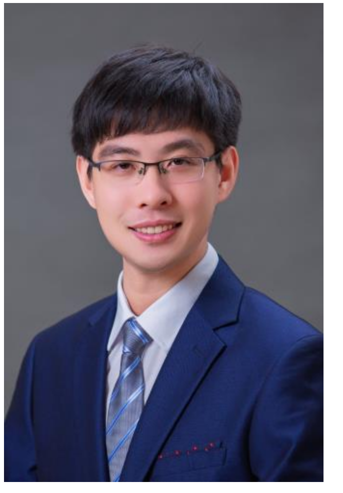
# Web Science

Prof. Tianxing Wu
School of Computer Science and Engineering
Southeast University

Email: tianxingwu@seu.edu.cn
Contact: 15077889931



# Introduction to
# Knowledge Graph Representation (III)

# Description Logic

- **Why adding such definitions in OWL? How to know them?**
  - We expect the vocabulary definition language should have following features:
    1. Easy to use and understand
    2. Formal representation
    3. Sufficient expressive ability
    4. Support automated reasoning  (decidable with suitable complexity)

**Can we use the semantics in any traditional knowledge representation language to support building OWL?**

## Description Logic

# Description Logic: Basics Review

- **Propositional Logic:** deals with propositions (which can be true or false) and relations between propositions, including the construction of arguments based on them.

  - A <u>Proposition</u> is a statement which has truth value: it is either true (T) or false (F).

Example:
Which of the following are propositions?
(a) 17 + 25 = 42 ✓
(b) July 4 occurs in the winter in the Northern Hemisphere. ✓
(c) The population of the United States is less than 250 million. ✓
(d) Is the moon round? ✗

Atomic Proposition (Statement)

# Description Logic: Basics Review

- **Propositional Logic:** deals with propositions (which can be true or false) and relations between propositions, including the construction of arguments based on them.

  - Compound proposition involves the assembly of multiple statements with logical connectives

    - Negation (not): $\neg$ ,
    - Conjunction (and): $\wedge$ ,
    - Disjunction (or): $\vee$ ,
    - Material implication (if...then): $\rightarrow$ ,
    - Biconditional (if and only if): $\leftrightarrow$ .

# Description Logic: Basics Review

- **Propositional Logic:** deals with propositions (which can be true or false) and relations between propositions, including the construction of arguments based on them.

  - Compound proposition involves the assembly of multiple statements with logical connectives

p, q: propositions

| $p$ | $q$ | $\neg p$ | $p \wedge q$ | $p \vee q$ | $p \rightarrow q$ | $p \leftrightarrow q$ |
|---|---|---|---|---|---|---|
| **F** | **F** | **T** | **F** | **F** | **T** | **T** |
| **F** | **T** | **T** | **F** | **T** | **T** | **F** |
| **T** | **F** | **F** | **F** | **T** | **F** | **F** |
| **T** | **T** | **F** | **T** | **T** | **T** | **T** |

# Description Logic: Basics Review

- **Propositional Logic:** deals with propositions (which can be true or false) and relations between propositions, including the construction of arguments based on them.

  - Compound proposition involves the assembly of multiple statements with logical connectives

  **Example:**
  **If it is sunny outside then I walk to work; otherwise I drive, and if it is raining then I carry my umbrella.**

  p= "If it is sunny outside" q= "I walk to work" r= "I drive"
  s= "It is raining" t= "I carry my umbrella"

  If p then q; otherwise r and if s then t.

  If p then q and (if not p then (r and (if s then t))).

  $(p \rightarrow q) \wedge (\neg p \rightarrow (r \wedge (s \rightarrow t)))$

# Description Logic: Basics Review

- **Propositional Logic:** deals with propositions (which can be true or false) and relations between propositions, including the construction of arguments based on them.

  - Compound proposition involves the assembly of multiple statements with logical connectives

**Rule Reasoning: Modus Ponens**

$$(p \to q) \wedge p \Rightarrow q$$

$$\begin{array}{l} \quad\quad\quad\quad p \to q \\ \quad\quad\quad\quad\quad\;\; p \\ \hline \text{Conclusion:}\;\; q \end{array}$$

If it is sunny outside then I walk to work

It is sunny outside

---

I walk to work

# Description Logic: Basics Review

- **Propositional Logic:** deals with propositions (which can be true or false) and relations between propositions, including the construction of arguments based on them.

  Problems:
  - Hard to identify "individuals" (e.g., Mary, 3)
  - Can't directly talk about properties of individuals or relations between individuals (e.g., "Bill is tall")
  - Generalizations, patterns, regularities can't easily be represented (e.g., "all triangles have 3 sides")

# Description Logic: Basics Review

- **First Order (Predicate) Logic:**
  - is a formal language in which propositions are expressed in terms of predicates (relations), variables and quantifiers.
  - contains different kinds of symbols:

  - **Constant symbols,** which represent individuals in the world
    - Mary
    - 3
    - Green
  - **Function symbols,** which map individuals to individuals
    - fatherOf(Mary) = John
    - colorOf(Sky) = Blue
  - **Predicate symbols,** which map individuals to truth values
    - greater(5,3)
    - green(Grass)
    - color(Grass, Green)

# Description Logic: Basics Review

- **First Order (Predicate) Logic:**

  - is a formal language in which propositions are expressed in terms of predicates (relations), variables and quantifiers.

  - contains different kinds of symbols:

  - Variable symbols
    - E.g., x, y,
  - Connectives
    - Same as in Propositional Logic
  - Quantifiers
    - Universal $\forall$
    - Existential $\exists$

# Description Logic: Basics Review

- **First Order (Predicate) Logic:**
  - is a formal language in which propositions are expressed in terms of predicates (relations), variables and quantifiers.
  - contains different kinds of symbols:

**Example:**
Every gardener likes the sun.
$\forall x \; gardener(x) \rightarrow likes(x, Sun)$

Some purple mushrooms are poisonous
$\exists x \; [mushroom(x) \wedge purple(x)] \rightarrow poisonous(x)$

Nigiri is a type of Sushi which has ingredient Rice and Fish.
$\forall x.[Nigiri(x) \rightarrow Sushi(x) \wedge \exists y.[hasIngredient(x, y) \wedge Rice(y)]$
$\wedge \exists z.[hasIngredient(x, z) \wedge Fish(z)]]$

# Description Logic

- **Description Logic:**
  - Fragments of first order logic (FOL)
  - Desirable computational properties: decidable and lower complexity than FOL
  - Succinct and variable free syntax

Example:
Natural Language: Nigiri is a type of Sushi which has ingredient Rice and Fish

FOL:
$\forall x.[Nigiri(x) \rightarrow Sushi(x) \land \exists y.[hasIngredient(x,y) \land Rice(y)] \land \exists z.[hasIngredient(x,z) \land Fish(z)]]$

Description Logic:
$Nigiri \sqsubseteq Sushi \sqcap \exists hasIngredient.Rice \sqcap \exists hasIngredient.Fish$

# Description Logic (DL)

- **Description Logic (DL):**
  - OWL is essentially based on DL,
  - Numerous reasoners:

| | | | |
|---|---|---|---|
| Quonto | JFact | FaCT++ | RacerPro |
| Owlgres | Pellet | SHER | snorocket |
| OWLIM | Jena | Oracle | Prime |
| QuOnto | Trowl | HermiT | condor |
| CB | ELK | konclude | RScale |
| DReW | Clipper | Ontop | |

# DL Building Blocks

- **Description Logic (DL):**
  - individuals: birte, cs63.800, sebastian, etc.
  - concepts: Person, Course, Student, etc.
  - roles: hasFather, attends, worksWith, etc.

| FOL | OWL | DL |
|---|---|---|
| constant | individual | individual |
| unary predicate | class | concept |
| binary predicate | property | role |

# Constituents of a DL Knowledge Base

| | |
|---|---|
| TBox $\mathcal{T}$ | Ontology (aka **Tbox**, Terminological Box) axioms define terminology (schema). |
| ABox $\mathcal{A}$ | Ground facts (aka **Abox,** Assertion Box) use the terminology (data). |

# Description Logic: ALC

- **ALC**: the simplest DL

$\top$ corresponds to `owl:Thing`

$\bot$ corresponds to `owl:Nothing`

$\sqcap$ corresponds to `owl:intersectionOf`

$\sqcup$ corresponds to `owl:unionOf`

$\neg$ corresponds to `owl:complementOf`

$\forall$ corresponds to `owl:allValuesFrom`

$\exists$ corresponds to `owl:someValuesFrom`

# Description Logic: ALC

- **ALC: the simplest DL**
  - complex concepts are defined as follows:
    1. $\bot$ and $\top$ are concepts；
    2. For concepts C and D, $\neg C$, $C \sqcap D$, and $C \sqcup D$ are concepts;
    3. For a role r and a concept C, $\exists r.C$ and $\forall r.C$ are concepts

Example: Student ⊓ ∀attendsCourse.MasterCourse

FOL: $\forall x\ (Student(x) \land \forall y(attendsCourse(x,y) \land MasterCourse(y)))$

It describes the concept comprising all students that attend only master courses.

# Description Logic: ALC

- **TBox:**
  - For concepts *C*, *D*, a general concept inclusion (GCI) axiom has the form:

$$C \sqsubseteq D$$

    1. $C \equiv D$ is an abbreviation for $C \sqsubseteq D$ and $D \sqsubseteq C$.
    2. a TBox (terminological Box) consists of a set of GCIs.

TBox $\mathcal{T}$

  - Exercise:
    Please use a GCI to denote apple disjoints with pear.

$$\text{Apple} \sqcap \text{Pear} \sqsubseteq \bot$$

# Description Logic: ALC

- **ABox:**
  - an ALC ABox assertion can be of one of the following forms:
    1. *C*(*a*), called concept assertion
    2. *r*(*a*, *b*), called role assertion

  - an ABox consists of a set of ABox assertions

ABox $\mathcal{A}$

# Description Logic: Representation Examples

(1) Any student must attend at least one lecture.

$$\text{Student} \sqsubseteq \exists \text{Attend.Lecture}$$

(2) John is a brother of Mary.

$$\text{Brother(John,Mary)}$$

(3) Parents are exactly those who are mothers or fathers.

$$\text{Parent} \equiv \text{Mother} \sqcup \text{Father}$$

# Exercise

Please express the following sentences in Description Logics:

1. Animals are not vegetables.

# Summary: OWL & Description Logic

- OWL is a Web ontology language/ vocabulary definition language, which is more expressive than RDFS.
- OWL is based on description logic, and almost all variants (except OWL Full) are decidable as well as supported by numerous reasoner tools.
- OWL (RDFS) provides a large number of widely accepted and reused term (entities and relations) definitions for knowledge graph.
- OWL (RDFS) inspires the way of defining entity IDs and relation IDs in knowledge graphs.

# Introduction to
# Knowledge Graph Reasoning (I)

# Knowledge Graph Reasoning

- **KG Reasoning is to infer new knowledge from the given KG.**
- **Classification:**
  - Logical reasoning
  - Statistical reasoning

# Logical Reasoning

- **Logical Reasoning include:**
  - **Deductive Reasoning** determines whether the truth of a conclusion can be determined for that rule, based solely on the truth of the premises. 推导结论
  - **Inductive Reasoning** attempts to support a determination of the rule. 学习规则
  - **Abductive reasoning** selects a cogent set of preconditions. 寻找前提

# Logical Reasoning

- **Example:**

pre-condition: it rains
rule: if it rains, then the grass gets wet
conclusion: the grass gets wet

Deductive Reasoning: If it rains, then the grass gets wet. Since today is raining, the grass is wet. 若下雨，则草地会变湿。因为今天下雨了，所以今天草地是湿的。

Inductive Reasoning: The grass is wet each time when it rains. Thus, if it rains, then the grass gets wet. 每次下雨，草地都是湿的。所以下雨会使草地变湿。

Abductive Reasoning: If it rains, then the grass gets wet. The grass is wet because it is raining. 若下雨，草地会变湿。之所以草地是湿的，因为正在下雨。

# Deductive Reasoning

- **Classification:**
  - Forward reasoning
  - Backward reasoning

# Deductive Reasoning

- **Classification:**
  - **Forward reasoning**
  - Backward reasoning

# Deductive Reasoning: Forward Reasoning

- **Forward Reasoning** (forward chaining) starts with the available data and uses inference rules to extract more data until a goal is reached.

  - An inference engine using forward chaining searches the inference rules until it finds one where the antecedent (If clause) is known to be true.

  - When such a rule is found, the engine can conclude, or infer, the consequent (Then clause), resulting in the addition of new information to its data.

  - The forward chaining approach is often employed by expert systems.

# Forward Reasoning: Materialization

- **Materialization** is to compute all implicit statements by applying rules on assertions.

Example1:

Before reasoning

axioms

$ElementarySchoolChild \sqcap Male \sqsubseteq Boy$

$\exists hasChild.Boy \sqsubseteq Parent$

assertions

$ElementarySchoolChild(Tom)$

$Male(Tom)$

$hasChild(Jack, Tom)$

reasoning $\longrightarrow$

After reasoning

$ElementarySchoolChild \sqcap Male \sqsubseteq Boy$

$\exists hasChild.Boy \sqsubseteq Parent$

$ElementarySchoolChild(Tom)$

$Male(Tom)$

$hasChild(Jack, Tom)$

$\mathbf{Boy(Tom)}$

$\mathbf{Parent(Jack)}$

# Forward Reasoning: Materialization

Example2:

Endocarditis ⊑ ∃occur-in.Endocardium
Endocarditis occurs in endocardium.

Endocardium ⊑ ∃part-of.Heart
Endocardium is a part of Heart.

Disease ⊓ ∃occur-in.Heart ⊑ HeartDisease
The diseases occurring in heart are heart diseases.

Endocarditis ⊑ Disease
Endocarditis is a disease.

*<Records>*

| Patient | Disease | Type |
|---|---|---|
| Jack | Myocarditis | Heart Disease |
| Tom | Endocarditis | null |
| Helen | Myocarditis | Heart Disease |
| John | Endocarditis | null |
| Frank | Myocarditis | Heart Disease |

Give me the list of all heart patients.

**select** ?X
**where** Records(?X, ?Y, ?Z),
**and** ?Z = ‘Heart Disease’

<Jack, Helen, Frank>

They are Jack, Helen, Frank.

**Tom and John should also be heart patients!!**

# Forward Reasoning: Materialization

Example2:

$\text{Endocarditis} \sqsubseteq \exists\text{occur-in.Endocardium}$

$\text{Endocardium} \sqsubseteq \exists\text{part-of.Heart}$

$\text{Disease} \sqcap \exists\text{occur-in.Heart} \sqsubseteq \text{HeartDisease}$

$\text{Endocarditis} \sqsubseteq \text{Disease}$

$\text{Endocarditis} \sqsubseteq \exists\text{occur-in.Heart}$

$\text{Endocarditis} \sqsubseteq \text{HeartDisease}$

DO REASONING!

*<Records>*

| **Patient** | **Disease** | **Type** |
|---|---|---|
| Jack | Myocarditis | Heart Disease |
| Tom | Endocarditis | null |
| Helen | Myocarditis | Heart Disease |
| John | Endocarditis | null |
| Frank | Myocarditis | Heart Disease |

# Forward Reasoning: Materialization

Example2:

$\text{Endocarditis} \sqsubseteq \exists \text{occur-in.Endocardium}$

$\text{Endocardium} \sqsubseteq \exists \text{part-of.Heart}$

$\text{Disease} \sqcap \exists \text{occur-in.Heart} \sqsubseteq \text{HeartDisease}$

$\text{Endocarditis} \sqsubseteq \text{Disease}$

$\text{Endocarditis} \sqsubseteq \exists \text{occur-in.Heart}$

$\text{Endocarditis} \sqsubseteq \text{HeartDisease}$

Give me the list of all heart patients.

enrich

**select** ?X
**where** Records(?X, ?Y, ?Z),
**and** ?Z = 'Heart Disease'

*<Records>*

| Patient | Disease | Type |
|---|---|---|
| Jack | Myocarditis | Heart Disease |
| Tom | Endocarditis | Heart Disease |
| Helen | Myocarditis | Heart Disease |
| John | Endocarditis | Heart Disease |
| Frank | Myocarditis | Heart Disease |

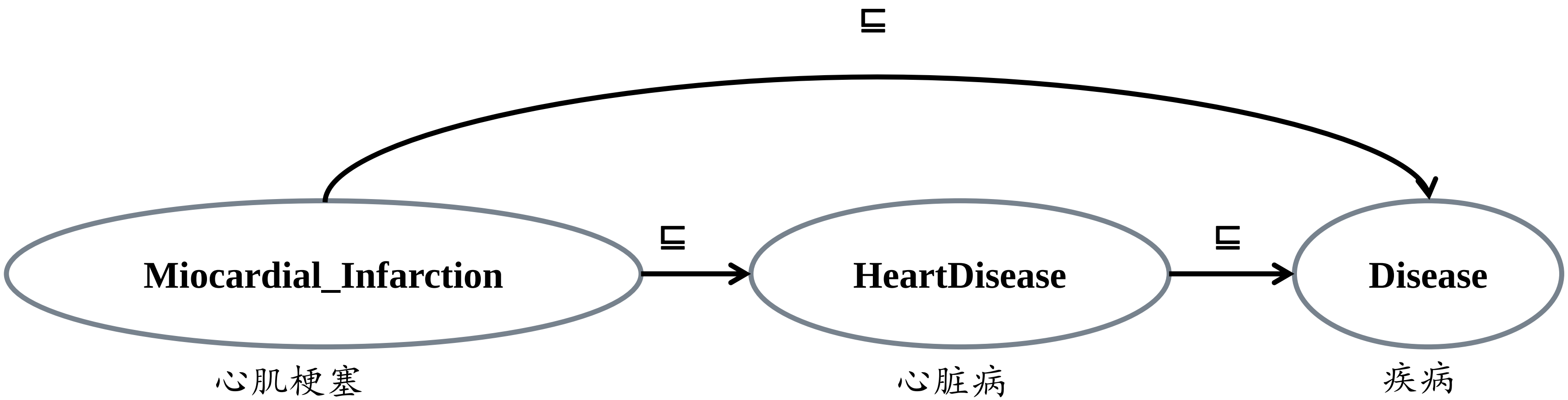
<Jack, Helen, Frank, Tom, John>

They are Jack, Helen, Frank, Tom and John.

# Forward Reasoning: Classification

- **Classification** is to compute all implicit subclass relations by applying rules on a TBox.

Example1:



# Forward Reasoning: Classification

- **Classification** is to compute all implicit subclass relations by applying rules on a TBox.

Example2:

Highvalue_Company $\sqsubseteq$ $\exists$beInvestedBy. Investment_Company
高价值公司由投资公司投资。

Investment_Company $\sqsubseteq$ Financial_Institute
投资公司属于金融机构

$\exists$beInvestedBy. Financial_Institute $\sqsubseteq$ Solvent_Company
借助金融机构投资的公司都是具备偿还能力的 企业。

Highvalue_Company $\sqsubseteq$ $\exists$beInvestedBy. Financial_Institute
Highvalue_Company $\sqsubseteq$ Solvent_Company

# Deductive Reasoning

- **Classification:**
  - Forward reasoning
  - **Backward reasoning**

# Deductive Reasoning: Backward Reasoning

- **Backward reasoning** (Backward chaining) is an inference method described colloquially as working backward from the goal.
  - It is often used in **entailment checking** or **query answering** in KG.
  - It uses the rules to **rewrite the query** in several ways and the initial query is entailed if a rewritten query maps to the initial facts.

# Deductive Reasoning: Backward Reasoning

- **Backward reasoning** (Backward chaining) is an inference method described colloquially as working backward from the goal.

Example1:

Rule: If you are human, then you are mortal.

$$Human(x) \rightarrow Mortal(x)$$

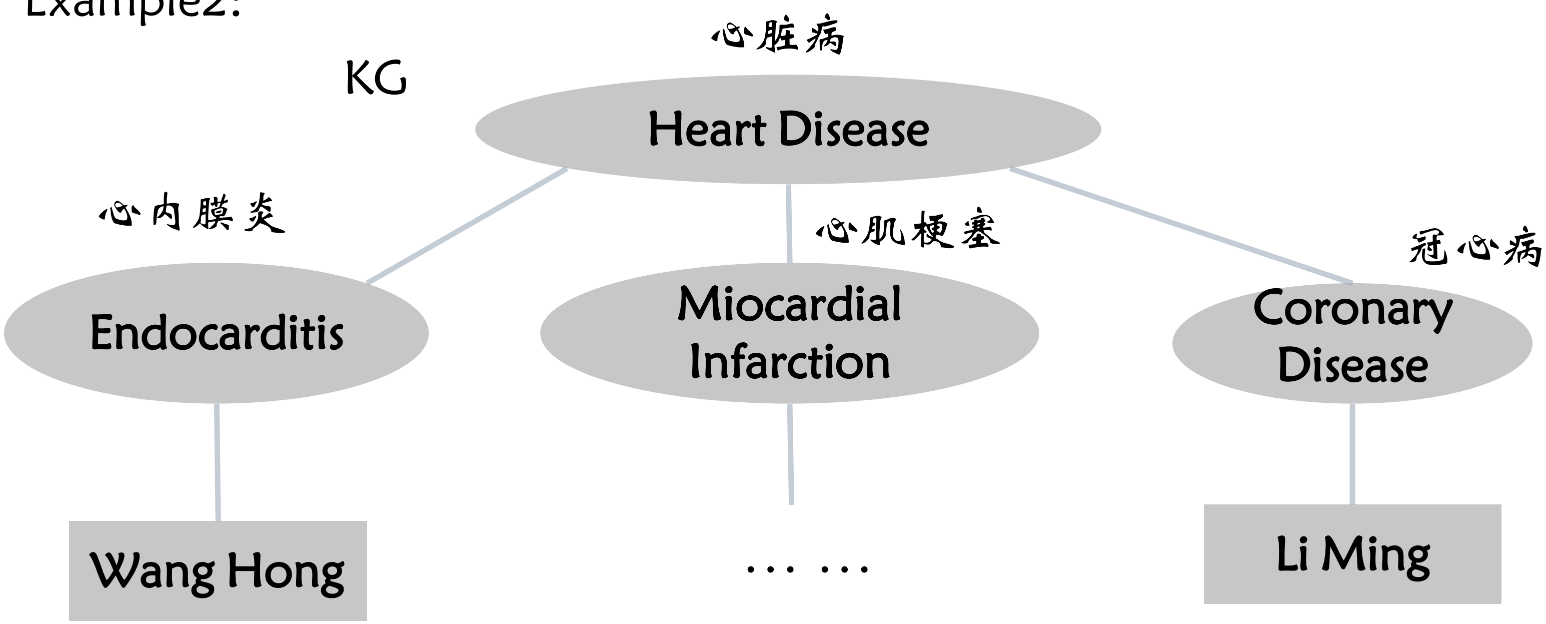
Question: Is Socrates a mortal?

Solution: Check whether Mortal(Scorates) or Human(Scorates) is true.

# Deductive Reasoning: Backward Reasoning

- **Backward reasoning** (Backward chaining) is an inference method described colloquially as working backward from the goal.

Example2:



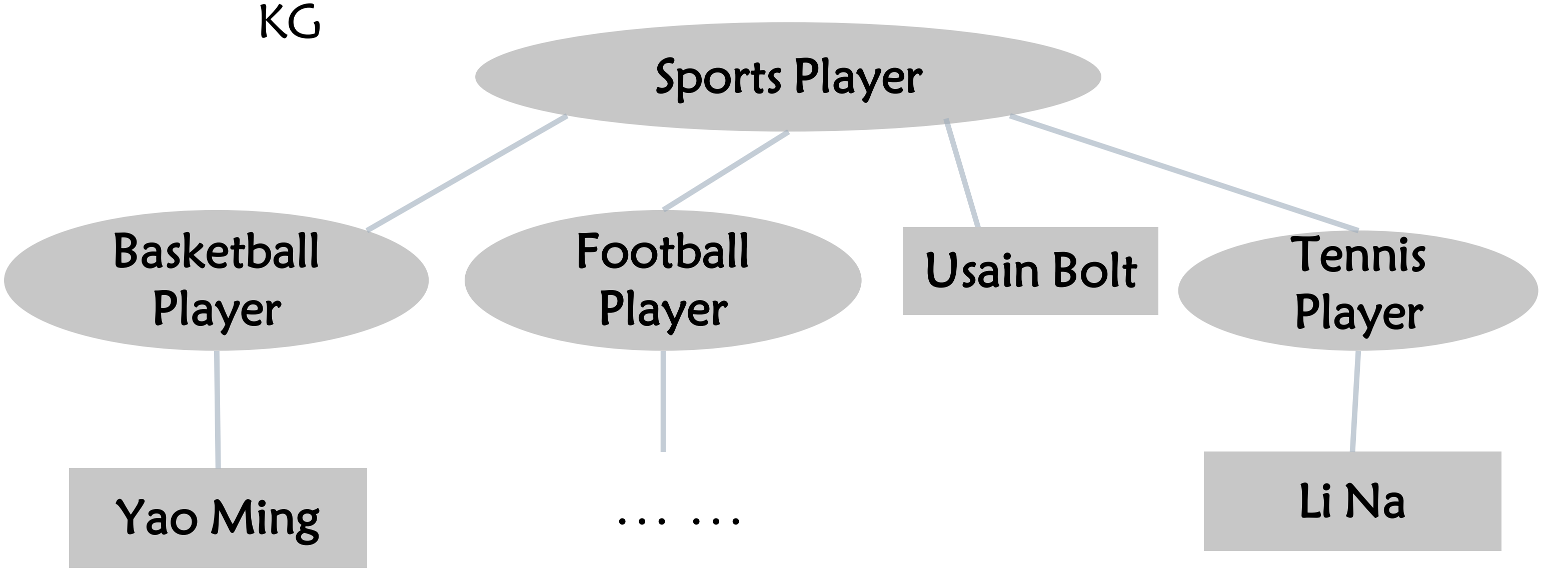

Query: Find all patients with heart diseases, i.e., Heart_Disease(x)

Rewriting: Endocarditis(x) $\vee$ Miocardial_Infarction(x) $\vee$ Coronary_disease(x)

# Exercise

Find all sports players with the following KG, i.e., rewrite the query Sports_Player(x).

KG



# Logical Reasoning: Tools

RDFox https://www.cs.ox.ac.uk/isg/tools/RDFox/

A RDF triplestore supporting Datalog Reasoning.

Drools https://www.drools.org/

A rule reasoning engine.

Jena http://jena.apache.org/

A Java framework for building Semantic Web applications
(it has a rule reasoning engine ).

# Logical Reasoning: Summary

- In logical reasoning,  we mainly focus on deductive reasoning, i.e., leveraging rules and preconditions to infer new knowledge.
- Deductive reasoning  including forward reasoning and backward reasoning.
- Forward reasoning actually gets as much knowledge as possible before application.
- Backward reasoning starts from the target/ application and can quickly get the result.

# Knowledge Graph Reasoning

- **KG Reasoning is to infer new knowledge from the given KG.**
- **Classification:**
  - Logical reasoning
  - Statistical reasoning

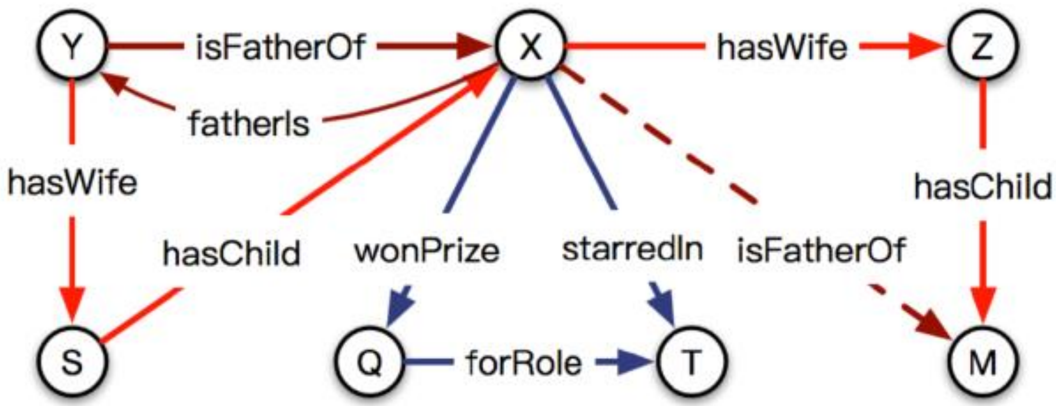
# Knowledge Graph Reasoning

- **Statistical Reasoning** tries to find suitable statistical models to fit the samples and predicts the expected probabilities of the inferred knowledge.

  - **knowledge graph embedding based reasoning**
  - inductive rule learning based reasoning
  - multi-hop reasoning

  …

# Knowledge Graph Embedding based Reasoning




- Predicting the missing link.

➤ Given $e_1$ and $e_2$, predict the relation $r$.

- Predicting the missing entity.

➤ Given $e_1$ （$e_2$） and relation $r$, predict the missing entity $e_2$ （$e_1$） .

- Fact Prediction.

➤ Given a triple, predict whether it is true or false.

# Embedding: Meaning of a Word

What is the meaning of a word?

By ontologies?
By Knowledge Graph?

But ontologies and KGs are hard to construct and often incomplete

How to encode the meaning of a word?

# One-hot Representation

Vocabulary: (cat, mat, on, sat, the)
=>
cat: 10000 mat: 01000 on: 00100
sat: 00010 the: 00001

"The cat sat on the mat"

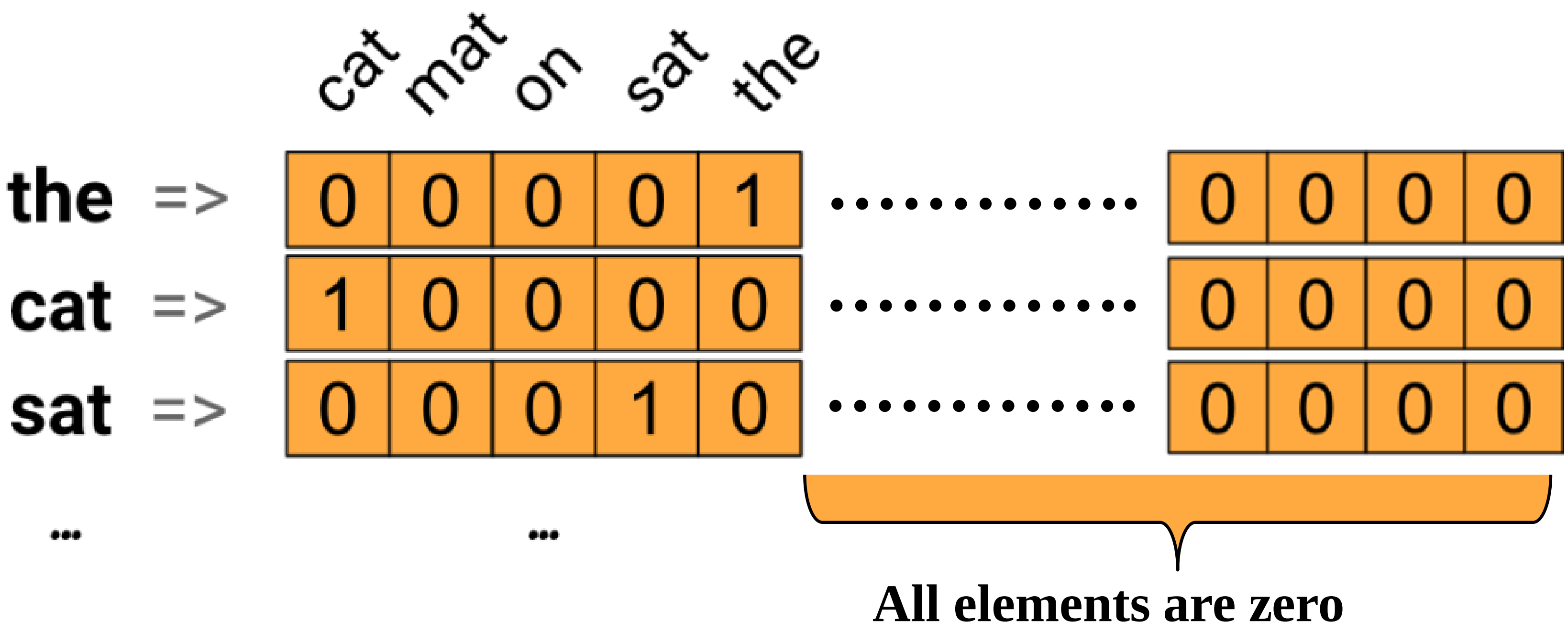
| | cat | mat | on | sat | the |
|---|---|---|---|---|---|
| **the** => | 0 | 0 | 0 | 0 | 1 |
| **cat** => | 1 | 0 | 0 | 0 | 0 |
| **sat** => | 0 | 0 | 0 | 1 | 0 |
| … | | | … | | |

# One-hot Representation

Vocabulary: (cat, mat, on, sat, the, ⋯) (>10000)

=> cat: $\underbrace{10000\cdots\cdots\cdots\cdots0}_{>10000}$

**“The cat sat on the mat”**



# One-hot Representation

- **One-hot representation:**
  - Foundation of Bag-of-words Model

star $[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, \cdots]$

sun $[0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, \cdots]$

$\text{sim}(\text{star}, \text{sun}) = 0$

# Distributional Representation

## Guiding hypotheses

Firth (1957)
"You shall know a word by the company it keeps."

Firth (1957)
"the complete meaning of a word is always contextual, and no study of meaning apart from context can be taken seriously."

Wittgenstein (1953)
"the meaning of a word is its use in the language"

Harris (1954)
"distributional statements can cover all of the material of a language without requiring support from other types of information."

Turney & Pantel (2010)
"If units of text have similar vectors in a text frequency matrix, then they tend to have similar meanings."

# Distributional Representation

When a word $w$ appears in text, its context is the set of words that appear nearby (within a fixed-size window)：

Use many contexts of $w$ to build up a representation of $w$

随着国外**新冠**患者数量的猛增，让原本松了一口气的我们，再次紧张起来。

治疗一列**新冠**轻症患者的费用在1万元上下。

肆虐的**新冠**病毒究竟长什么样？

基因测序等研究结果显示，**新冠**病毒与SARS冠状病毒同属冠状病毒科的β属冠状病毒

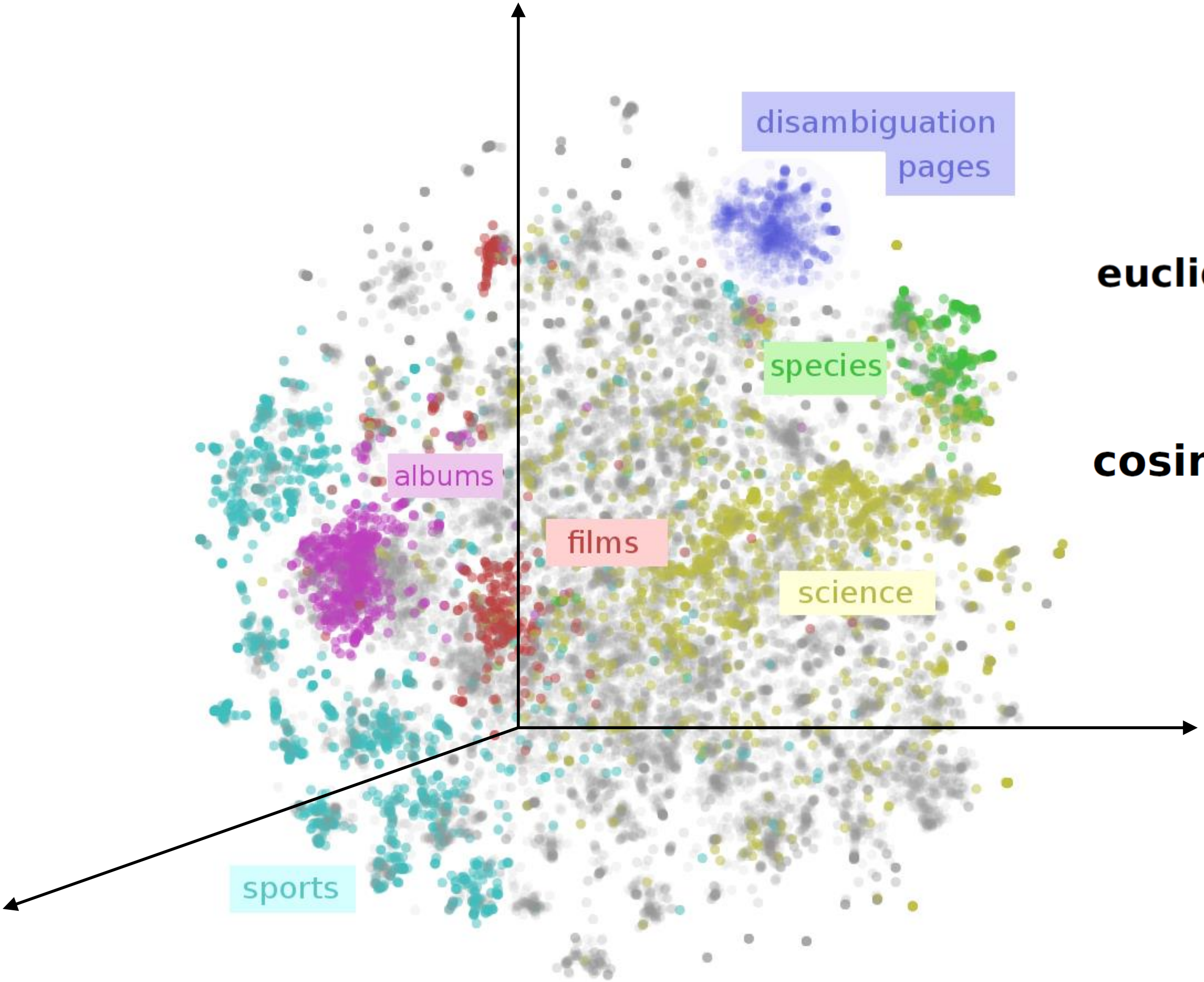
# Word Vectors

We will build a dense vector for each word, chosen so that it is similar to vectors of words that appear in similar contexts.

$$
\text{新冠} = \begin{pmatrix} 0.286 \\ 0.792 \\ -0.177 \\ -0.107 \\ 0.109 \\ -0.542 \\ 0.349 \\ 0.271 \end{pmatrix}
$$

Note: word vectors are sometimes called word embeddings. They are a distributed representation.

# Word Vectors



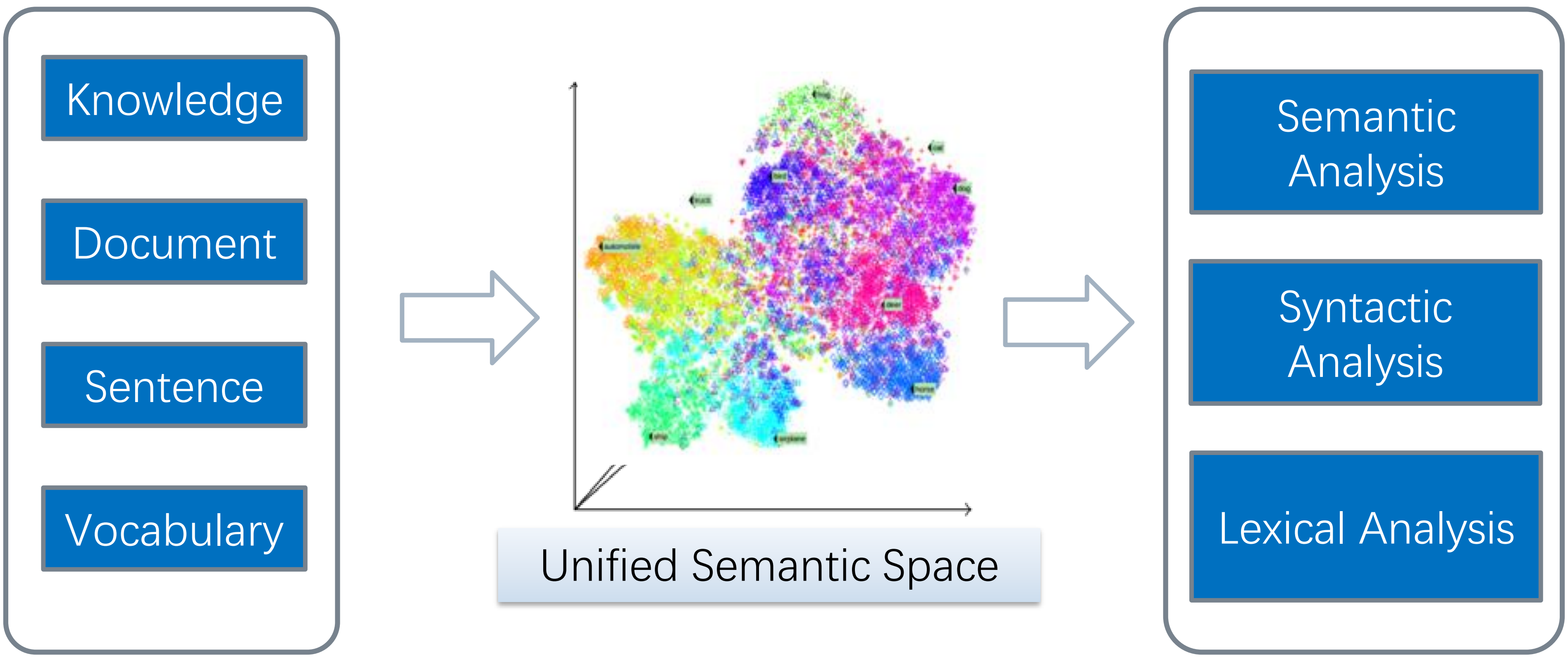

**Similarity:**

$$\textbf{euclidean}(u, v) = \sqrt{\sum_{i=1}^{n} |u_i - v_i|^2}$$

$$\textbf{cosine}(u, v) = 1 - \frac{\sum_{i=1}^{n} u_i \times v_i}{||u||_2 \times ||v||_2}$$

# Advantage of Distributed Representation

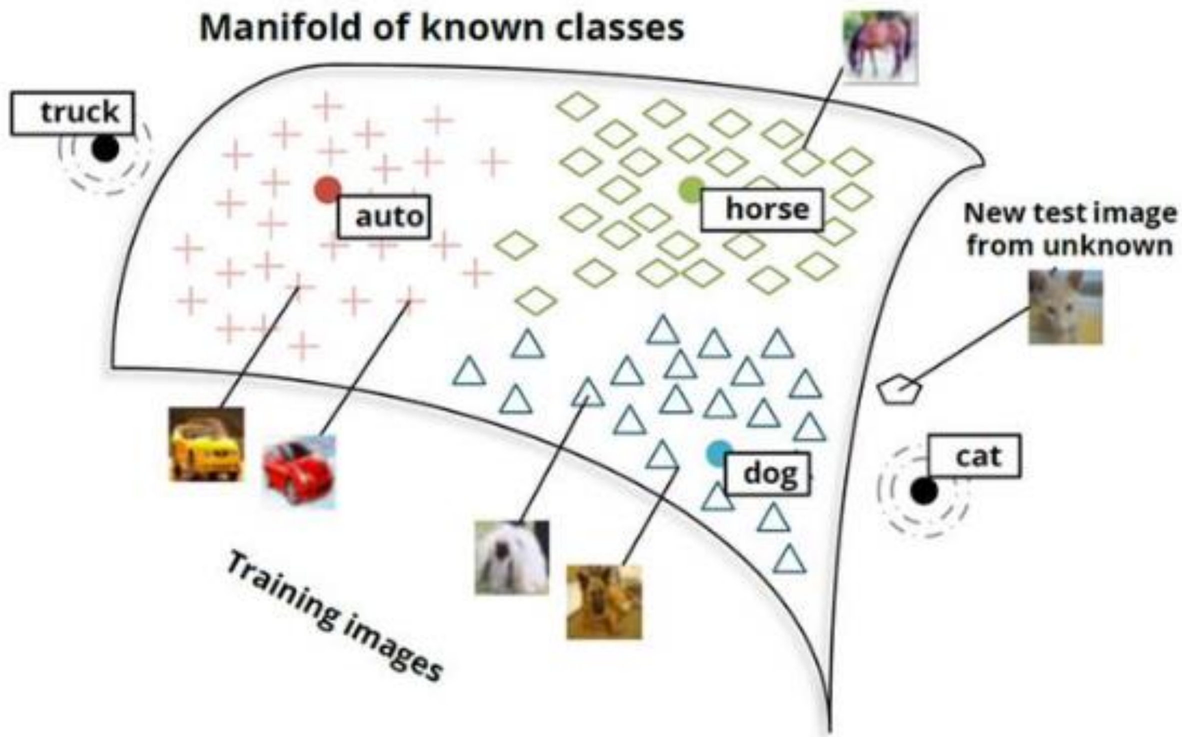
1. Deal with data sparsity problem in NLP
2. Realize knowledge transfer across domains and across objects
3. Provide a unified representation for multi-task learning



# Representation Learning

What is the representation learning?

- Objects are represented as dense, real-value and low-dimensional vector

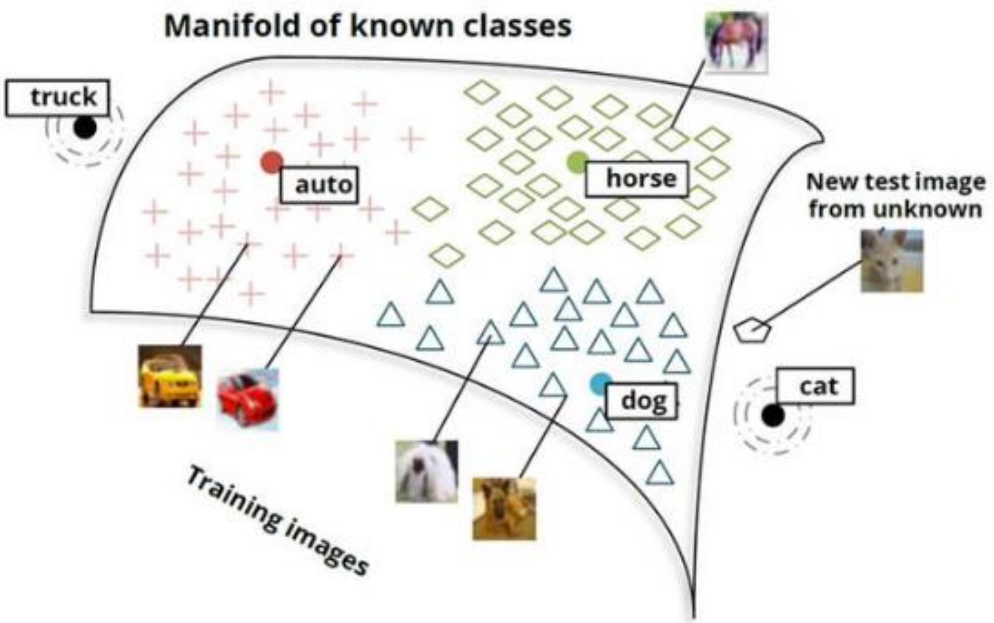


# Representation Learning

What is the representation learning?

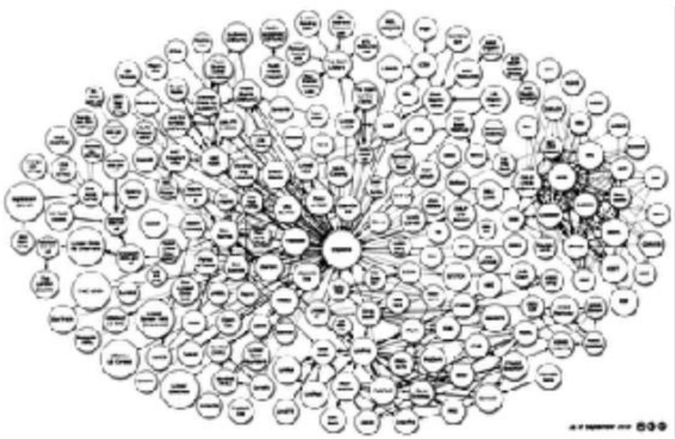
- Objects are represented as dense, real-value and low-dimensional vector

**ML= Representation + Objective +Optimize**



# Different ways of KG Representation

## Based on discrete symbols

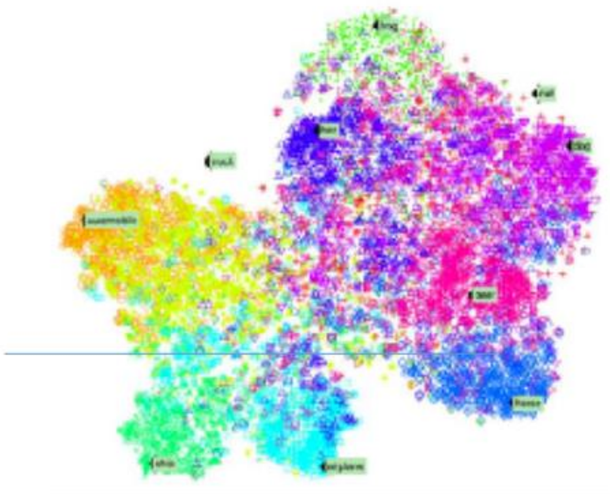


**RDF, OWL, Rule Language……**

explicit knowledge
strong logical constraints
easy to explain
not easy to expand reasoning

**vs**

## Based on continuous vectors



**Tensor, Embedding, neural network representation……**

implicit knowledge
weak logic constraints
not easy to explain
closely connected to neural networks

# Knowledge Graph Embedding: Application

- Entity Prediction

***卧虎藏龙***　　***Has-director***　　?

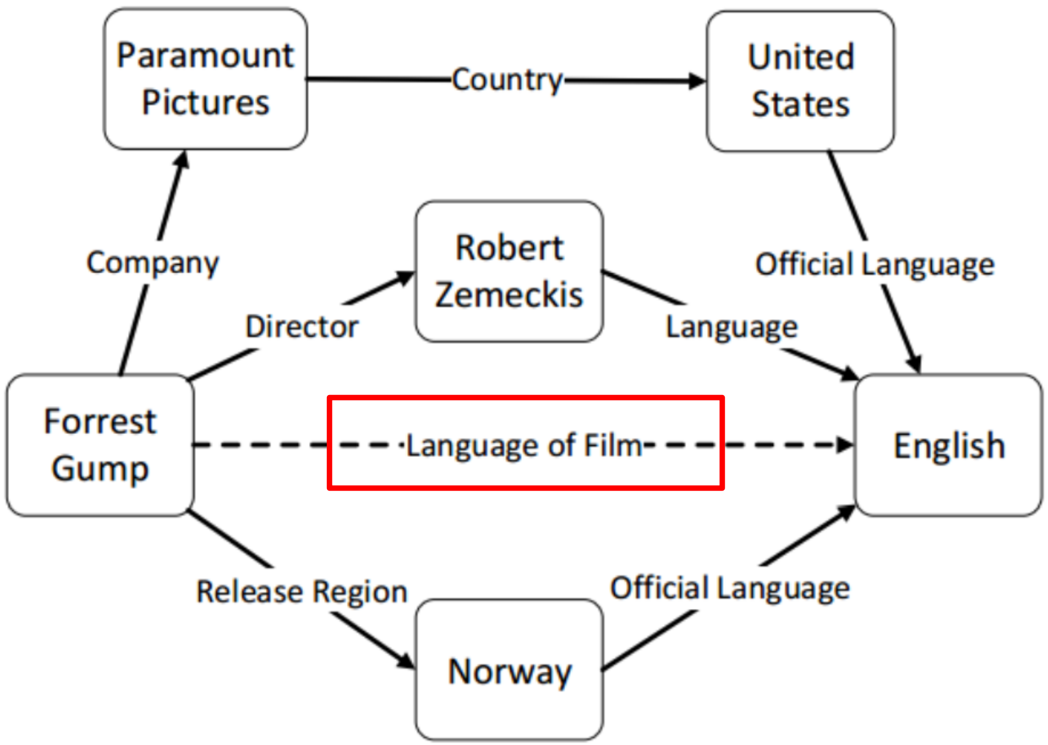
# Knowledge Graph Embedding: Application

- Entity Prediction

*卧虎藏龙*　　*Has-director*　　*Ang Lee*

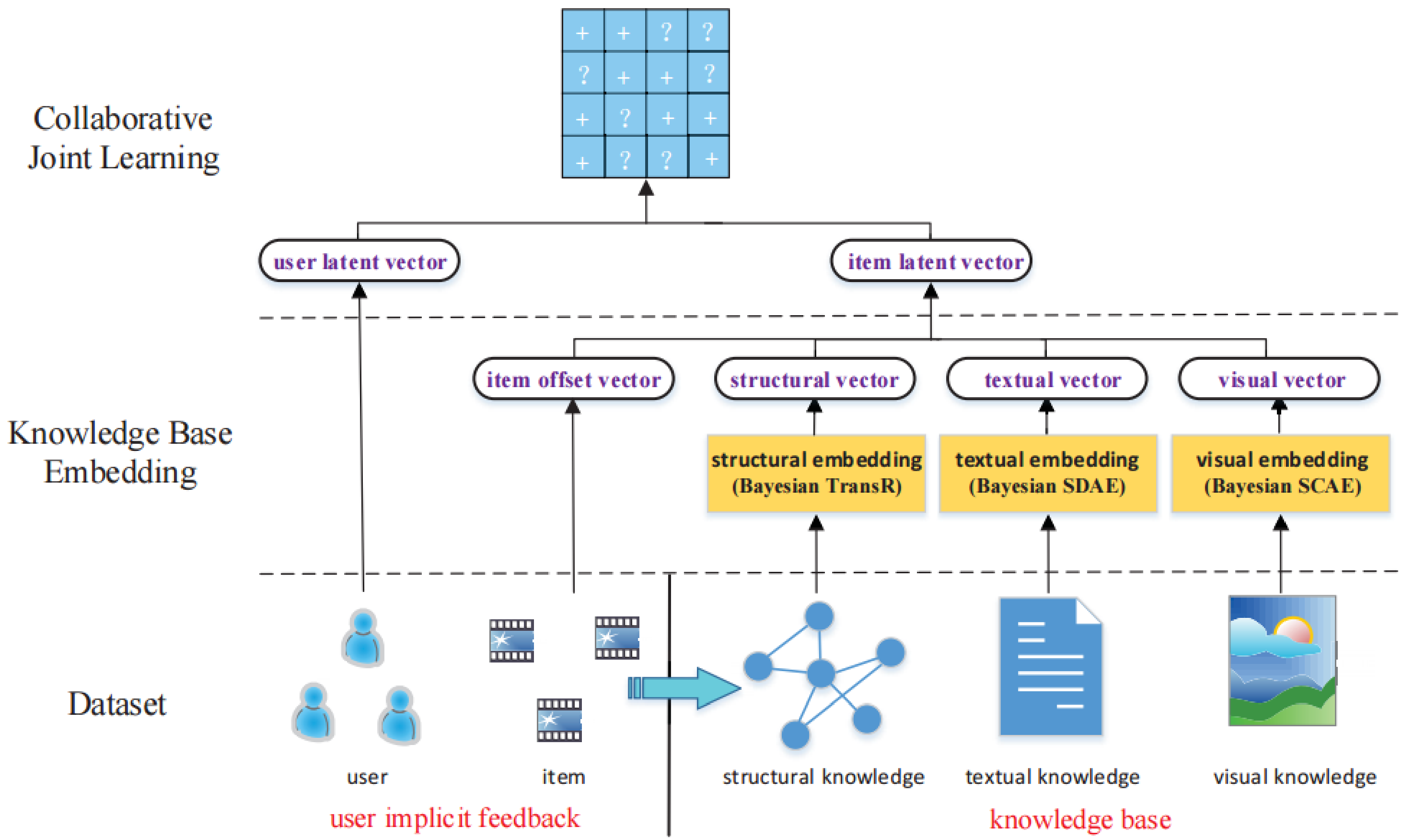
# Knowledge Graph Embedding: Application

- Relation Prediction



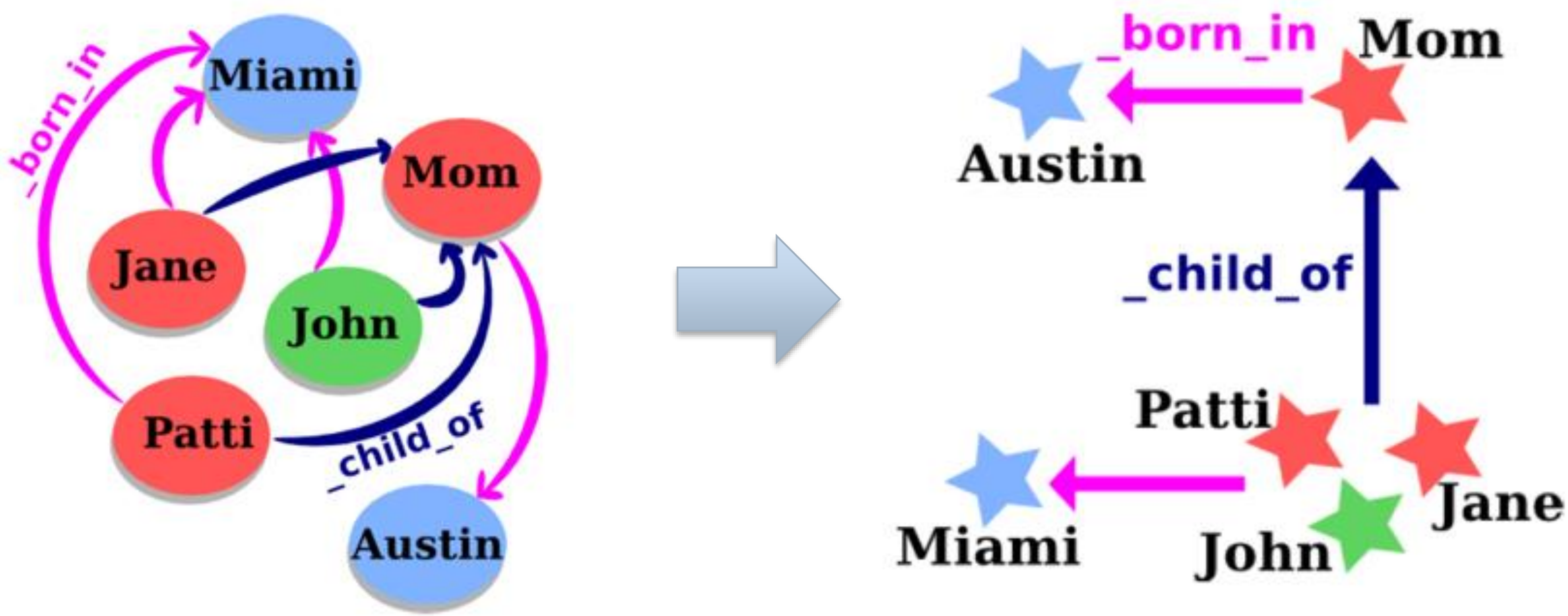
# Knowledge Graph Embedding: Application

- Recommendation System



# TransE: Take Relation as Translation

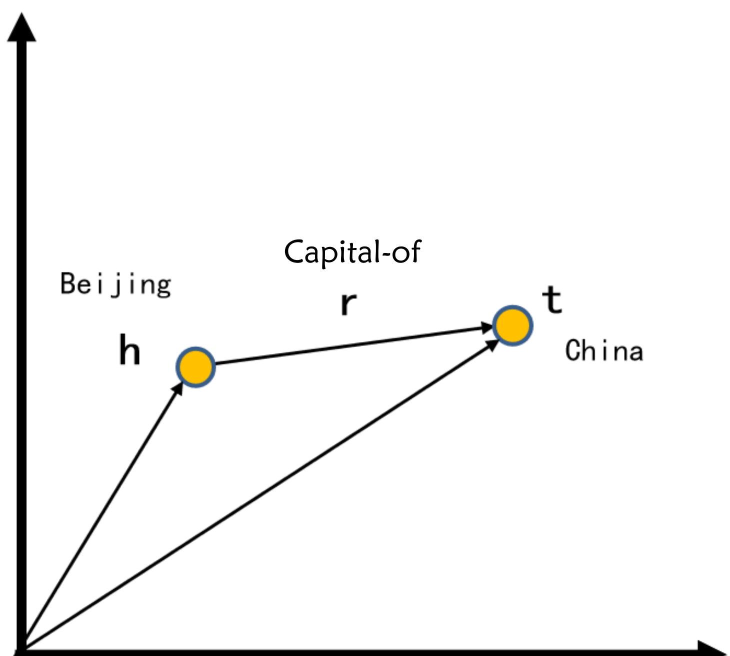
- For a fact (head, relation, tail), take the relation as a translation operator from the head to the tail .



# Translation-Based Models

- **TransE**
  - For each triple <*h*,*r*,*t*>, *h* is translated to *t* by *r.*

    *<Beijing, Capital-of , China>*




The “perfect” model

# Translation-Based Models

- **TransE**
  - For each triple $<h,r,t>$, $h$ is translated to $t$ by $r$.

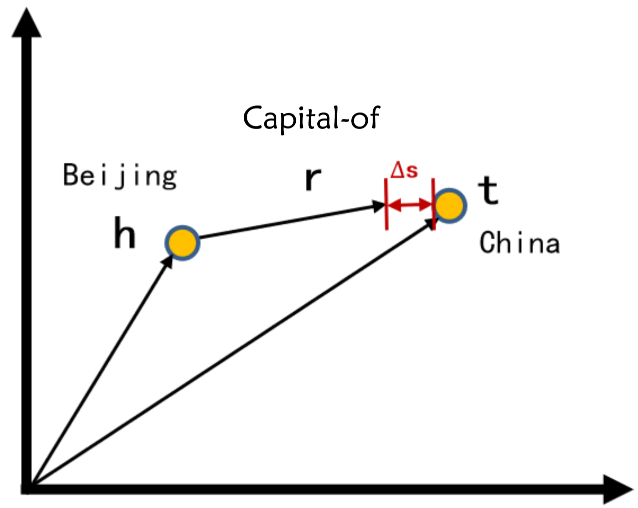
    *<Beijing, Capital-of , China>*

  - Train TransE

    Energy Function:

$$f(h,r,t) = |h + r - t|_{L_1/L_2}$$

**If the triple is true, the translated distance between *h* and *t* is shorter.**




Single triple, actual situation

# Translation-Based Models

- TransE
  - $L_1$ (Manhattan) distance:

$$\mathbf{d}_1(a,b) = \|a-b\|_1 = \sum_{i=1} |a_i - b_i|.$$

  - $L_2$ (Euclidean) distance:

$$\mathbf{d}_2(a,b) = \|a-b\| = \|a-b\|_2 = \sqrt{\sum_{i=1}^{d} (a_i - b_i)^2}$$

# Translation-Based Models

- **TransE**
  - Triple1: <Beijing, Capital-of, China>
  - Triple2: <London, Capital-of, England>
  - Triple3: <Paris, Capital-of, France>

  …

true triples

false triple examples：

<Beijing, Capital-of, England>

<London, Capital-of, France>

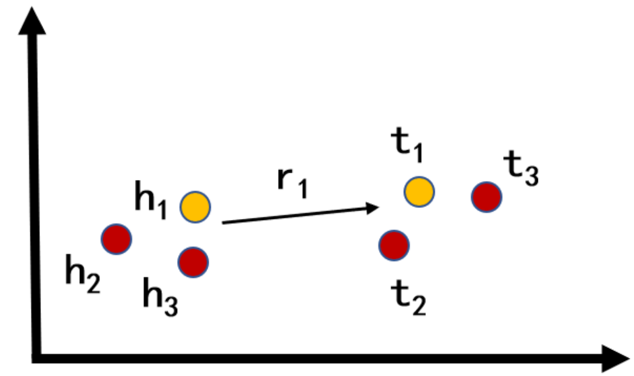
<London, Capital-of, France>

…

**How to distinguish?**

# Translation-Based Models

- **TransE**
  - Triple1: <Beijing, Capital-of, China>
  - Triple2: <London, Capital-of, England>
  - Triple3: <Paris, Capital-of, France>




  - Train TransE:
    Loss Function:

Margin

$$\mathcal{L} = \sum_{(h,l,t)\in S} \sum_{(h',l,t')\in S'_{(h,l,t)}} [\lambda + d(h+l,t) - d(h'+l,t')]_+$$

True triples

False triples

Distance function

- Minimize the distance between (h+l) and t.
- Maximize the distance between (h'+l) to a randomly sampled tail t' (negative example).

# Translation-Based Models

- **TransE**
  - Train TransE:

**Algorithm 1** Learning TransE

**input** Training set $S = \{(h, \ell, t)\}$, entities and rel. sets $E$ and $L$, margin $\gamma$, embeddings dim. $k$.

**initialize** $\boldsymbol{\ell} \leftarrow \text{uniform}(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})$ for each $\ell \in L$
    $\boldsymbol{\ell} \leftarrow \boldsymbol{\ell} / \|\boldsymbol{\ell}\|$ for each $\ell \in L$
    $\mathbf{e} \leftarrow \text{uniform}(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})$ for each entity $e \in E$

**loop**
  $\mathbf{e} \leftarrow \mathbf{e} / \|\mathbf{e}\|$ for each entity $e \in E$
  $S_{batch} \leftarrow \text{sample}(S, b)$ // sample a minibatch of size $b$
  $T_{batch} \leftarrow \emptyset$ // initialize the set of pairs of triplets
  **for** $(h, \ell, t) \in S_{batch}$ **do**
    $(h', \ell, t') \leftarrow \text{sample}(S'_{(h,\ell,t)})$ // sample a corrupted triplet
    $T_{batch} \leftarrow T_{batch} \cup \{((h, \ell, t), (h', \ell, t'))\}$
  **end for**
  Update embeddings w.r.t. $$\sum_{((h,\ell,t),(h',\ell,t')) \in T_{batch}} \nabla \left[\gamma + d(\boldsymbol{h} + \boldsymbol{\ell}, \boldsymbol{t}) - d(\boldsymbol{h'} + \boldsymbol{\ell}, \boldsymbol{t'})\right]_+$$
**end loop**

Entities and relations are initialized uniformly, and normalized

Favors lower distance (or higher score) for true triplets, high distance (or lower score) for false ones

Output: the translated distance between **h**+**l** and **t.**

# Translation-Based Models

- **TransE**
  - Train TransE:

1、 **input** Training set $S = \{(h, \ell, t)\}$, entities and rel. sets $E$ and $L$, margin λ, embeddings dim. $k$.

2、 **Initialize** entity and relationship embedding;

**uniform distribution**

# Translation-Based Models

- **TransE**
  - Train TransE:

**3**、Entity and relationship embedding **normalization**;
For each entity e （M: dimension）

$$e = \frac{e_i}{\sqrt{e_1^2 + e_2^2 + \cdots + e_M^2}}$$

**4**、Negative Sampling and

**for** $(h, \ell, t) \in S_{batch}$ **do**
$(h', \ell, t') \leftarrow$sample$(S'_{(h,\ell,t)})$ // sample a corrupted triplet
$T_{batch} \leftarrow T_{batch} \cup \{((h, \ell, t), (h', \ell, t'))\}$

stochastic gradient descent， SGD

# Translation-Based Models

- **TransE**
  - Evaluation protocol:

    **Link Prediction**

    ( WALL-E ,  _has_genre ,  ? )

  - Metrics:

    - **Mean Ranks**: the mean of those predicted ranks.

      e.g.  Entity 1: rank -> 50;  Entity 2: rank -> 100;

      MR = (50+100)/2 = 75

    - **Hits@10**: the proportion of correct entities ranked in the top 10.

# Translation-Based Models

- **TransE**
  - Link prediction results. Test performance of the different methods:

| DATASET | WN | | | | FB15K | | | | FB1M | |
|---|---|---|---|---|---|---|---|---|---|---|
| METRIC | MEAN RANK | | HITS@10 (%) | | MEAN RANK | | HITS@10 (%) | | MEAN RANK | HITS@10 (%) |
| *Eval. setting* | *Raw* | *Filt.* | *Raw* | *Filt.* | *Raw* | *Filt.* | *Raw* | *Filt.* | *Raw* | *Raw* |
| Unstructured [2] | 315 | 304 | 35.3 | 38.2 | 1,074 | 979 | 4.5 | 6.3 | 15,139 | 2.9 |
| RESCAL [11] | 1,180 | 1,163 | 37.2 | 52.8 | 828 | 683 | 28.4 | 44.1 | - | - |
| SE [3] | 1,011 | 985 | 68.5 | 80.5 | 273 | 162 | 28.8 | 39.8 | 22,044 | 17.5 |
| SME(LINEAR) [2] | 545 | 533 | 65.1 | 74.1 | 274 | 154 | 30.7 | 40.8 | - | - |
| SME(BILINEAR) [2] | 526 | 509 | 54.7 | 61.3 | 284 | 158 | 31.3 | 41.3 | - | - |
| LFM [6] | 469 | 456 | 71.4 | 81.6 | 283 | 164 | 26.0 | 33.1 | - | - |
| TransE | **263** | **251** | **75.4** | **89.2** | **243** | **125** | **34.9** | **47.1** | **14,615** | **34.0** |

# Question

- We have two types of relations in KG, for example:
  - Symmetric Relation:
    e.g., (stu1, classmate, stu2), (stu2, classmate, stu1)

  - Composition Relation:
    e.g., (B, husband_of, A), (A, mother_of, C), (B, father_of, C)

## Which Relation can be modeled by TransE? Why?

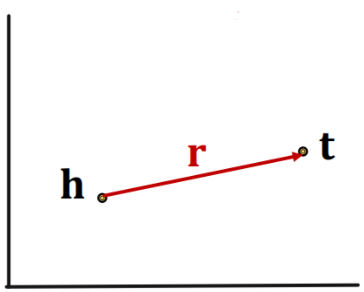
# Question

- **We have two types of relations in KG, for example:**
  - Symmetric Relation:
    e.g., (stu1, classmate, stu2), (stu2, classmate, stu1)

  - Composition Relation:
    e.g., (B, husband_of, A), (A, mother_of, C), (B, father_of, C)

## Which Relation can be modeled by TransE? Why?

TransE cannot model symmetric relations ✗
Only if $\mathbf{r} = 0,\ \mathbf{h} = \mathbf{t}$




For all Symmetric (Antisymmetric) Relations $h$, which means $||h + r - t|| = 0$ and $||t + r - h|| = 0$. Then $r = 0$ and $h = t$, however $h$ and $t$ are two different entities and should be mapped to different locations.

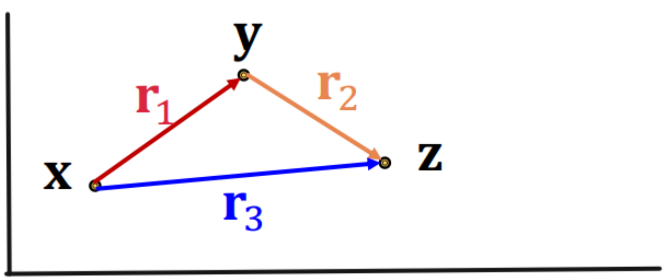
# Question

- **We have two types of relations in KG, for example:**
  - Symmetric Relation:
    e.g., (stu1, classmate, stu2), (stu2, classmate, stu1)

  - Composition Relation:
    e.g., (B, husband_of, A), (A, mother_of, C), (B, father_of, C)

## Which Relation can be modeled by TransE? Why?

TransE can model composition relations ✓
when $\mathbf{r}_3 = \mathbf{r}_1 + \mathbf{r}_2$



# Exercise

- **Can TransE model 1-to-N relations?**
  - e.g., (qiguilin, teacher_of, stu1), (qiguilin, teacher_of, stu2) ,
  (qiguilin, teacher_of, stu3), (qiguilin, teacher_of, stu4)…

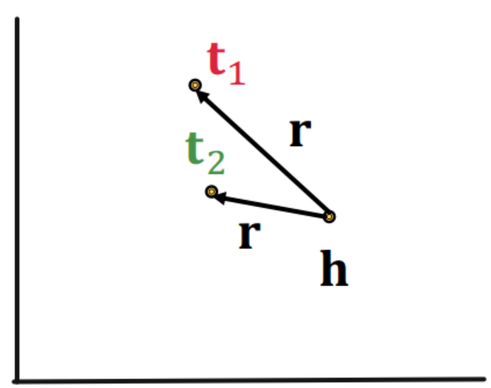
# Exercise

- **Can TransE model 1-to-N relations?**
  - e.g., (qiguilin, teacher_of, stu1), (qiguilin, teacher_of, stu2) ,
    (qiguilin, teacher_of, stu3), (qiguilin, teacher_of, stu4)…

TransE cannot model 1-to-N relations ✗

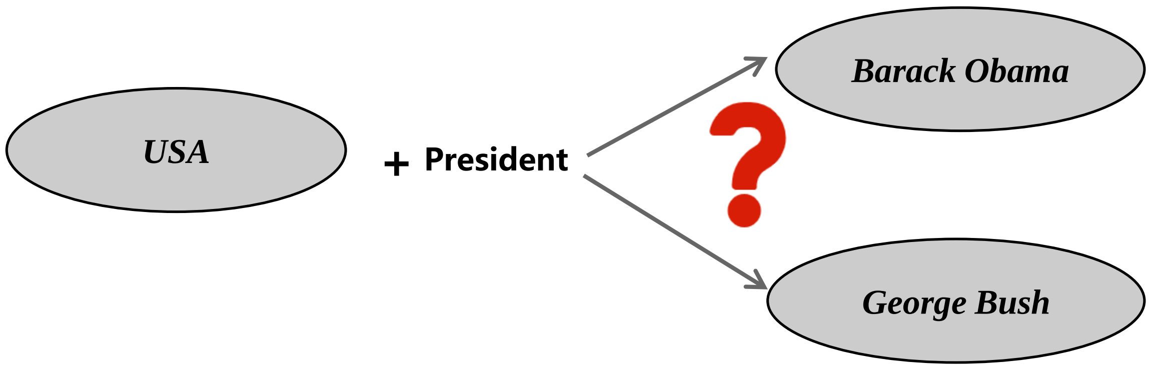
$\mathbf{t}_1$ and $\mathbf{t}_2$ will map to the same vector, although they are different entities

$$\mathbf{t}_1 = \mathbf{h} + \mathbf{r} = \mathbf{t}_2$$
$$\mathbf{t}_1 \neq \mathbf{t}_2$$

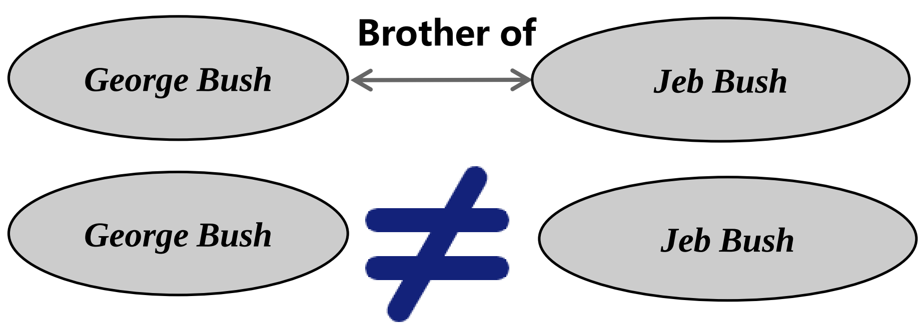


# Issue of TransE

- TransE is too simple to handle complex relations
  - 1-to-N, N-to-1, N-to-N relations

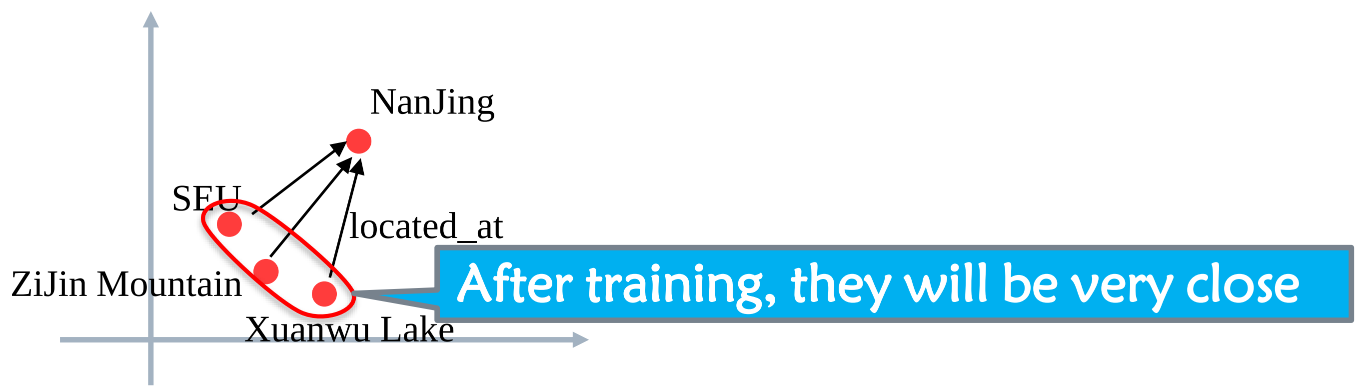



  - symmetric relations



# Variants of TransE: TransH

- **TransH:**
  - Motivation: TransE does not do well in dealing with the relations, such as one-to-many, many-to-one, and many-to-many.




  - TransH proposes:

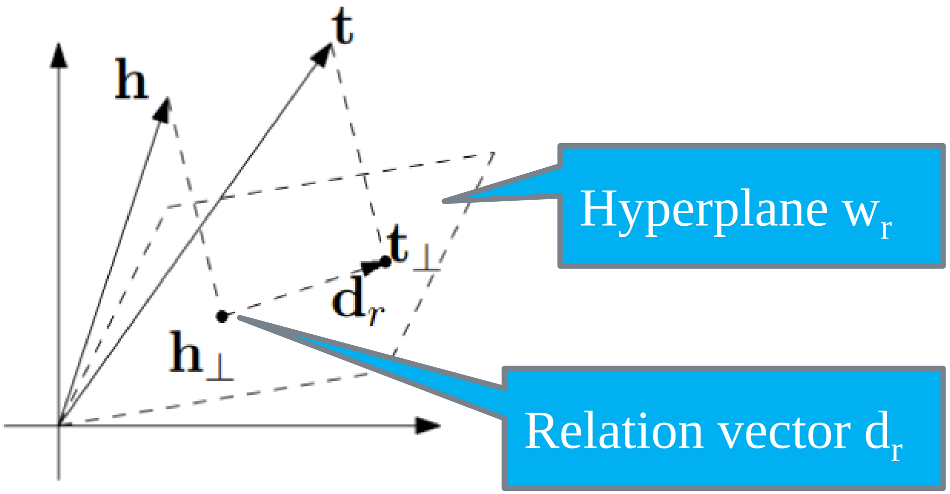
    modeling a relation as a hyperplane together with a translation operation on it.

Wang, et al. (2014). Knowledge graph embedding by translating on hyperplanes, AAAI.

# Variants of TransE: TransH

- **TransH:**
  - For each relation, define a hyperplane $\mathbf{w_r}$ and a relation vector $\mathbf{d_r}$. Then project the head entity vector **h** and the tail entity vector **t** onto the hyperplane $\mathbf{w_r}$.

$$h_\perp = h - w_r^T h w_r$$
$$t_\perp = t - w_r^T t w_r$$

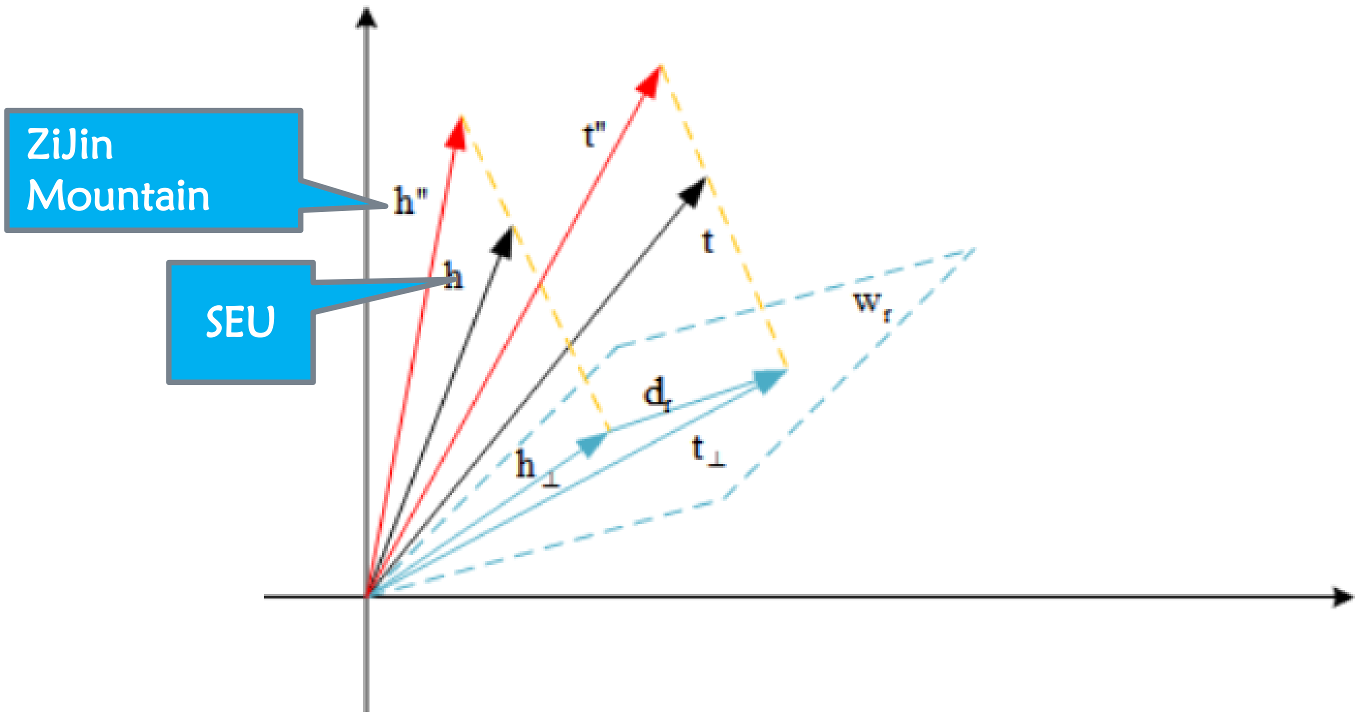



TransH

Wang, et al. (2014). Knowledge graph embedding by translating on hyperplanes, AAAI.

# Variants of TransE: TransH

- **TransH:**
  - For example: in TransE, h and h'' will overlap. While in TransH, entity h and entity h'' will overlap only with the projection h⊥.




Wang, et al. (2014). Knowledge graph embedding by translating on hyperplanes, AAAI.

# Variants of TransE: TransH

- TransH:
  - Hits@10 of TransE and TransH on some examples of one-to-many∗ , many-to-one† , many-to-many‡ , and reflexive § relations

| Relation | Hits@10 (TransE / TransH) Left | Right |
|---|---|---|
| football_position/players* | 100 / 100 | 16.7 / **22.2** |
| production_company/films* | 65.6 / **85.6** | 9.3 / **16.0** |
| director/film* | 75.8 / **89.6** | 50.5 / **80.2** |
| disease/treatments† | 33.3 / **66.6** | 100 / 100 |
| person/place_of_birth† | 30.0 / **37.5** | 72.1 / **87.6** |
| film/production_companies† | 11.3 / **21.0** | 77.6 / **87.8** |
| field_of_study/students_majoring‡ | 24.5 / **66.0** | 28.3 / **62.3** |
| award_winner/awards_won‡ | 40.2 / **87.5** | 42.8 / **86.6** |
| sports_position/players‡ | 28.6 / **100** | 64.3 / **86.2** |
| person/sibling_s§ | 21.1 / **63.2** | 21.1 / **36.8** |
| person/spouse_s§ | 18.5 / **35.2** | 18.5 / **42.6** |

obvious improvement

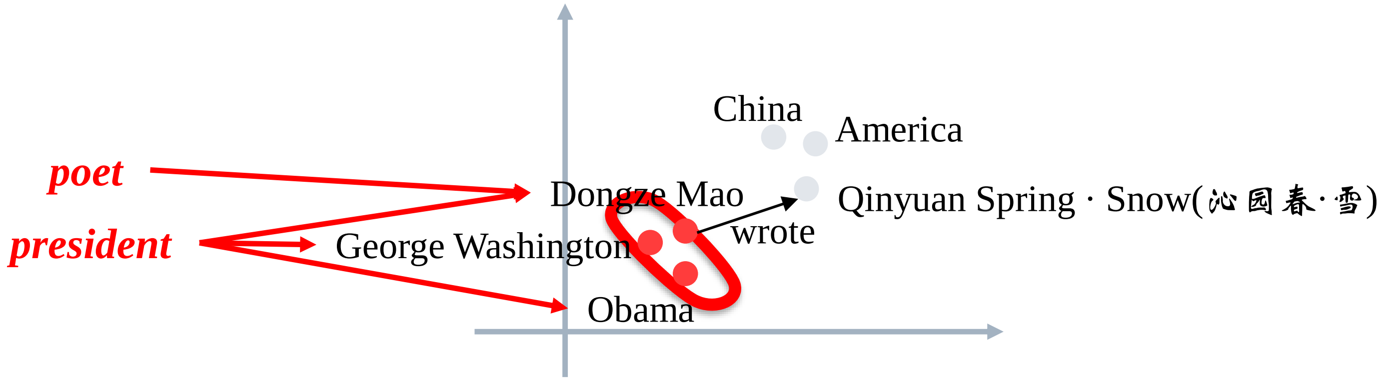
Wang, et al. (2014). Knowledge graph embedding by translating on hyperplanes, AAAI.

# Variants of TransE: TransR

- **TransR:**
  - Both TransE and TransH models assume that entities and relationships are vectors in the same semantic space.

In fact each entity(Dongze Mao, George Washington, Obama) should have many aspects of semantics.

A single vector space is insufficient in modeling semantics.




Lin, et al. (2015). Learning entity and relation embeddings for knowledge graph completion, AAAI.

# Variants of TransE: TransR

- **TransR:**
  - Both TransE and TransH models assume that entities and relationships are vectors in the same semantic space.

    In fact each entity(Dongze Mao, George Washington, Obama) should have many aspects of semantics.

    A single vector space is insufficient in modeling semantics.

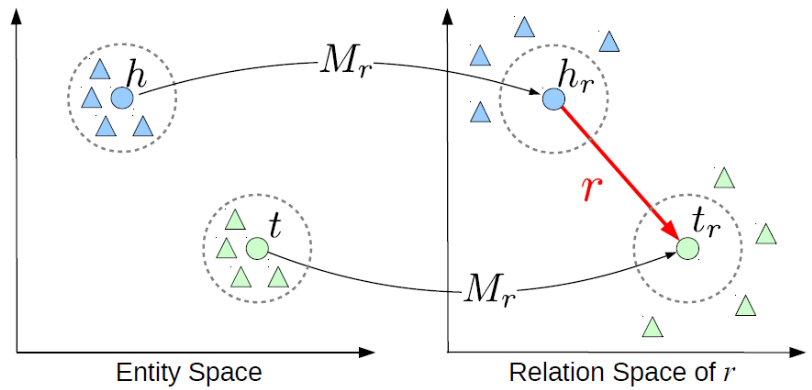
  - TransR proposes:
    - Build entity and relation embeddings in the separate entity space and relation spaces;
    - Then projecting entities from entity space to the corresponding relation space and building translations between projected entities.

Lin, et al. (2015). Learning entity and relation embeddings for knowledge graph completion, AAAI.

# Variants of TransE: TransR

- **TransR:**
  - Mapping entity embeddings into different semantic spaces

$$\mathbf{h}_r = \mathbf{h}\mathbf{M}_r, \quad \mathbf{t}_r = \mathbf{t}\mathbf{M}_r.$$




- The score(energy) function is correspondingly defined as (same as TransE):

$$f_r(h,t) = \|\mathbf{h}_r + \mathbf{r} - \mathbf{t}_r\|_2^2.$$

Lin, et al. (2015). Learning entity and relation embeddings for knowledge graph completion, AAAI.

# Variants of TransE: TransR

- TransR:

| Data Sets | WN18 | | | | FB15K | | | |
|---|---|---|---|---|---|---|---|---|
| Metric | Mean Rank | | Hits@10 (%) | | Mean Rank | | Hits@10 (%) | |
| | Raw | Filter | Raw | Filter | Raw | Filter | Raw | Filter |
| Unstructured (Bordes et al. 2012) | 315 | 304 | 35.3 | 38.2 | 1,074 | 979 | 4.5 | 6.3 |
| RESCAL (Nickel, Tresp, and Kriegel 2011) | 1,180 | 1,163 | 37.2 | 52.8 | 828 | 683 | 28.4 | 44.1 |
| SE (Bordes et al. 2011) | 1,011 | 985 | 68.5 | 80.5 | 273 | 162 | 28.8 | 39.8 |
| SME (linear) (Bordes et al. 2012) | 545 | 533 | 65.1 | 74.1 | 274 | 154 | 30.7 | 40.8 |
| SME (bilinear) (Bordes et al. 2012) | 526 | 509 | 54.7 | 61.3 | 284 | 158 | 31.3 | 41.3 |
| LFM (Jenatton et al. 2012) | 469 | 456 | 71.4 | 81.6 | 283 | 164 | 26.0 | 33.1 |
| TransE (Bordes et al. 2013) | 263 | 251 | 75.4 | 89.2 | 243 | 125 | 34.9 | 47.1 |
| TransH (unif) (Wang et al. 2014) | 318 | 303 | 75.4 | 86.7 | 211 | 84 | 42.5 | 58.5 |
| TransH (bern) (Wang et al. 2014) | 401 | 388 | 73.0 | 82.3 | 212 | 87 | 45.7 | 64.4 |
| TransR (unif) | 232 | 219 | 78.3 | 91.7 | 226 | 78 | 43.8 | 65.5 |
| TransR (bern) | 238 | 225 | **79.8** | 92.0 | **198** | 77 | 48.2 | 68.7 |
| CTransR (unif) | 243 | 230 | 78.9 | **92.3** | 233 | 82 | 44 | 66.3 |
| CTransR (bern) | **231** | **218** | 79.4 | **92.3** | 199 | **75** | **48.4** | **70.2** |

Lin, et al. (2015). Learning entity and relation embeddings for knowledge graph completion, AAAI.

# Summary

- Statistical reasoning uses statistical models to fit the samples and predicts the expected probabilities of the inferred knowledge .
- Knowledge graph embedding based reasoning actually performs entity prediction and relation prediction with vector calculations.
- Translation-based models are now widely used KG embedding models for KG completion and other applications due to its good performance and succinctness.

# Thanks!