expert write rules (template) to identify the entities and relations
  - e.g., use vocabulary/dictionary
- Is there any method to further automatically discover rules?



Automatic Entity Recognition

- Identify meaningful entities across various texts.
- Input: Documents (text)
- Output: A set of entities

Automatic Entity Recognition

- Identify meaningful entities based on the statistical metrics of vocabulary across various texts.
    - **TF-IDF** (Term Frequency–Inverse Document Frequency):
        - If a word appears frequently in one document but infrequently in others, it is more likely to be a meaningful entity.

For a corpus of documents D:

- Term frequency (TF): $P(w|d)$
- Inverse document frequency (IDF): $\log\left(\frac{|D|}{|\{d \in D | w \in d\}|}\right)$ $\quad (\log(0) = 0)$
- TF-IDF: TF $\times$ IDF

Automatic Entity Recognition

- Identify meaningful entities based on the statistical metrics of vocabulary across various texts.
  - **Entropy**:
    - If a word has a rich variety of neighboring words, it is likely be a meaningful entity.

$$H(u) = -\sum_{x \in \mathcal{X}} p(x) \log p(x)$$

      - $p(x)$ is the probability of a certain left neighbor (right neighbor) word, $\mathcal{X}$ is the set of all left neighbor (right neighbor) characters of u.
      - The larger $H(u)$ is, more abundant the set of u's neighbors is.

Automatic Entity Recognition

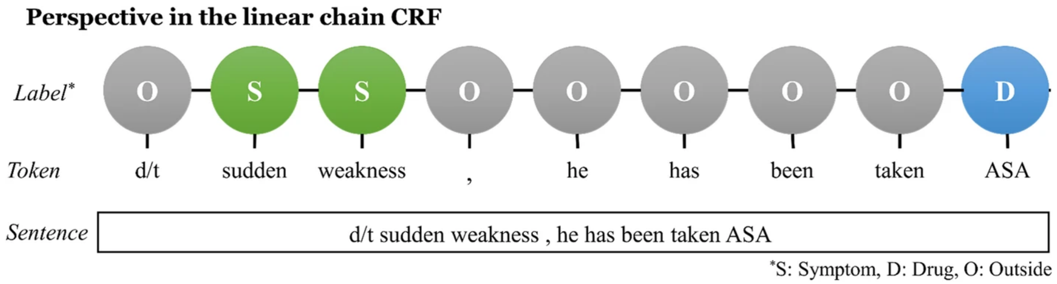
- Using machine learning techniques, model the Entity Recognition process as a Sequence Labeling problem.
  - It's also called as **NER** (Named Entity Recognition)
  - Supervised Learning
- Input is a sentence. Output is the label of each word in the sentence.

Input	Zihan	Zhang	will	join	the	ICPC
Output	B-People	I-People	O	O	O	B-Contest

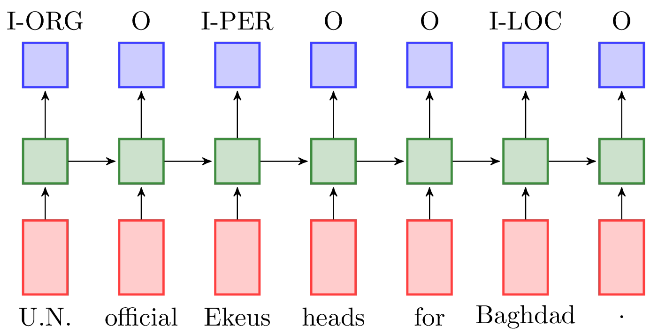
People: Zihan Zhang
Contest: ICPC

Automatic Entity Recognition

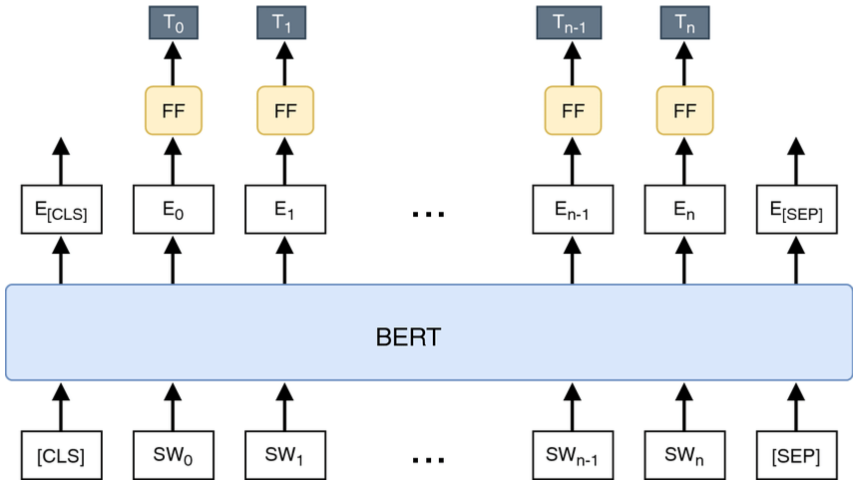
- NER task can be solved by the following technologies:



CRF (Conditional Random Field)



RNN (Recurrent Neural Network)



Transformer

Automatic Relation Extraction

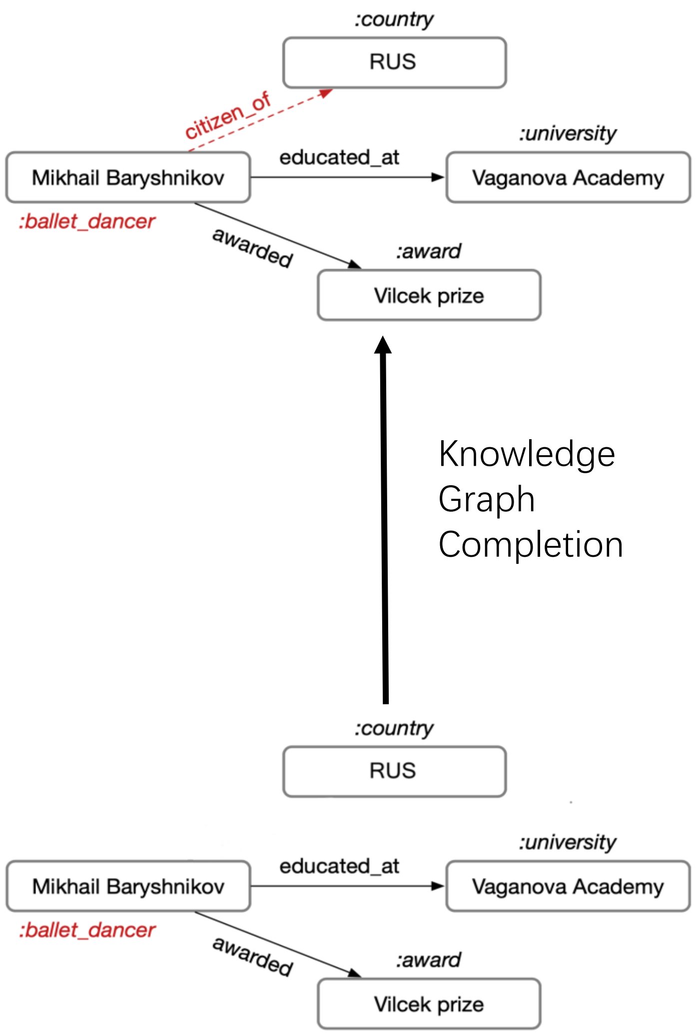
- Using machine learning techniques, model the Relation Extraction process as a Text Classification Problem.
  - It’s also a supervised learning task.
- Input is a sentence that contains 2 entities. Output is the category of the relation that the sentence express.
  - Input: Zihan Zhang will join the ICPC.
  - Output: *participate in*

Automatic Relation Extraction

- Relation extraction task can be solved by the following technologies:
  - RNN
  - Transformer
  - ...

Completing KG based on an initial KG

- KGC (Knowledge Graph Completion) task is also a method to assist KG construction.
  - Finding missing relations in an existing KG.
- Why we need KGC?
  - It's too expensive to build a thorough rule base. KGC can reduce the cost of rule base building.
  - Current relation extraction technology cannot extract all the relation.
  - The knowledge contained in textual data used to describe relationships is itself not complete.



Completing KG based on an initial KG

- There are two kinds of methods for knowledge graph completion tasks.
  - Path-based method
  - Embedding-based method

Completing KG based on an initial KG

- Path-based methods take the path between two entities in the KG as the feature of this pair of entities.

- Path is a sequence of relation types.
- E.g. In a KG, there is a relations chain between entities Qikun Xue and China:

$$\text{Qikun Xue} \xrightarrow{work\ at} \text{SUSTech} \xrightarrow{locate\ at} \text{Shenzhen} \xrightarrow{belongs\ to} \text{Guangdong} \xrightarrow{belongs\ to} \text{China}$$

  - We can extract a path $< work\ at, locate\ at, belongs\ to, belongs\ to >$
  - There could be multiple paths between two entities.

- Path-based methods determine the existence and type of relation between two entities based on the paths between them.

Completing KG based on an initial KG

- Embedding-based methods represent the entities and relation types in the KG as a low-dimensional real value vector (also called **embedding**).

- Design a score function $g(\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t})$. Get suitable embedding for entities and relation types.
  - $\boldsymbol{h}$, $\boldsymbol{r}$, and $\boldsymbol{t}$ are embeddings of head entity h, relation type r, and tail entity t respectively.
  - Higher $g(\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t})$ means that the relation (h, r, t) is more possible to be true.

Completing KG based on an initial KG

- How to get suitable embedding for entities and relation types?

- All the relations in the KG should have higher score than any relation that is not in the KG.
- We can get an objective function:

$$\min \sum_{(h,r,t)\in\mathcal{G}} \sum_{(h',r',t')\notin\mathcal{G}} [g(h',r',t') - g(h,r,t)]_+$$

- Get suitable embedding by gradient descent.

III. KG-Based Recommender System

The Need for Side Information

- In the construction of the recommender system:
  - The core data is the historical user-item interaction records.
  - Additional side information is also needed.

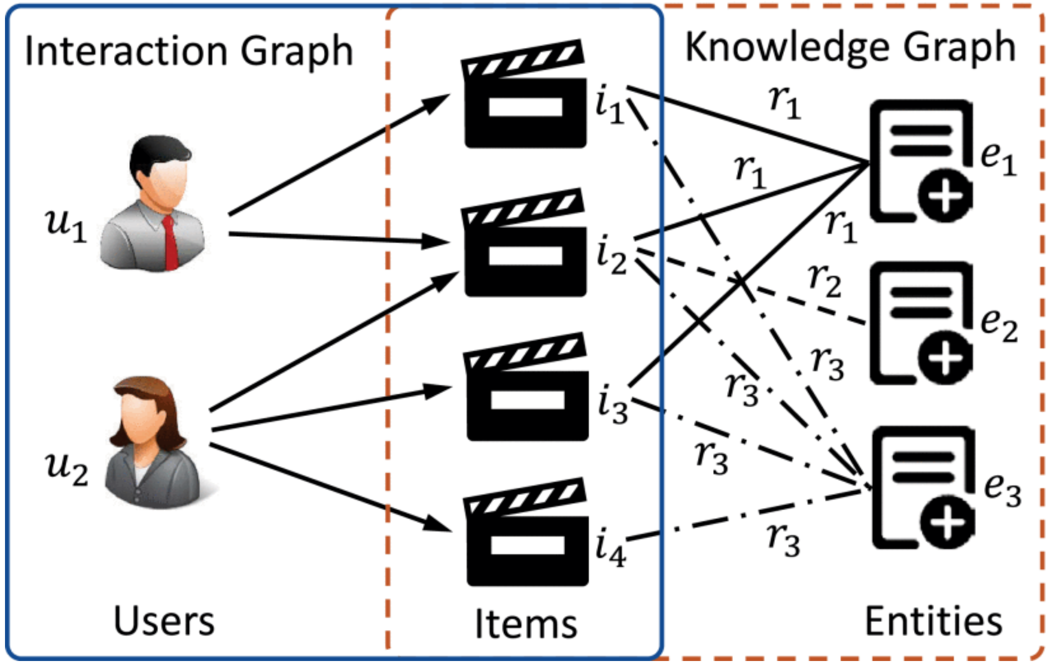
- The additional side information is used to describe users and items.
  - Sometimes, there is **no enough** interaction records.
  - Additional side information can provide more prior information would be appreciated.

Use KG as Side Information

- KG serve as a structured description of additional side (prior) information.
    - An intuitive example:胡服骑射→赵武灵王
    - Both KG and interaction data are represented as graph.
        - It is easy to integration the information in KG and interaction record.

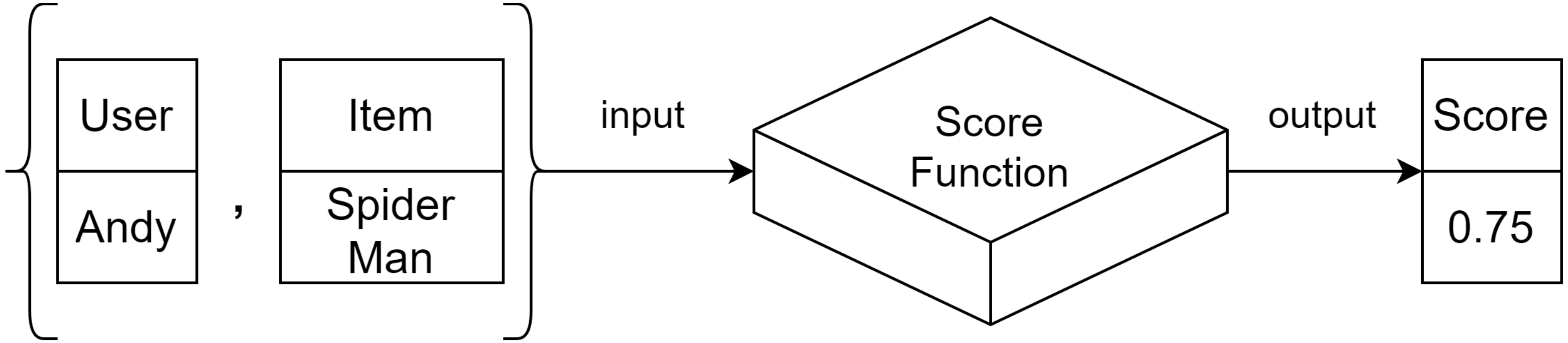
Problem Formulation

- **Input:**
  - Historical user-item interaction records.
  - A KG that is used to describe items or users (always items).
- **Output:**
  - A function used to predict how likely a user would interact with a target.



The key is also the Score Function

- Similar to the conventional recommender system, the key to a KG-based recommender system is also the score function.
- How to calculate the score?
  - Getting the **feature of the user and item** from historical user-item interaction records **and KG**.
  - Calculating score by a **well-designed model** according to the feature of the user and item.

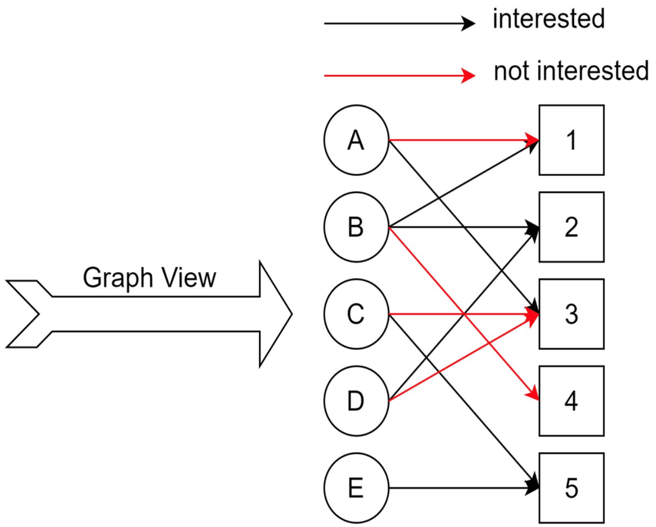


Recommender System under Graph view

- Interaction records can be seen as a bipartite graph.

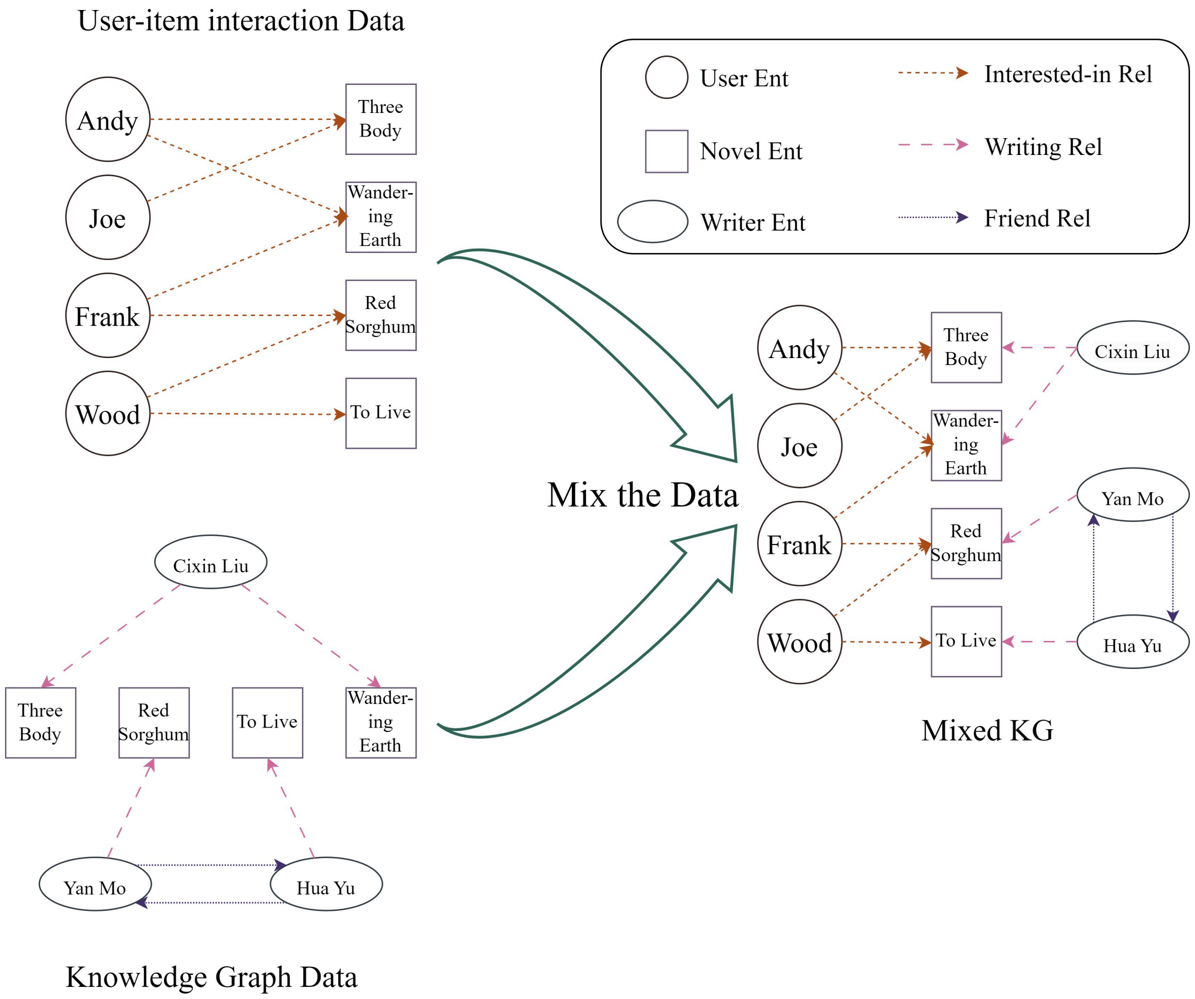
- The score function $f(u, w)$ predicts how probably there is an edge between the user u and item w, or the value of the edge between the user u and item w.

User	Item	Interest
A	1	0
A	3	1
B	1	1
B	2	1
B	4	0
C	5	1
C	3	0
D	2	1
D	3	0
E	4	1



Mix KG and Interaction Records

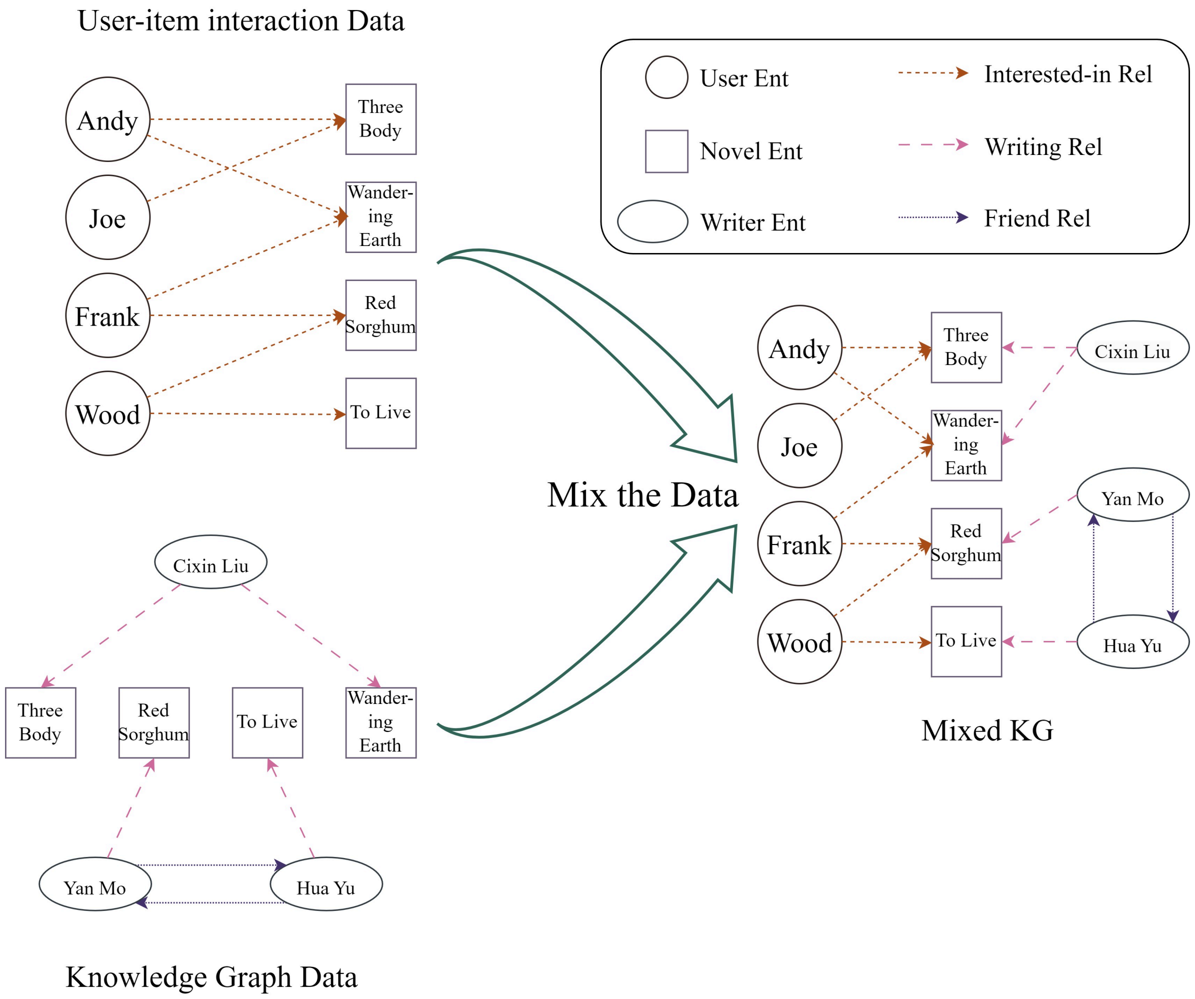
- Interaction records is a **bipartite graph**.
- KG is a **heterogeneous directed graph.**
- Mixing KG and interaction records will get a new graph.



Mix KG and Interaction Records

- We can get the **feature of the user and item** from the new graph that is mixed by KG and interaction records.

- As the content in RS and KG, the features we can get contain:
  - Paths between users and items.
  - Correlation among users or items.
  - Embeddings of users and items.



Constrain the feature of user/item

- We can represent the user/item as an embedding vector according to the interaction records.
  - There may simultaneously exist multiple representations that conform to the interaction records.

- We also can get correlations among users or items according to the KG or the mixed graph.
  - We can add a constraint that the embeddings of users/items should be consistent with the correlation among them.

Get Embedding from Mixed Graph

- We can represent the user/item as an embedding vector according to the mixed graph.
- A typical method is GNN (Graph Neural Network):
  - There is an initial embedding for each node in the graph.
  - The final embedding of each node is calculated by the embeddings of its neighborhood.
  - Result of $f(u, w)$ is calculated according to the final embeddings of user u and item w by a model M, such as MLP or matrix multiplication.

The End of the Knowledge and Reasoning Section