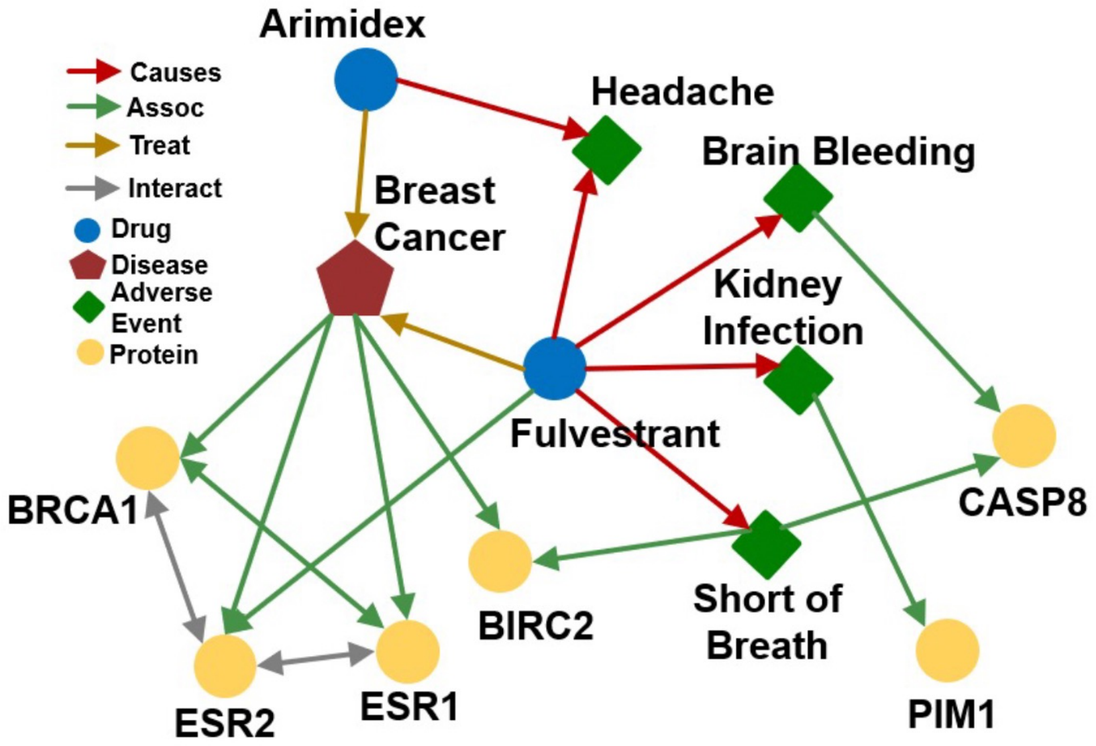
Reasoning over Knowledge Graphs

최예은, 최지우

EWHA EURON

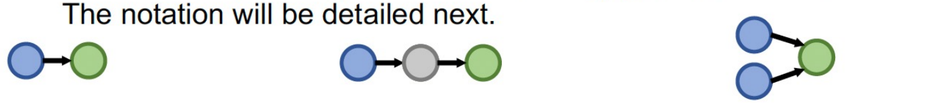
Reasoning in Knowledge Graphs



Can we do multi-hop reasoning, i.e., answer complex queries on an incomplete, massive KG?

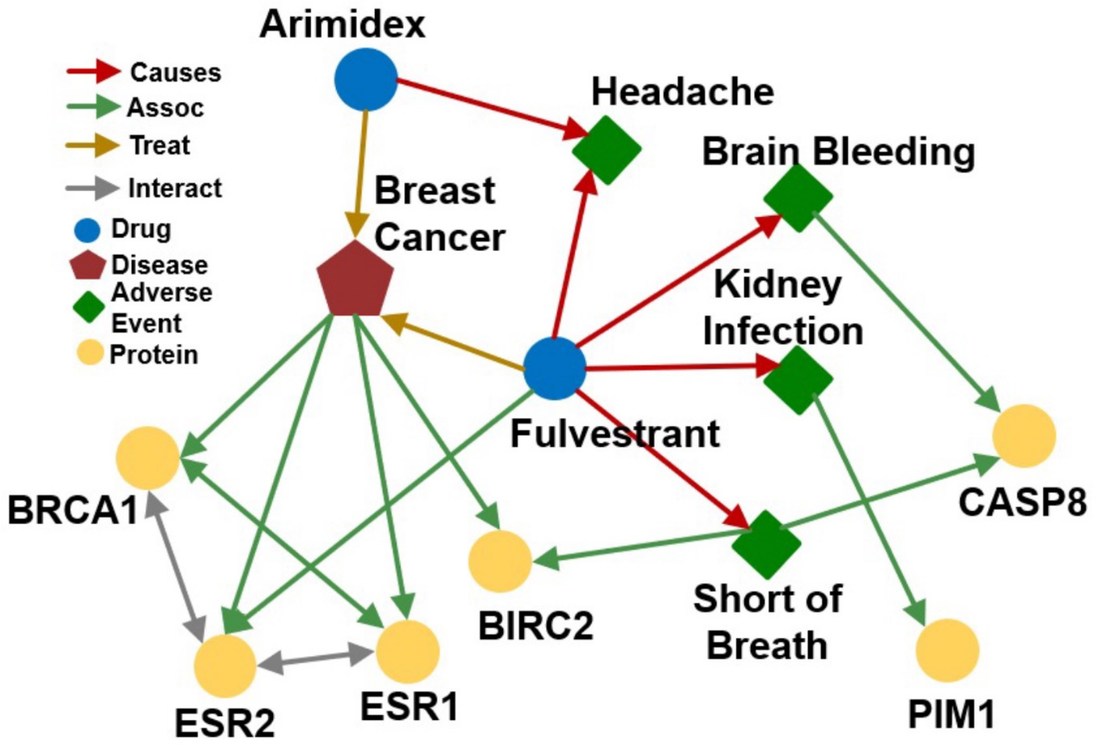
Query Types	Examples: Natural Language Question, Query
One-hop Queries	What adverse event is caused by Fulvestrant? (e:Fulvestrant, (r:Causes))
Path Queries	What protein is associated with the adverse event caused by Fulvestrant? (e:Fulvestrant, (r:Causes, r:Assoc))
Conjunctive Queries	What is the drug that treats breast cancer and caused headache? ((e:BreastCancer, (r:TreatedBy)), (e:Migraine, (r:CausedBy))

In this lecture, we only focus on answering **queries** on a KG!
The notation will be detailed next.



One-hop Queries Path Queries Conjunctive Queries

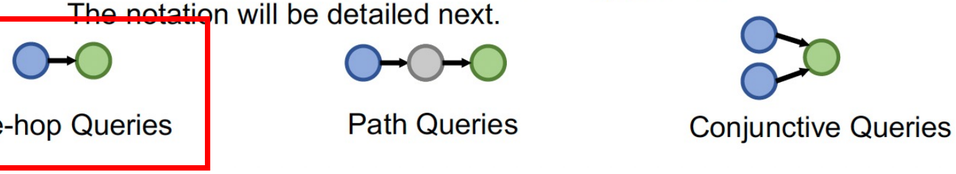
Reasoning in Knowledge Graphs



Can we do multi-hop reasoning, i.e., answer complex queries on an incomplete, massive KG?

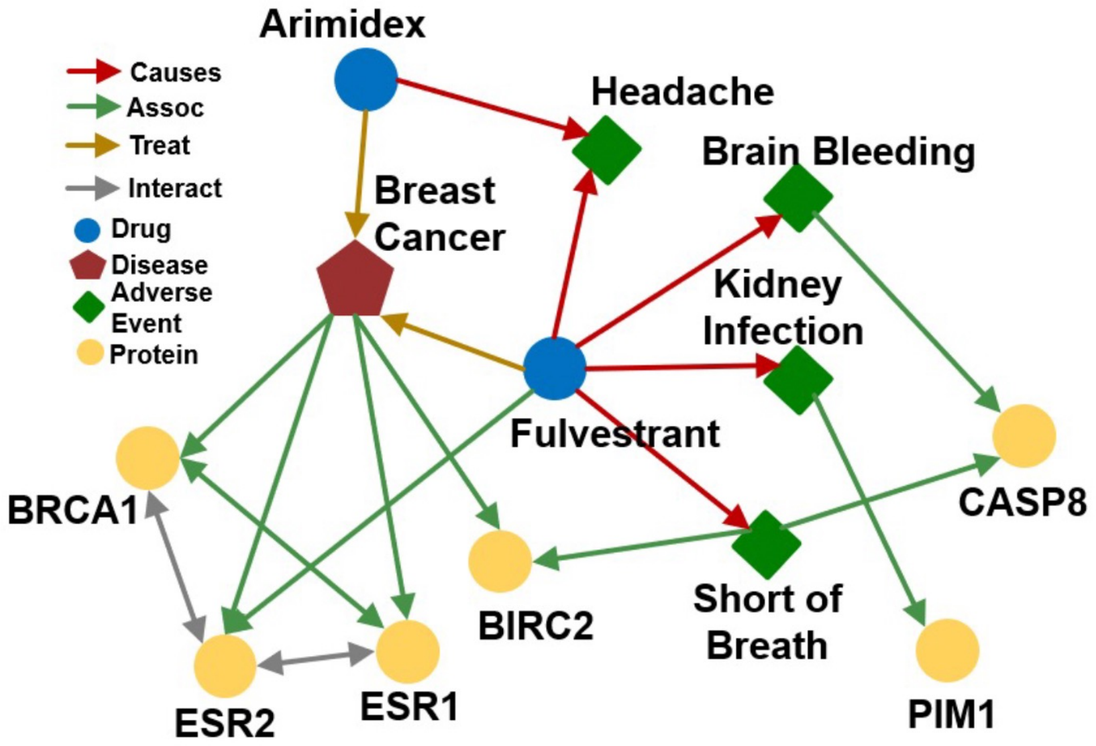
Query Types	Examples: Natural Language Question, Query
One-hop Queries	What adverse event is caused by Fulvestrant? (e:Fulvestrant, (r:Causes))
Path Queries	What protein is associated with the adverse event caused by Fulvestrant? (e:Fulvestrant, (r:Causes, r:Assoc))
Conjunctive Queries	What is the drug that treats breast cancer and caused headache? ((e:BreastCancer, (r:TreatedBy)), (e:Migraine, (r:CausedBy))

In this lecture, we only focus on answering **queries** on a KG!
The notation will be detailed next.



KG completion과 동일한 task로 head와 relation이 주어졌을 때 tail을 예측함.

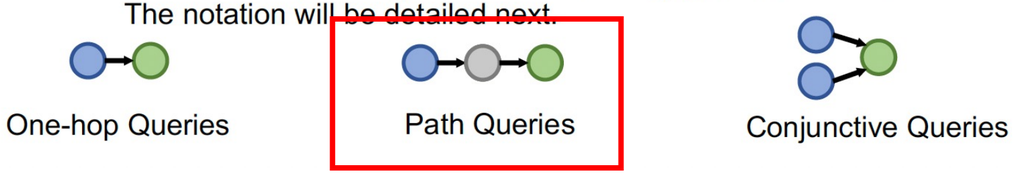
Reasoning in Knowledge Graphs



Can we do multi-hop reasoning, i.e., answer complex queries on an incomplete, massive KG?

Query Types	Examples: Natural Language Question, Query
One-hop Queries	What adverse event is caused by Fulvestrant? (e:Fulvestrant, (r:Causes))
Path Queries	What protein is associated with the adverse event caused by Fulvestrant? (e:Fulvestrant, (r:Causes, r:Assoc))
Conjunctive Queries	What is the drug that treats breast cancer and caused headache? ((e:BreastCancer, (r:TreatedBy)), (e:Migraine, (r:CausedBy))

In this lecture, we only focus on answering **queries** on a KG!
The notation will be detailed next.

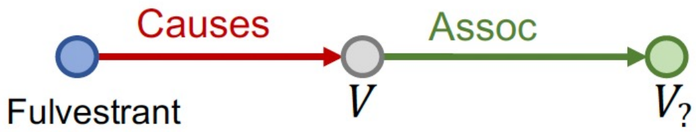


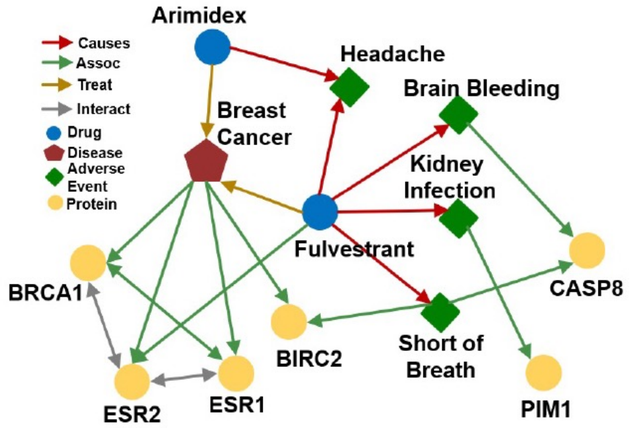
출발하는 노드(anchor) *v_a*로부터 **n개의 관계를 거쳐 도달하는 노드에 대해 예측**한다.

Reasoning in Knowledge Graphs

- Path Queries

  - v_a is e:Fulvestrant
  - (r_1, r_2) is (r:Causes, r:Assoc)
  - **Query:** (e:Fulvestrant, (r:Causes, r:Assoc))



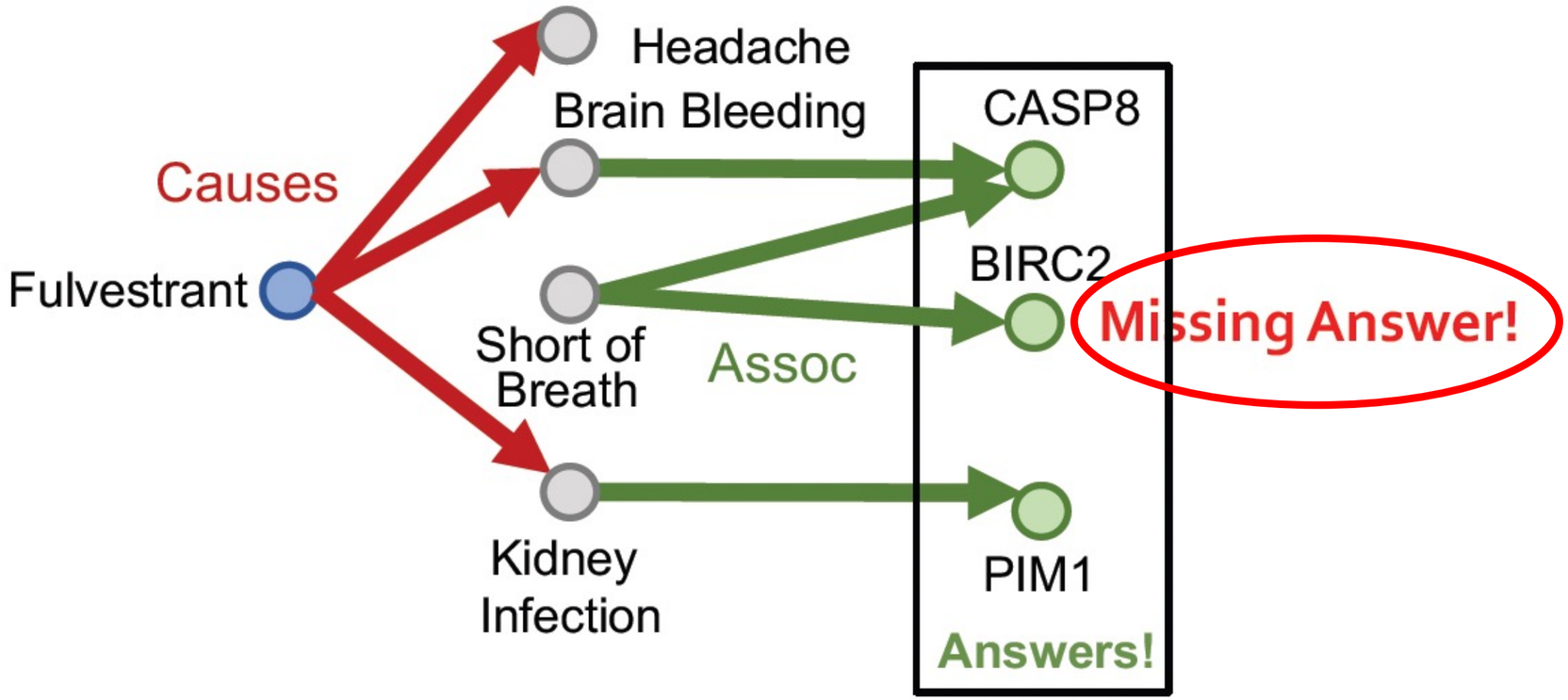


"풀베스트란트에 의해 야기된 부작용과 관련된 단백질은 무엇인가?"

Reasoning in Knowledge Graphs

- Path Queries

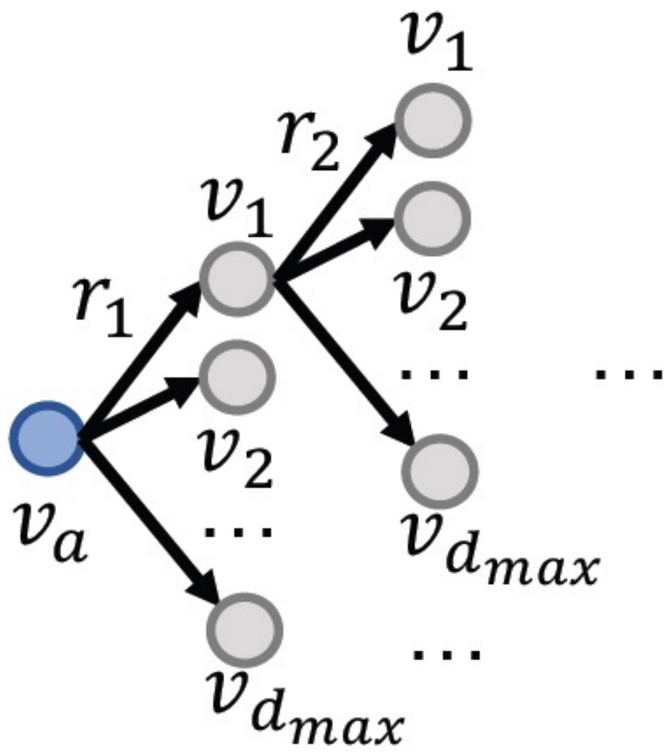
"풀베스트란트에 의해 야기된 부작용과 관련된 단백질은 무엇인가?"



KGs는 불완전한 경우가 많아 위와 같이 **엣지가 빠져있을 경우 정답을 못 찾을 수 있다.**

Reasoning in Knowledge Graphs

- Can KG Completion Help? ✗



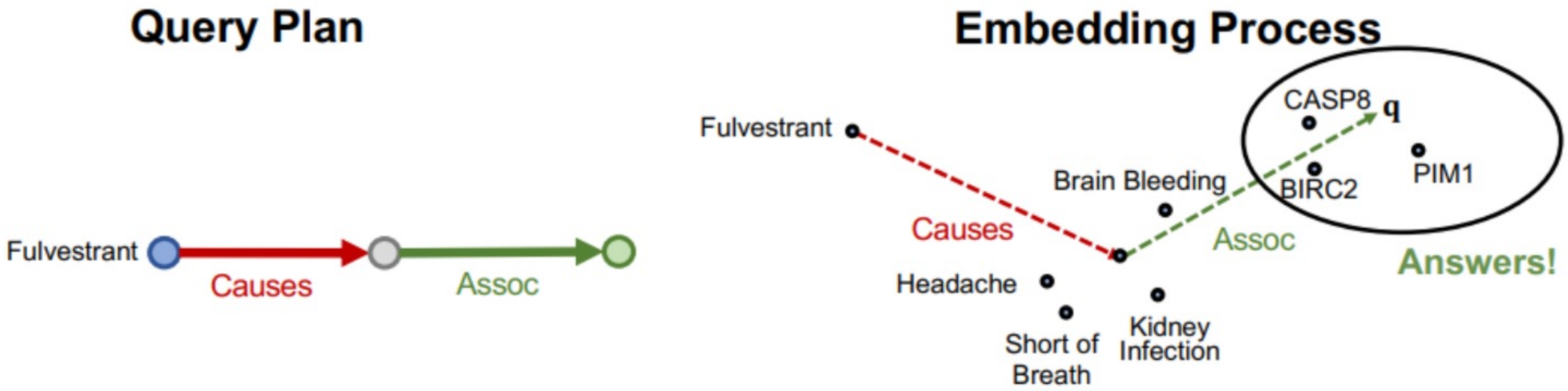
KG completion을 통해 그래프를 완성시키고 탐색하는 방법이 있지만 Length L에 대해 시간복잡도가 $O(d^L)$로 너무 크다는 단점이 있다.

$$O(d_{max}^L)$$

predictive queries를 수행해서 해결!

Traversing KG in Vector Space

- Key idea: Embed queries

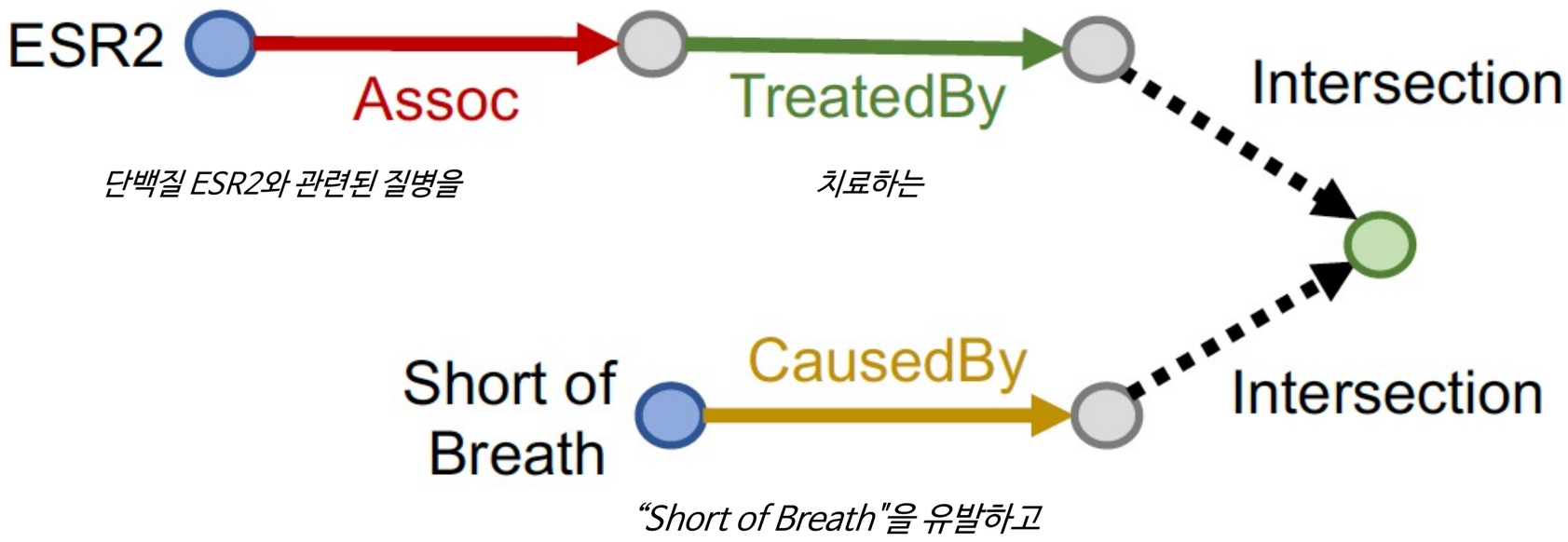


$$q = v_a + r_1 + \cdots r_n$$

•지난시간에 배운 TransE를 활용한다면 **임베딩을 통해 합성 관계를 표현**할 수 있다.
•n-hop을 거친 후의 벡터와 가까운 entity 임베딩 벡터를 예측값으로 삼는다.

More complex queries- Conjunctive Queries

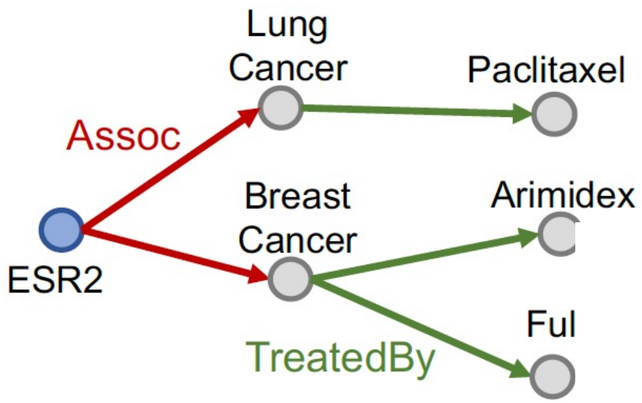
Conjunctive Queries :
“Short of Breath"을 유발하고 단백질 ESR2와 관련된 질병을 치료하는 약물은 무엇입니까?

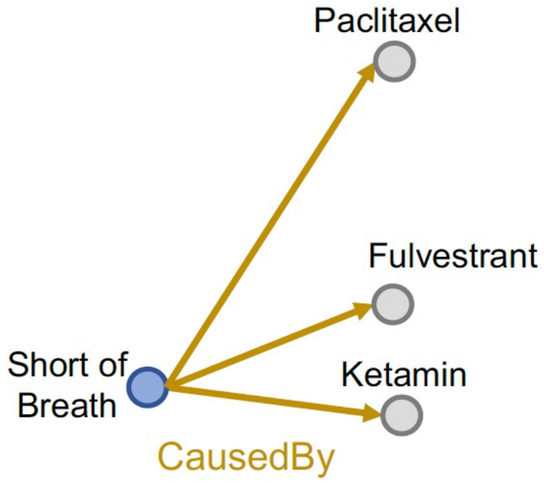


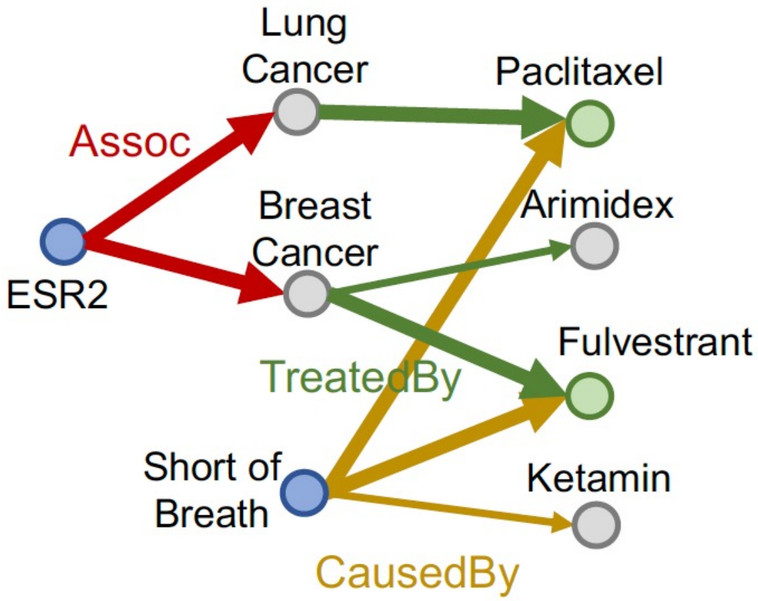
((e:ESR2, (r:Assoc, r:TreatedBy)), (e:Short of Breath, (r:CausedBy))

More complex queries- Conjunctive Queries

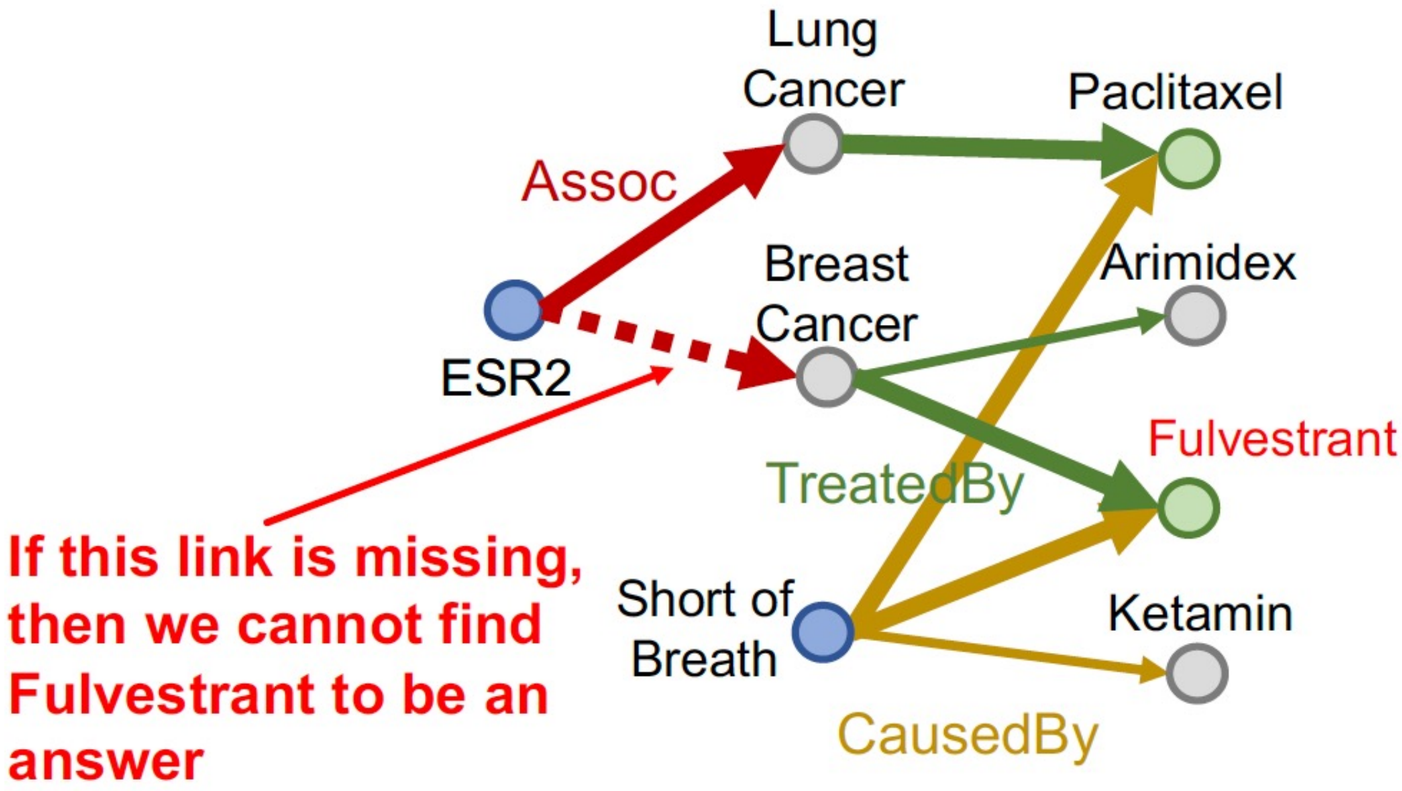
Conjunctive Queries :
“Short of Breath"을 유발하고 단백질 ESR2와 관련된 질병을 치료하는 약물은 무엇입니까?





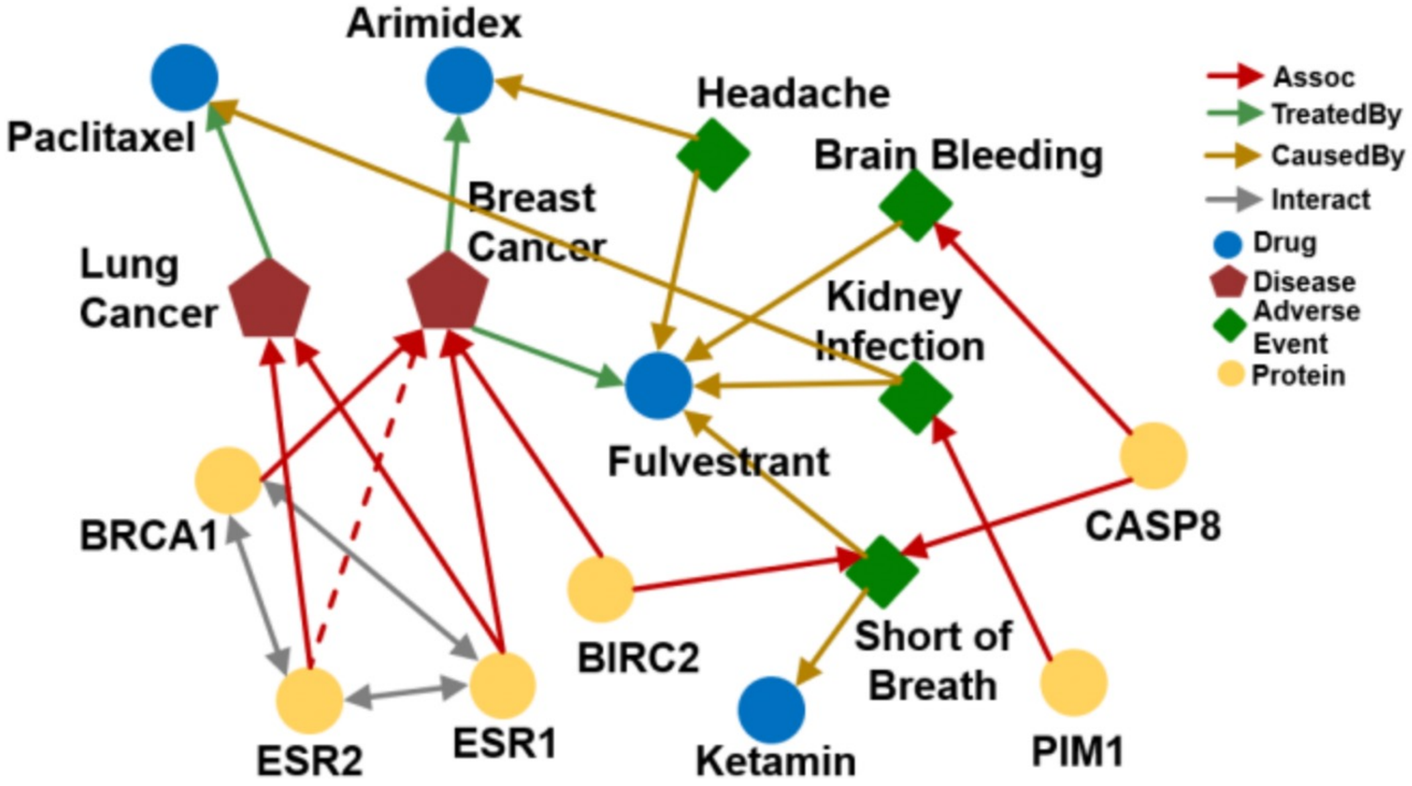


More complex queries- Conjunctive Queries



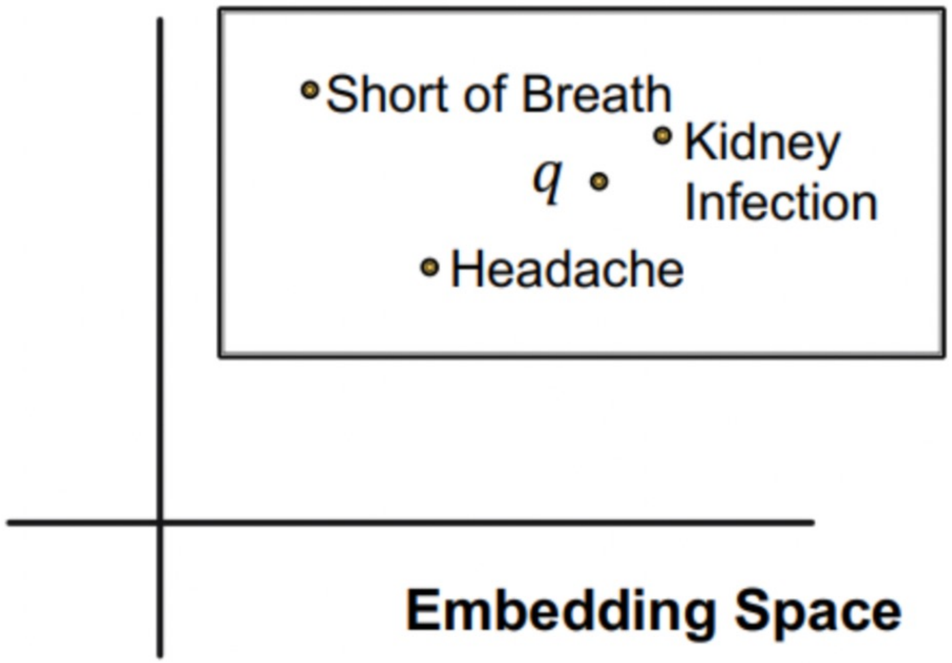
하지만, **그래프는 완전하지 않기 때문에** ESR2-〉Breast Cancer 엣지가 없을 수 있으며
이럴 경우 Fulvestrant는 예측하지 못하게 된다.

More complex queries- Conjunctive Queries



이 때, Breast Cancer은 BRCA1을 거쳐 ESR2와 연결되어 있기 때문에 둘의 **연결관계를 모델이 내재적으로 인지할 수 있어야 한다.**

Box Embeddings



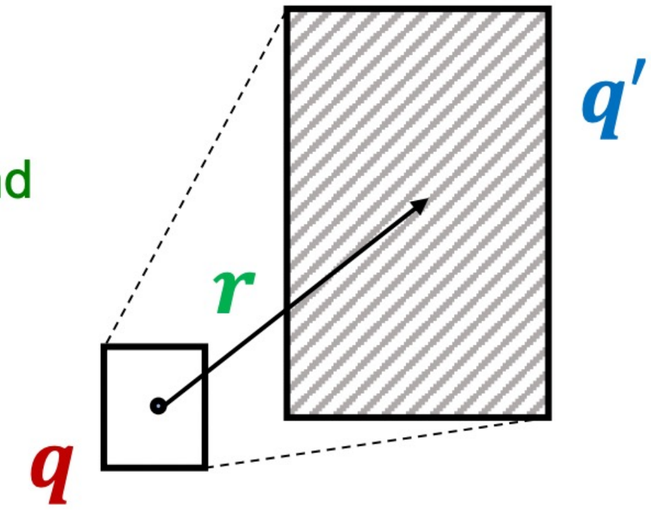
- 임베딩 공간에서 query(entity+relation)는 **박스 형태로 표현**할 수 있다.
- **Query의 tail이 박스 내에 위치**하며 q=(Center(q),offset(q))이다.
- 박스를 활용할 경우 **Conjunctive queries의 정답**을 두 anchor로부터 나오는 예측 tail set들의 **교집합으로 쉽게 표현**할 수 있다.
- Entity embedding는 크기가 0인 박스로 취급하며 파라미터 수는 d|V|이다. d는 out degree, |V|는 entities의 수를 의미한다.
- Relation embedding의 파라미터 수는 2d|R|이며 R은 relations의 수를 의미한다.
- f는 두 박스를 받아 교집합 박스를 output으로 내주는 함수이다.

Projection Operator

- $\mathcal{P}$: Box × Relation → Box

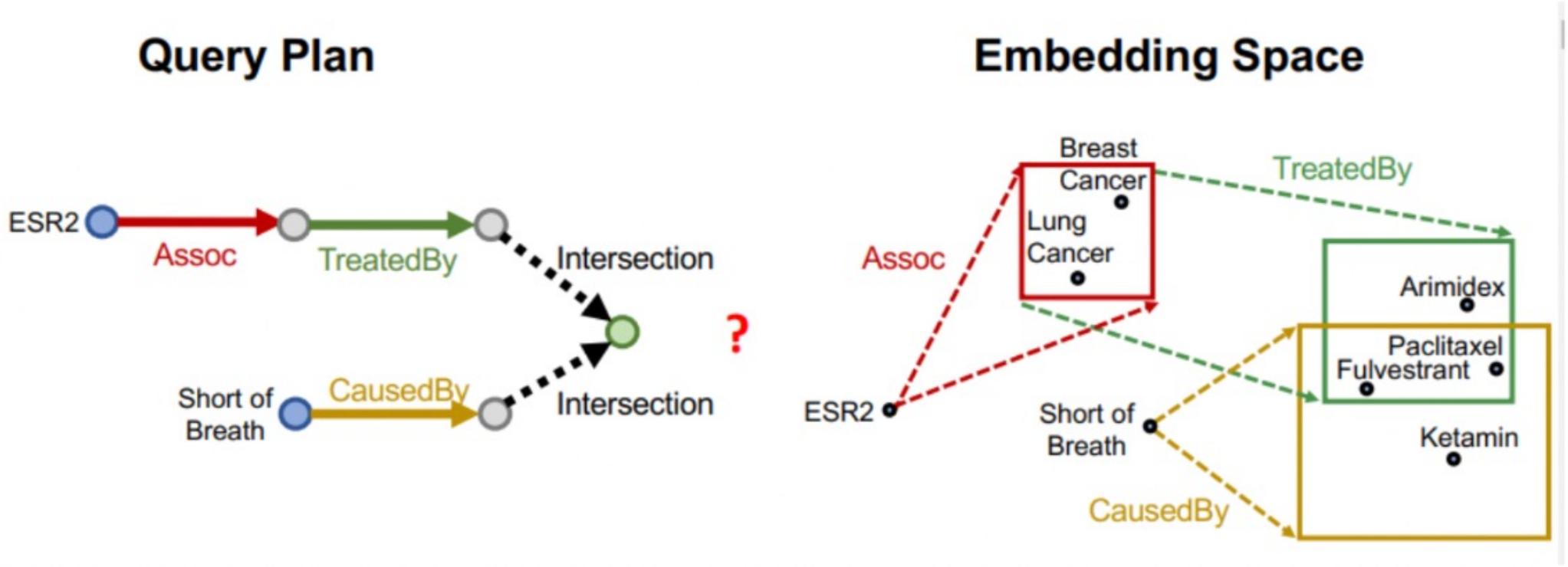
$$Cen(q') = Cen(q) + Cen(r)$$
$$Off(q') = Off(q) + Off(r)$$

"×" (cross) means the projection operator is a **relation** from any box and **relation** to a new box



- Projection Operator P는 **box와 relation을 입력으로 받아 box의 중심과 크기를 변형시킨다.**
- 기존 박스에 대한 벡터 q와 relation 벡터를 선형결합한다.

Embed with Box Embedding



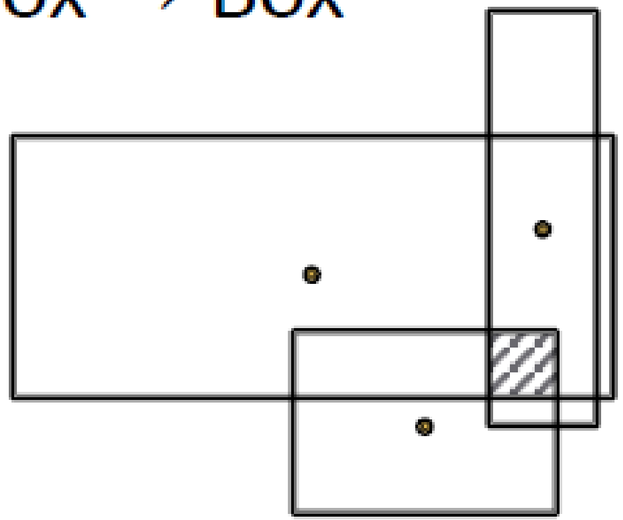
- Anchor 노드의 임베딩 벡터는 크기가 0인 박스로 표현된다.
- q=P(*ESR2*,*Assoc*)을 통해 tail에 해당하는 노드 집합을 담는 center와 offset을 갖는 박스 임베딩이 형성된다.
- q' =P(q, *TreatedBy*)를 통해 새로운 박스 임베딩이 만들어진다.
- 또 다른 anchor인 Short of Breath에서 시작하는 tail에 대한 박스 임베딩도 q=P(*ShortofBreath*,*CausedBy*)를 통해 만들어준다.
- 최종적으로 만들어진 **초록 박스와 노랑 박스의 교집합**을 구한다.

Intersection Operator

- 후보 노드들 간에 어떤 노드가 교집합이 생기는지 알고싶음

1. input box의 center point와 새로운 box의 center point는 가까워야 한다.
2. 새로운 box의 크기는 줄어들어야 함! 입력값으로 사용된 박스 중 가장 작은 박스보다 작아야 한다.

$$\mathcal{J} : \text{Box} \times \cdots \times \text{Box} \rightarrow \text{Box}$$



- Geometric Intersection Operator J을 도입

Intersection Operator

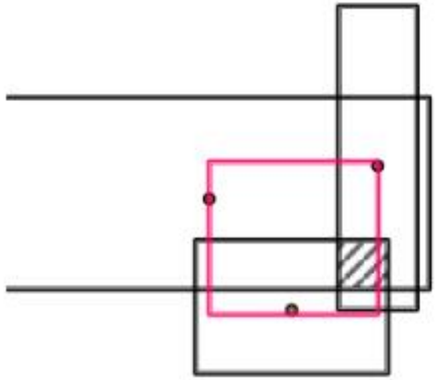
- $\mathcal{I}$: Box $\times \cdots \times$ Box $\rightarrow$ Box

Hadamard product
(element-wise produ

$$Cen(q_{inter}) = \sum_i \boldsymbol{w}_i \odot Cen(q_i)$$

$$\boldsymbol{w}_i = \frac{\exp(f_{cen}(Cen(q_i)))}{\sum_j \exp(f_{cen}(Cen(q_j)))}$$

$Cen(q_i) \in \mathbb{R}^d$
$\boldsymbol{w}_i \in \mathbb{R}^d$



Intuition: The center should be in the red region!
Implementation: The center is a **weighted sum** of the input box centers

$\boldsymbol{w}_i \in \mathbb{R}^d$ is calculated by a neural network f_{cen} (with trainable weights)
$\boldsymbol{w}_i$ **represents a "self-attention" score for the center of each input** $Cen(q_i)$.

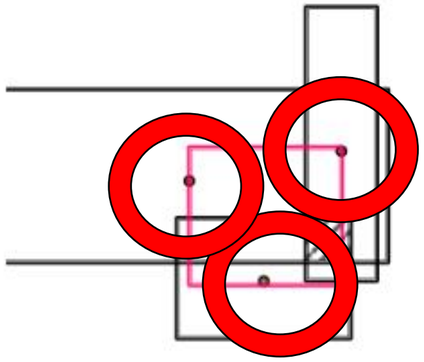
Intersection Operator-Center

- $\mathcal{I}$: Box $\times \cdots \times$ Box $\rightarrow$ Box

Hadamard product (element-wise produ

$$Cen(q_{inter}) = \sum_i \boldsymbol{w}_i \odot Cen(q_i)$$

$$\boldsymbol{w}_i = \frac{\exp(f_{cen}(Cen(q_i)))}{\sum_j \exp(f_{cen}(Cen(q_j)))} \qquad \begin{matrix} Cen(q_i) \in \mathbb{R}^d \\ \boldsymbol{w}_i \in \mathbb{R}^d \end{matrix}$$



Intuition: The center should be in the red region!
Implementation: The center is a **weighted sum** of the input box centers

$\boldsymbol{w}_i \in \mathbb{R}^d$ is calculated by a neural network f_{cen} (with trainable weights)
$\boldsymbol{w}_i$ **represents a "self-attention" score for the center of each input** $Cen(q_i)$.

1. 각 박스의 중심 벡터가 임의의 함수 fcen를 통과.
2. 소프트함수를 통과 후 가중치 벡터 wi가 됨.
3. 가중치 벡터 wi와 중심 벡터를 Hadamard 곱하여 가중합한다.
4. 중심들과 근접한 점을 새로운 박스의 중심으로 갱신한다.

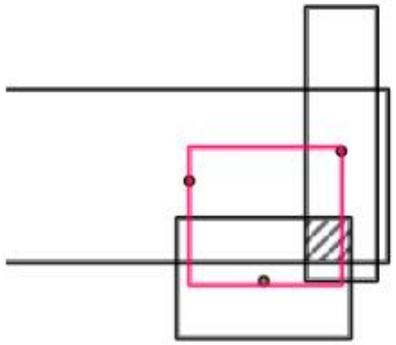
Intersection Operator-Center

- $\mathcal{I}$: Box × ⋯ × Box → Box

Hadamard product (element-wise produ

$$Cen(q_{inter}) = \sum_i \boldsymbol{w}_i \odot Cen(q_i)$$

$$\boldsymbol{w}_i = \frac{\exp(f_{cen}(Cen(q_i)))}{\sum_j \exp(f_{cen}(Cen(q_j)))} \qquad Cen(q_i) \in \mathbb{R}^d \quad \boldsymbol{w}_i \in \mathbb{R}^d$$



Intuition: The center should be in the red region!
Implementation: The center is a **weighted sum** of the input box centers

$\boldsymbol{w}_i \in \mathbb{R}^d$ is calculated by a neural network f_{cen} (with trainable weights)

$\boldsymbol{w}_i$ represents a "self-attention" score for the center of each input $Cen(q_i)$.

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_\mathbf{K}}})\mathbf{V}$$

1. 각 박스의 중심 벡터가 임의의 함수 fcen를 통과.
2. 소프트함수를 통과 후 가중치 벡터 wi가 됨.
3. 가중치 벡터 wi와 중심 벡터를 Hadamard 곱하여 가중합한다.
4. 중심들과 근접한 점을 새로운 박스의 중심으로 갱신한다.

+transformer 에서 쓰이는 self attention 구조를 띄우는 것을 확인할 수 있다.

Intersection Operator-Offset

Geometric Intersection Operator $\mathcal{I}$

- $\mathcal{I}$: Box $\times \cdots \times$ Box $\rightarrow$ Box

$$Off(q_{inter}) = \min\big(Off(q_1), \ldots, Off(q_n)\big) \odot \sigma(f_{off}(Off(q_1), \ldots, Off(q_n)))$$

guarantees shrinking

Center, Offset 에서 쓰이는 foff, fcen은 단순하게 겹치는 교집합을 추출하는 것 보다 더 효과적으로 representation을 구하여 모델의 표현력을 높이기 위해 도입하였다.

Intersection Operator-Offset

Geometric Intersection Operator $\mathcal{I}$

- $\mathcal{I}$: Box $\times \cdots \times$ Box $\rightarrow$ Box

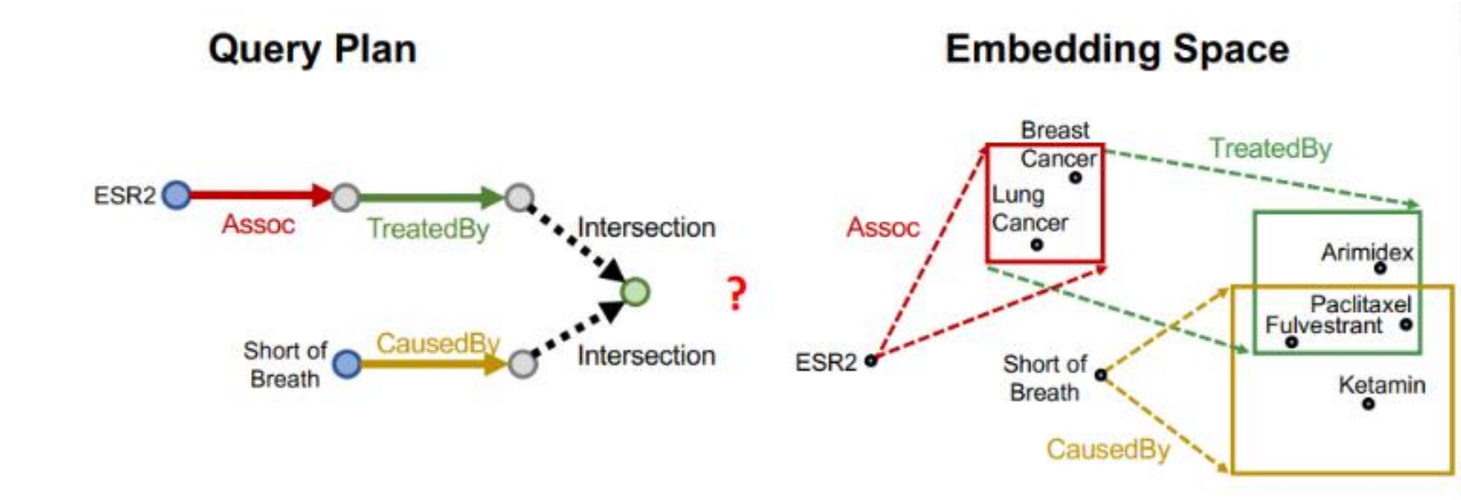
$$Off(q_{inter}) = \min(Off(q_1), \ldots, Off(q_n)) \odot \sigma(f_{off}(Off(q_1), \ldots, Off(q_n)))$$

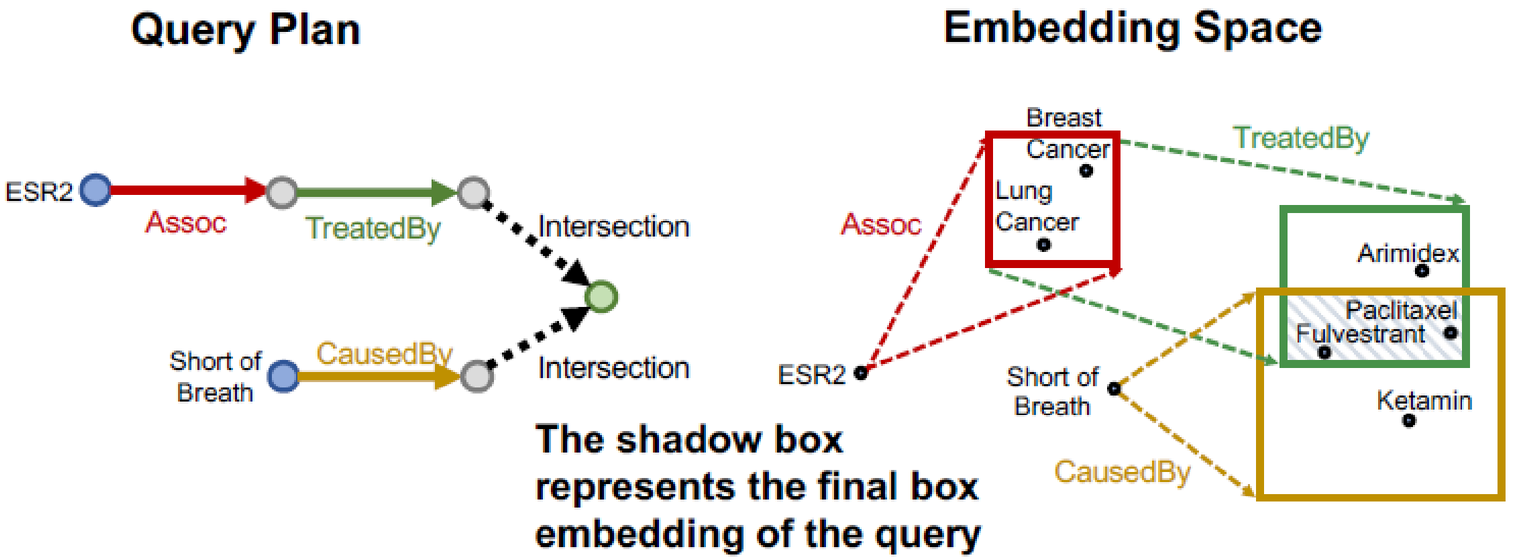
guarantees shrinking

Foff를 통과한 벡터에 대해 시그모이드 함수를 취해주고 있음. Off항은 input box의 크기 벡터를 입력으로 하는데 min()함수를 취해 가장 작은 박스를 골라서 아다마르 곱을 취한다.

→ 새로운 박스의 크기는 가장 작은 박스의 크기보다 더 element wise하게 축소시키게 된다.

Intersection Operator





Entity to Box Distance

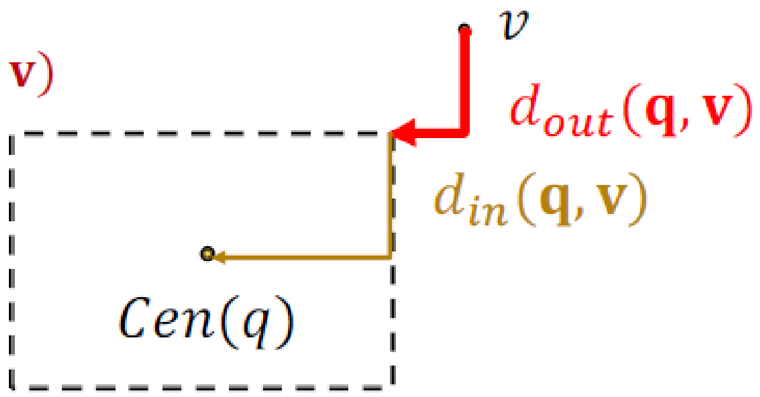
데이터가 이상적으로 깔끔 x, 노이즈가 껴있을 가능성이 높다.
Query2Box에서는 박스 내부의 노드만 결과 노드로 사용하는 것이 아니라 박스에 가까운 노드에 대해서도 결과 노드로 사용하고 있음.

Q. 어떻게 박스로부터 각 노드까지의 거리를 구해서 결과 노드를 결정할까?

A. 직관적으로 박스의 중심에 가까워야 하므로 중심으로부터 노드까지의 거리를 재면 된다.

Intuition: if the point is enclosed in the box, the distance should be **downweighted**.

$$f_q(v) = -d_{box}(\mathbf{q}, \mathbf{v})$$



Given a query box $\mathbf{q}$ and entity embedding (box) $\mathbf{v}$,

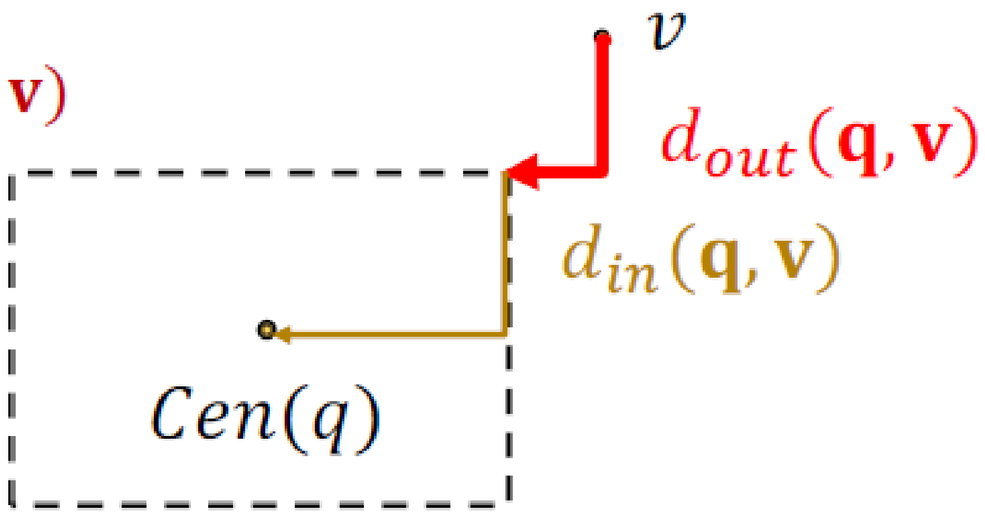
$$d_{box}(\mathbf{q}, \mathbf{v}) = d_{out}(\mathbf{q}, \mathbf{v}) + \alpha \cdot d_{in}(\mathbf{q}, \mathbf{v})$$

where $0 < \alpha < 1$.

Entity to Box Distance

Intuition: if the point is enclosed in the box, the distance should be **downweighted**.

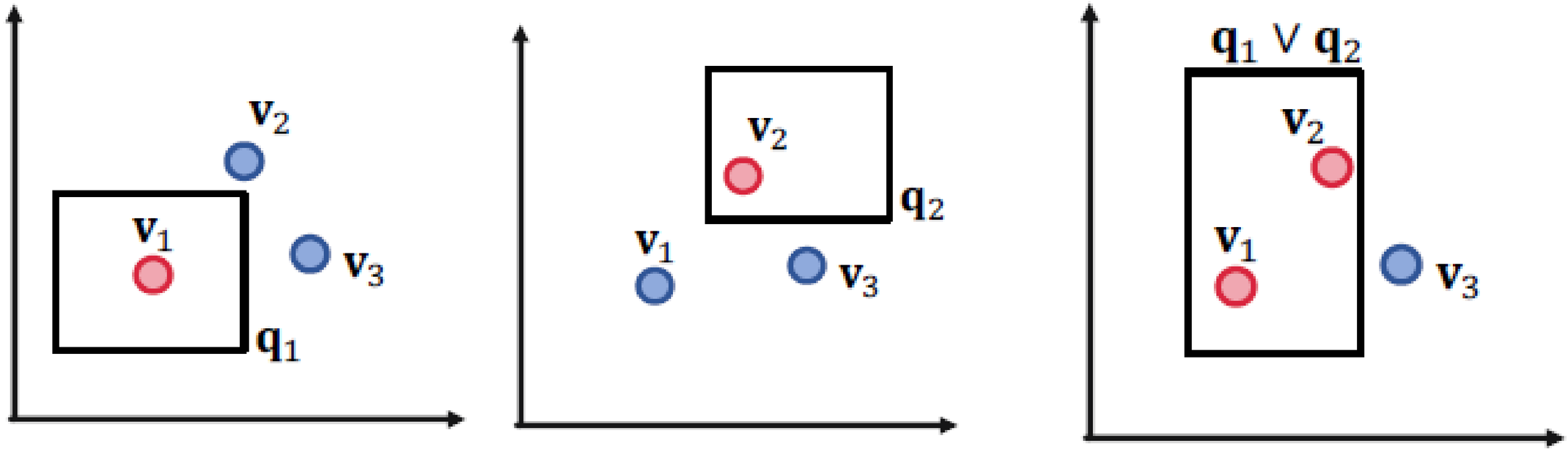
$$f_q(v) = -d_{box}(\mathbf{q}, \mathbf{v})$$



노드 V에 대해 박스 경계면까지의 거리는 그대로 사용.
박스 경계면 내부의 거리는 가중치를 주어 작게 측정하게 함.
(박스 내부에 있다면 어느 정도 올바르게 선택된 노드이기 때문에.)

And-OR Query

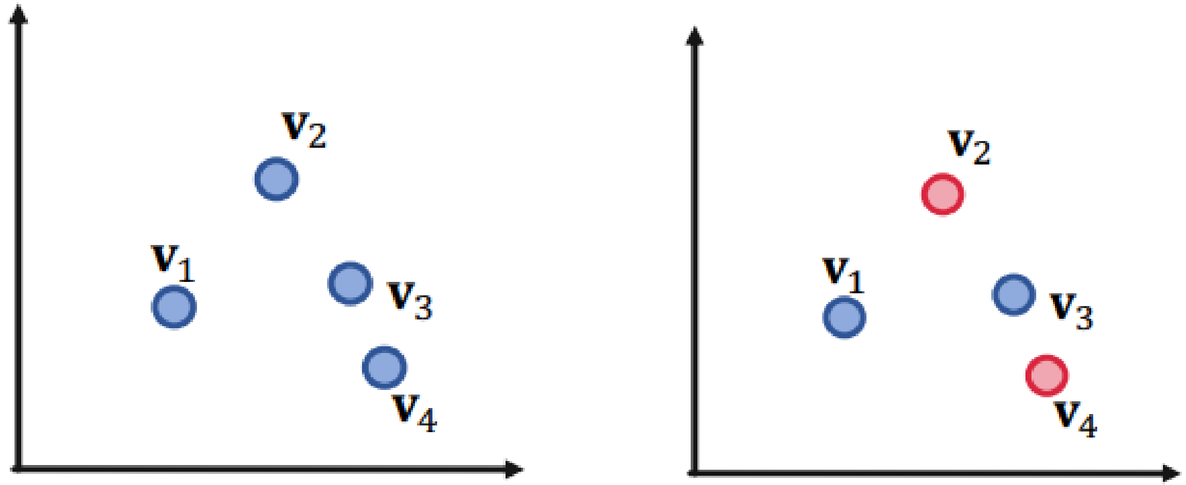
AND-OR 연산은 저차원이 아닌 고차원 임베딩 공간에서 가능하다.



정답 노드를 갖고 있는 쿼리들 사이의 or 연산은 box를 그려서 구할 수 있다.

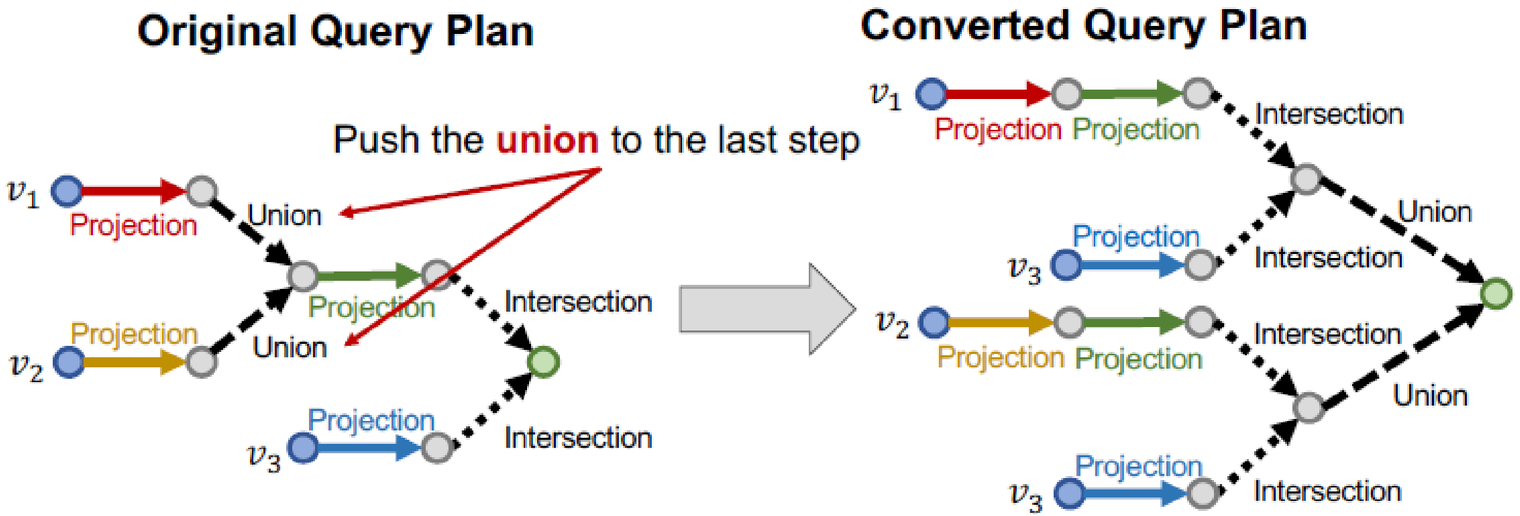
And-OR Query

AND-OR 연산은 저차원이 아닌 고차원 임베딩 공간에서 가능하다.



데이터가 늘어나는 경우 정답 노드끼리 쿼리 박스를 구하기가 어려워진다. 2차원에서는 절대로 불가능한 연산.

And-OR Query



OR 연산은 연산량이 많아서 AND와 OR의 순서를 바꿔 연산량을 줄인다.

(a∪b)∩c=(a∩c)∪(a∩b) 로 변경할 수 있으며 a∪b하기는 어렵지만 a∩c과 a∩b는 연산할 수 있다. AND연산을 먼저 처리한다.

And-OR Query

AND 연산 후 최종적으로 구한 박스와 실제 정답 노드 간의 거리를 구해서 거리를 최소화하는 함수를 이용하기 때문에! OR 연산을 실제로 하진 않는다.

$$d_{box}(\mathbf{q}, \mathbf{v}) = min(d_{box}(\mathbf{q}_1, \mathbf{v}), \ldots, d_{box}(\mathbf{q}_m, \mathbf{v}))$$

AND연산을 모두 수행하고 OR 연산만 남은 상태에서

OR연산의 각각의 항에 대해 정답 노드와 가장 거리가 짧은 박스와의 거리가 최종 박스와의 거리가 될 것이므로,

각 항의 박스와의 거리가 최소인 값을 최종 거리로 선택하게 된다.

Embedding any AND-OR query q

query2box의 AND-OR 연산의 전체적인 과정

1. Transform q to **equivalent DNF** $q_1 \vee \cdots \vee q_m$
2. **Embed** q_1 to q_m
3. Calculate the (box) distance $d_{box}(\mathbf{q}_i, \mathbf{v})$
4. Take the **minimum** of all distance
5. **The final score** $f_q(v) = -d_{box}(\mathbf{q}, \mathbf{v})$

Training

정답 노드 v, 오답 노드 v’ 를 이용해 f(v)를 최대화하고 f(v’)를 최소화하는 것이 목표

- Trainable parameters:
  - Entity embeddings with $d|V|$ # params
  - Relation embeddings with $2d|R|$ # params
  - Intersection operator

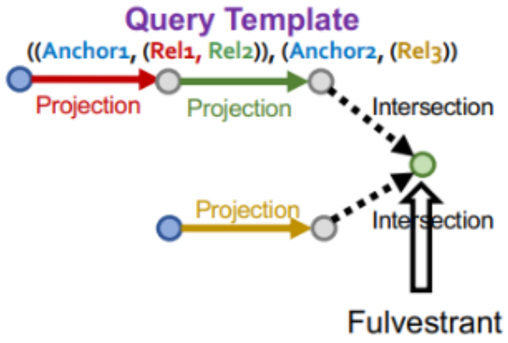
Parameter를 학습하기 위한Query는 어떻게 만들까?
Query answering를 위한 KG는 어떻게 split할까?

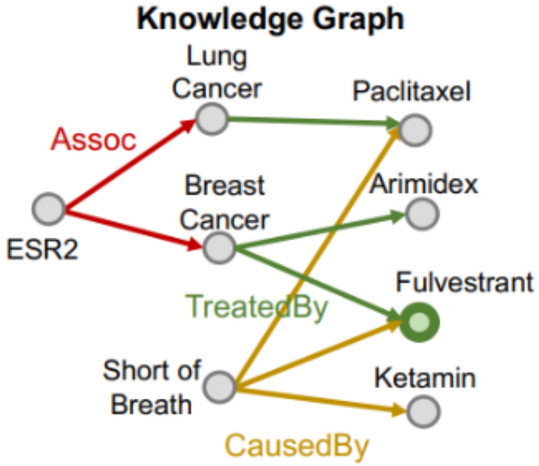
Training

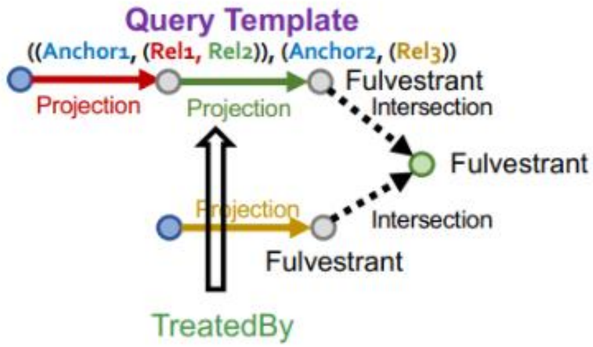
- **Training**:
  1. Randomly sample a query q from the training graph G_{train}, answer $v \in [\![q]\!]_{G_{train}}$, and a negative sample $v' \notin [\![q]\!]_{G_{train}}$.
     - Negative sample: Entity of same type as v but not answer.
  2. Embed the query $\mathbf{q}$.
  3. Calculate the score $f_q(v)$ and $f_q(v')$.
  4. Optimize the loss ℓ to maximize $f_q(v)$ while minimize $f_q(v')$:

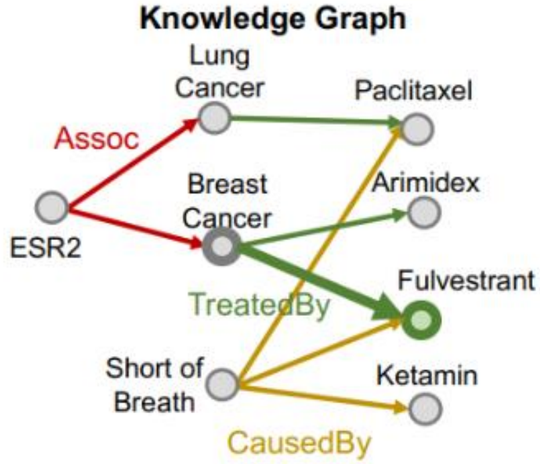
$$\ell = -\log \sigma\left(f_q(v)\right) - \log(1 - \sigma(f_q(v')))$$

Training



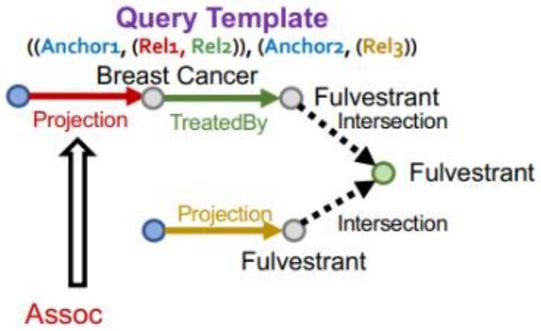


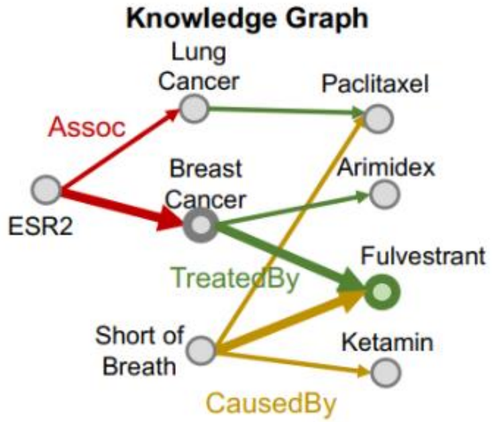


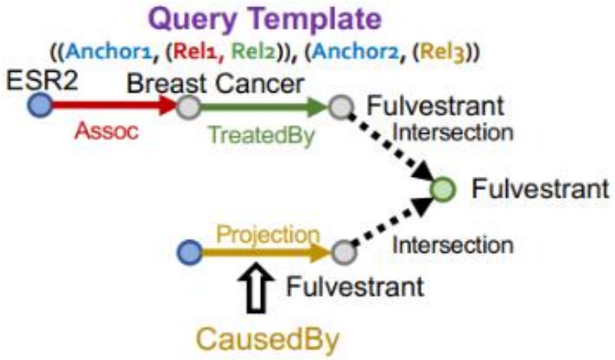


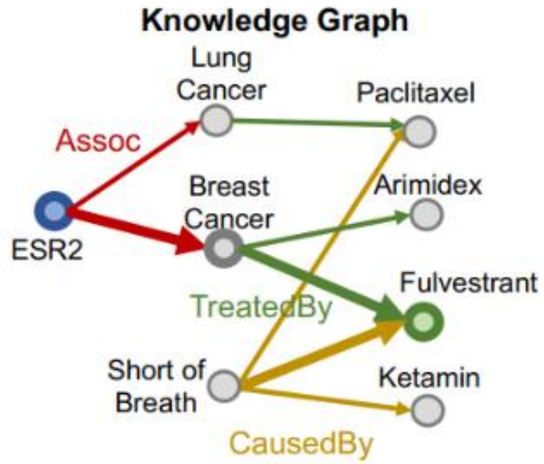
1. 랜덤하게 지식 그래프에서 하나의 노드를 루트 노드로 선택한다.
2. 랜덤하게 루트 노드와 연결된 intersection을 선택한다. 그것을 따라 relation을 기록한다.

Training



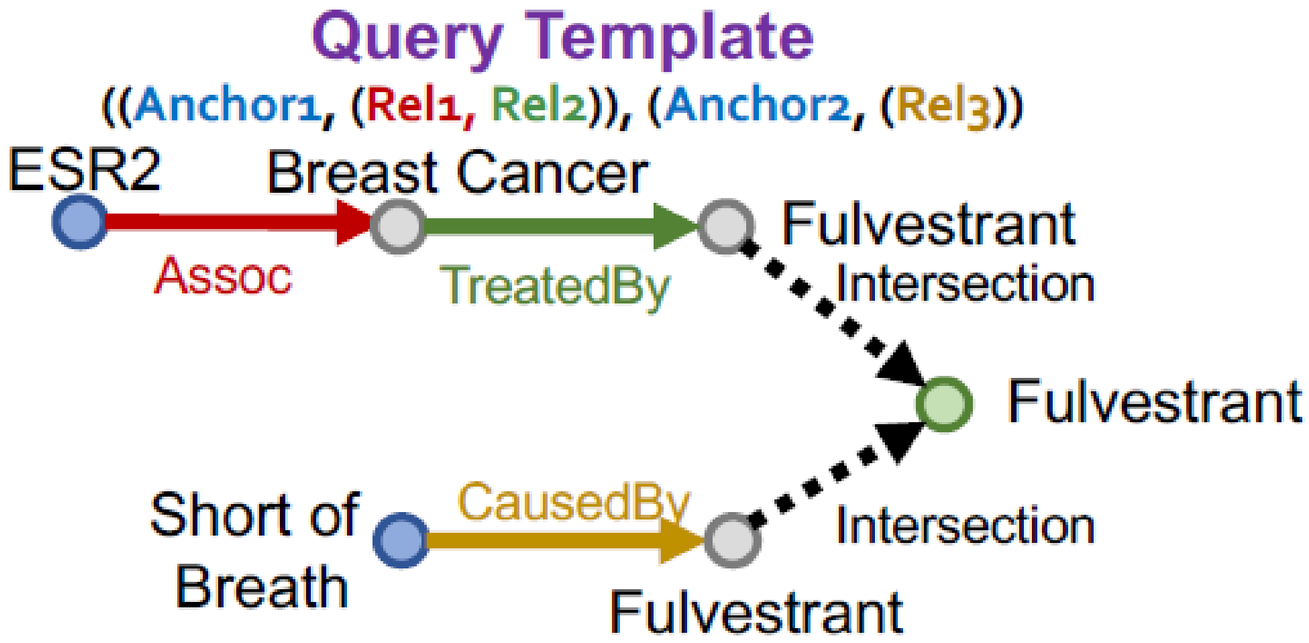






3. 더 이상 거슬러 올라갈 노드가 없을 때까지 relation을 거슬러 올라간다.
4. 2에서 선택하지 않았던 intersection에 대해 마저 relation을 구상한다.

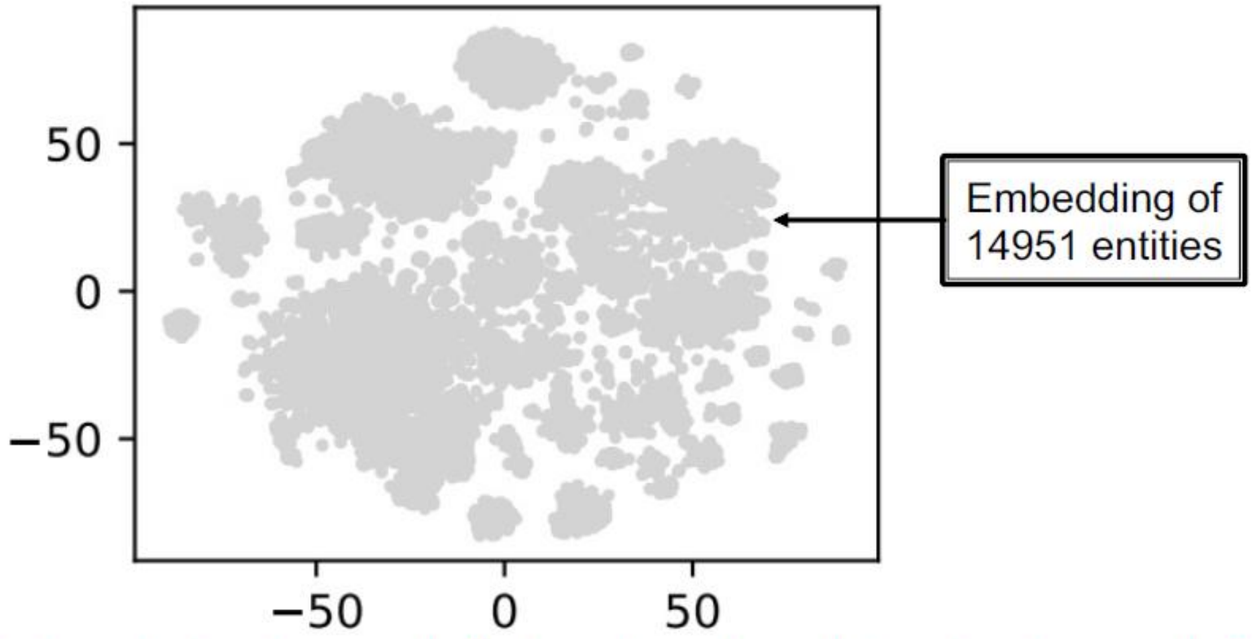
Training



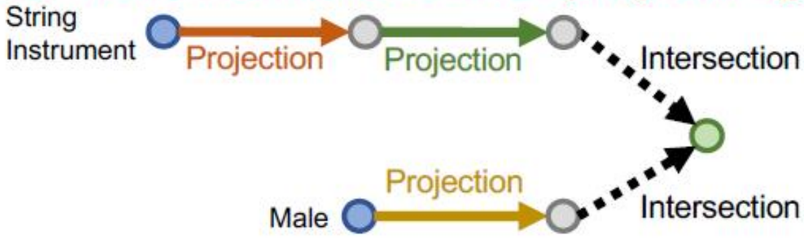
5. 전체 query를 표현한 query template를 가질 수 있다. 전체 정답 집합과 오답 노드 또한 구할 수 있게 된다.

Visualization

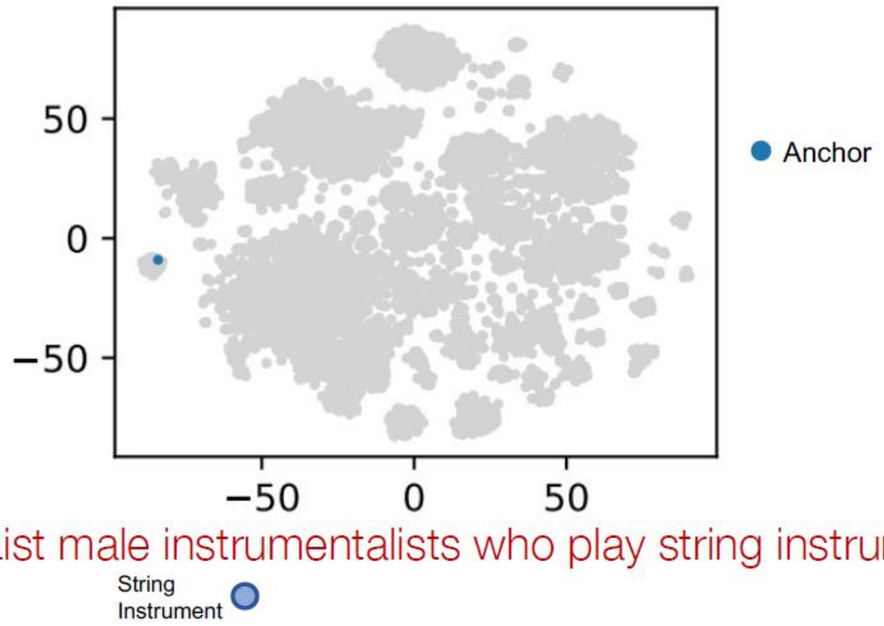
$$((e : \text{Instrument}, (r : \text{WhichIs}, r : \text{PlayedBy}), (e : \text{Male}, (r : \text{Occupying}))$$

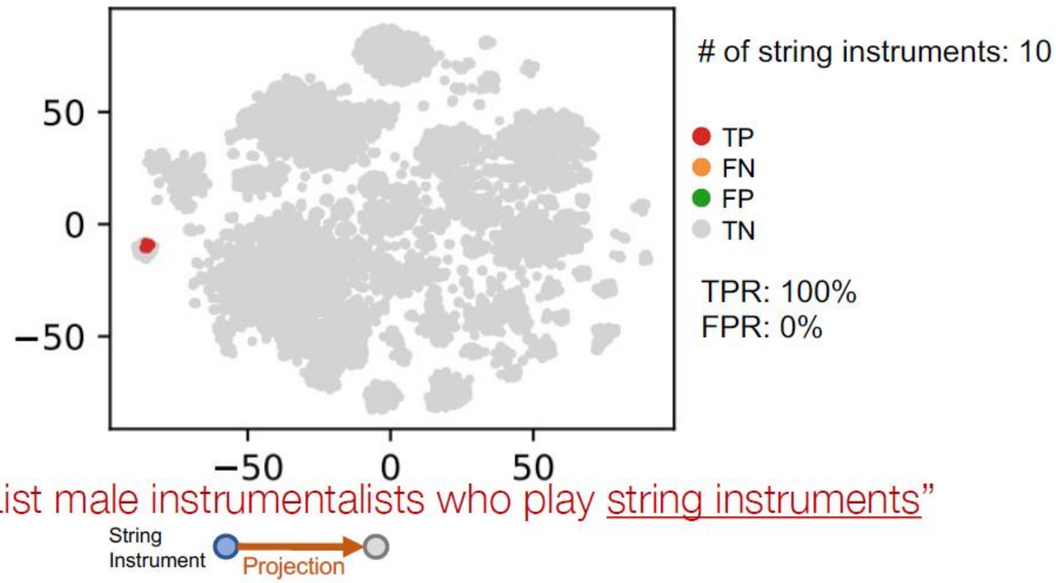


"List male instrumentalists who play string instruments"



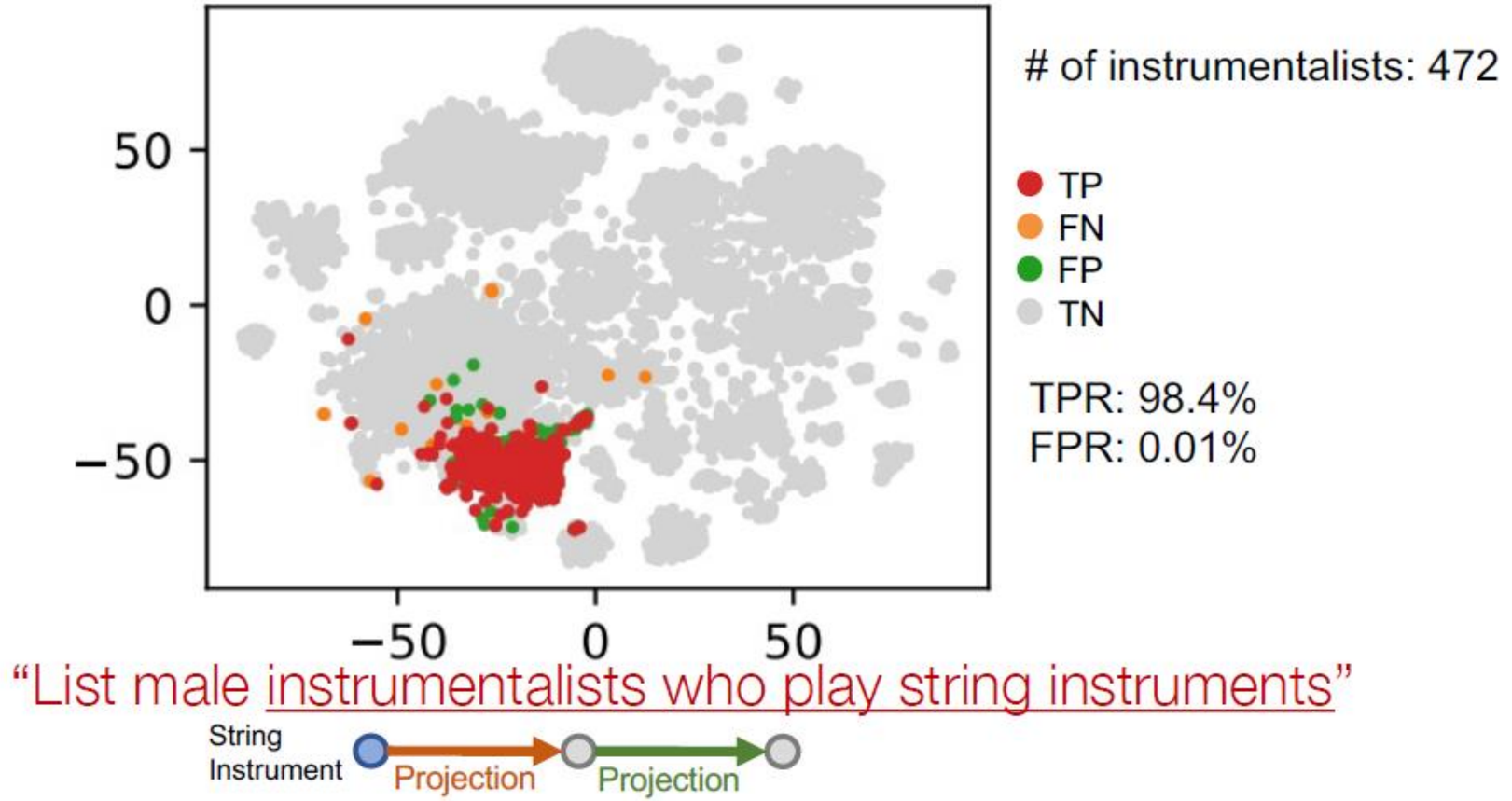
Visualization





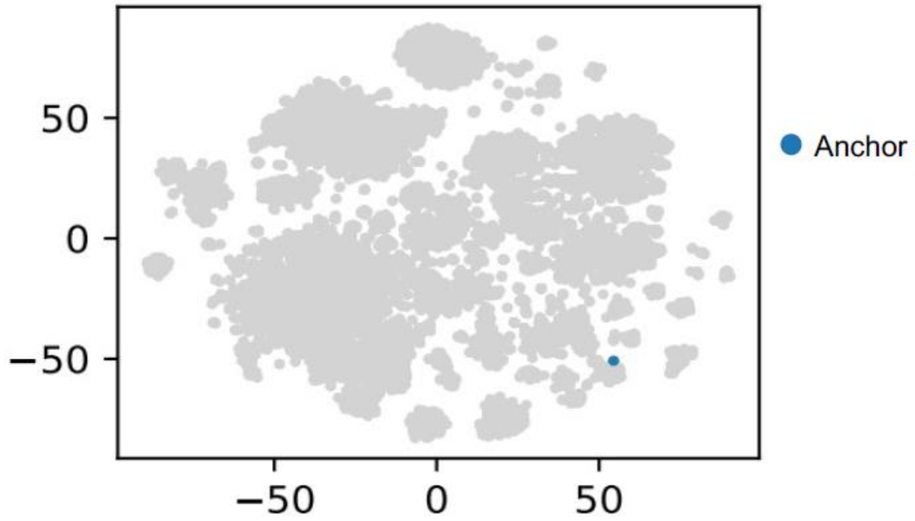
첫번째 anchor 노드에서 시작하여 첫번째 relation으로 이동.

Visualization

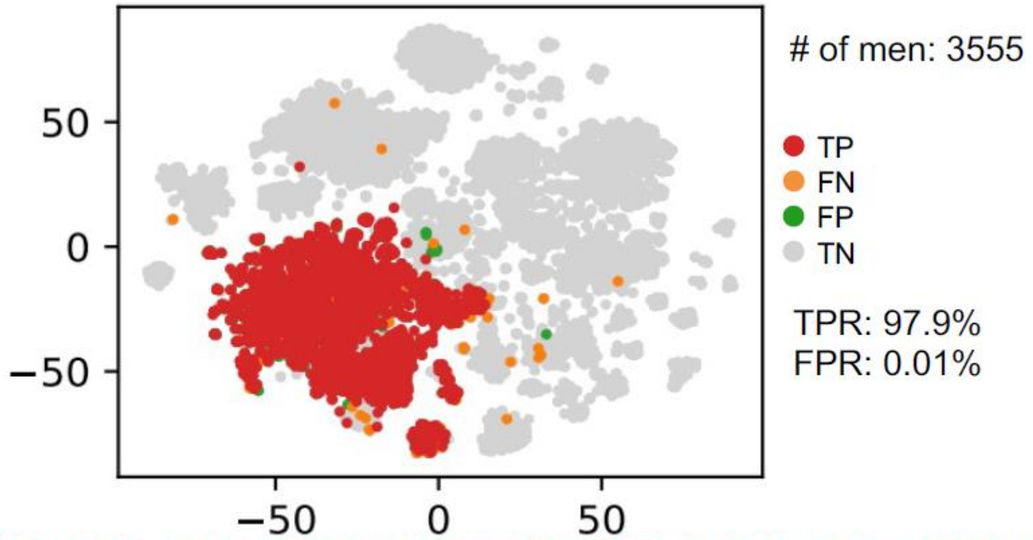


두번째 relation으로 이동. 박스에 오차가 생기면서 TPR이 낮아진 것을 확인할 수 있다.

Visualization



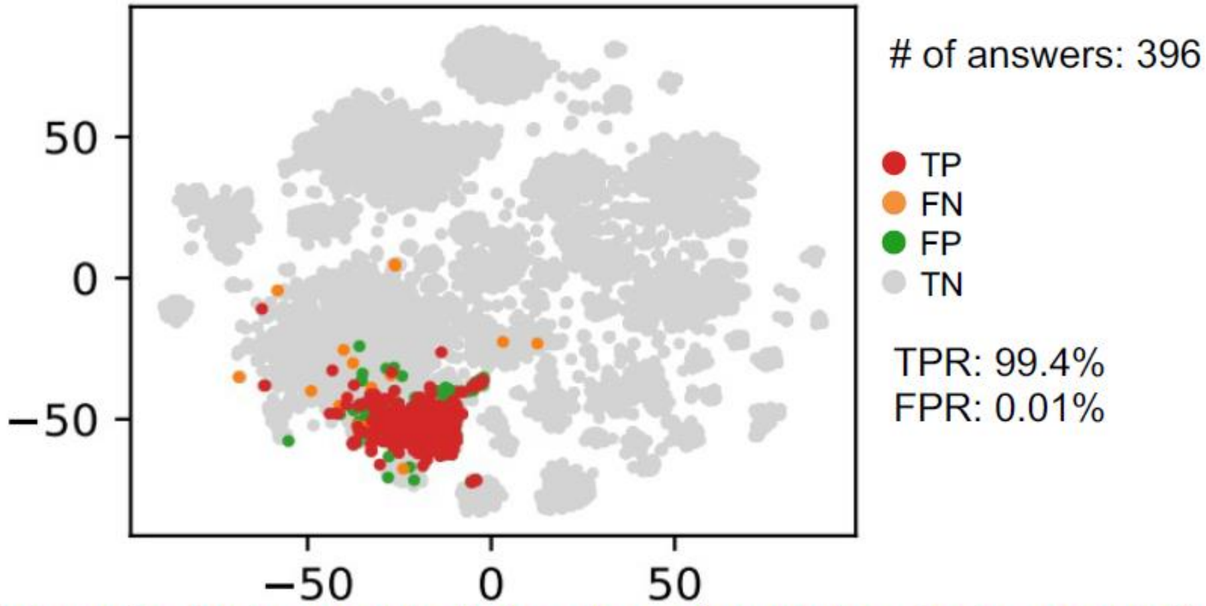
List male instrumentalists who play string instruments”



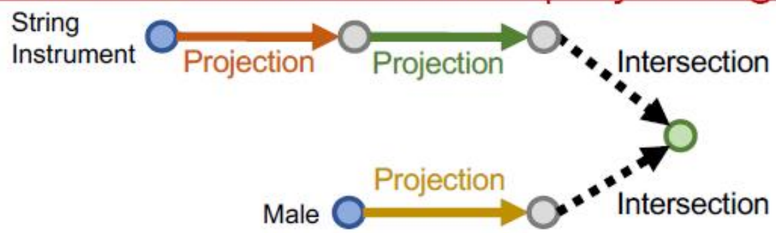
“List male instrumentalists who play string instruments”

두번째 anchor 노드에서 시작 후 세번째 relation으로 이동.

Visualization



"List male instrumentalists who play string instruments"



각각 나온 박스를 intersecton 함수에 삽입. 99.4%의 TPR을 보인다.