Lecture 2: Knowledge Graph Embedding Models

Relational Learning

İsmail İlkan Ceylan Advanced Topics in Machine Learning, University of Oxford 20.01.2021

Overview

Overview

- A glimpse at embedding models

Overview

- A glimpse at embedding models
- Translational models: TransE and RotatE

Overview

- A glimpse at embedding models
- Translational models: TransE and RotatE
- Bilinear models: RESCAL, DistMult, and ComplEx

Overview

- A glimpse at embedding models
- Translational models: TransE and RotatE
- Bilinear models: RESCAL, DistMult, and ComplEx
- Box embedding models

Overview

- A glimpse at embedding models
- Translational models: TransE and RotatE
- Bilinear models: RESCAL, DistMult, and ComplEx
- Box embedding models
- Overview of the embedding models

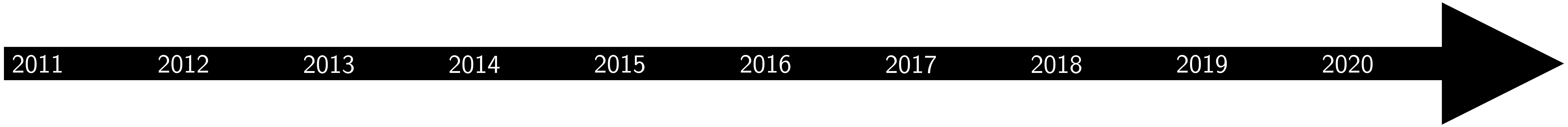
Overview

- A glimpse at embedding models
- Translational models: TransE and RotatE
- Bilinear models: RESCAL, DistMult, and ComplEx
- Box embedding models
- Overview of the embedding models
- Outlook and Discussions

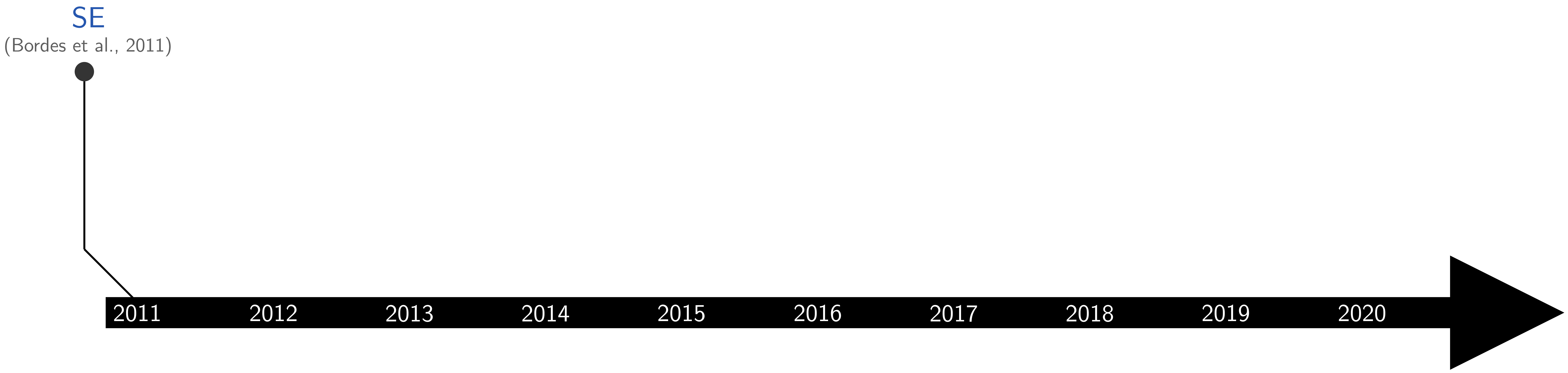
Overview

- A glimpse at embedding models
- Translational models: TransE and RotatE
- Bilinear models: RESCAL, DistMult, and ComplEx
- Box embedding models
- Overview of the embedding models
- Outlook and Discussions
- Summary

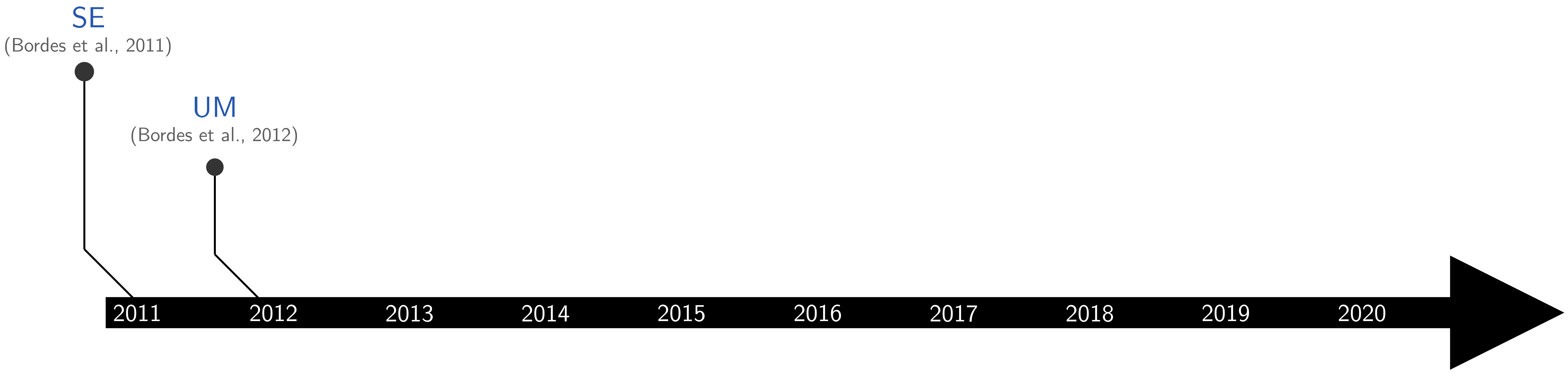
A Glimpse at Embedding Models



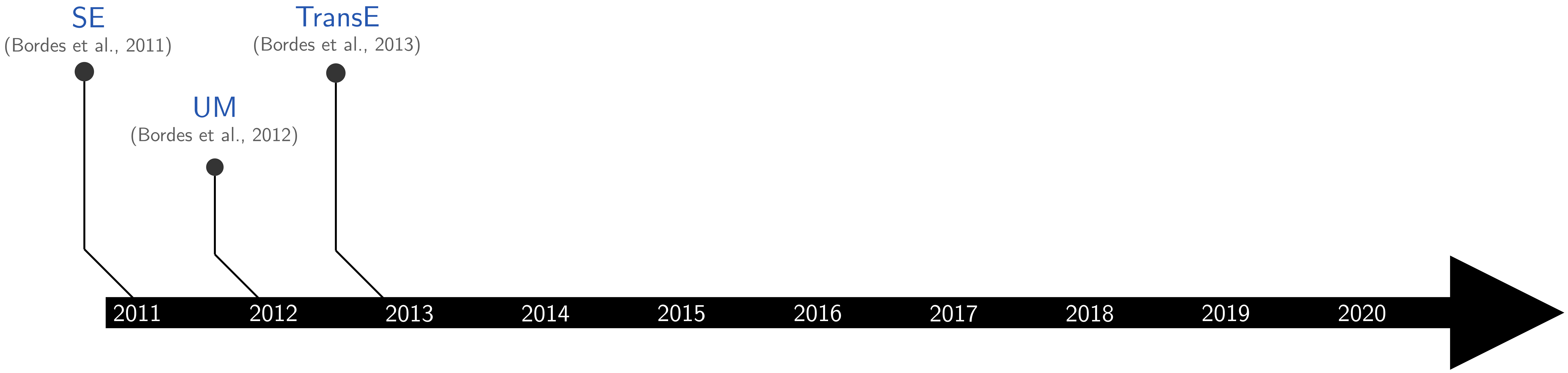
A Glimpse at Embedding Models



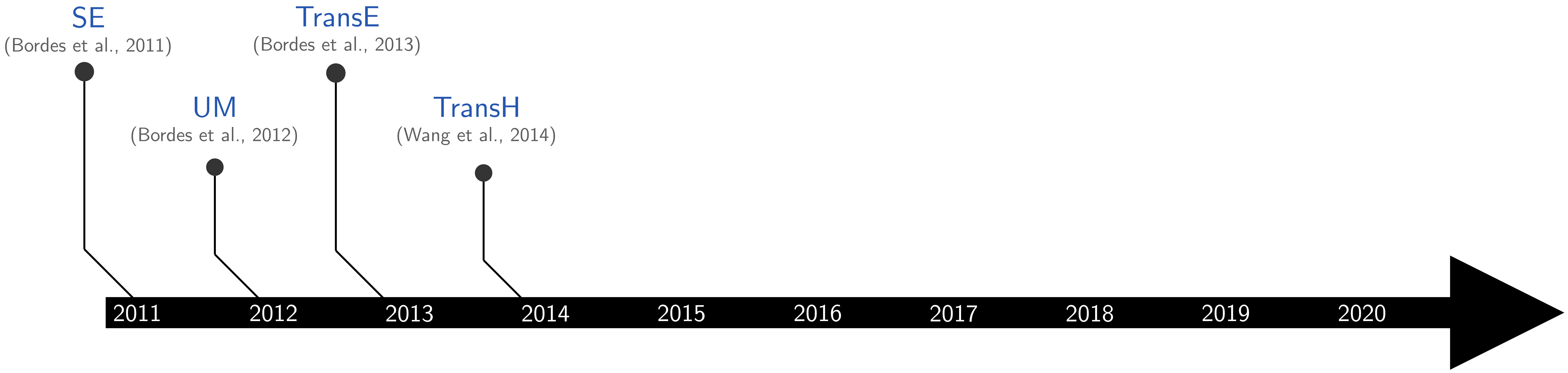
A Glimpse at Embedding Models



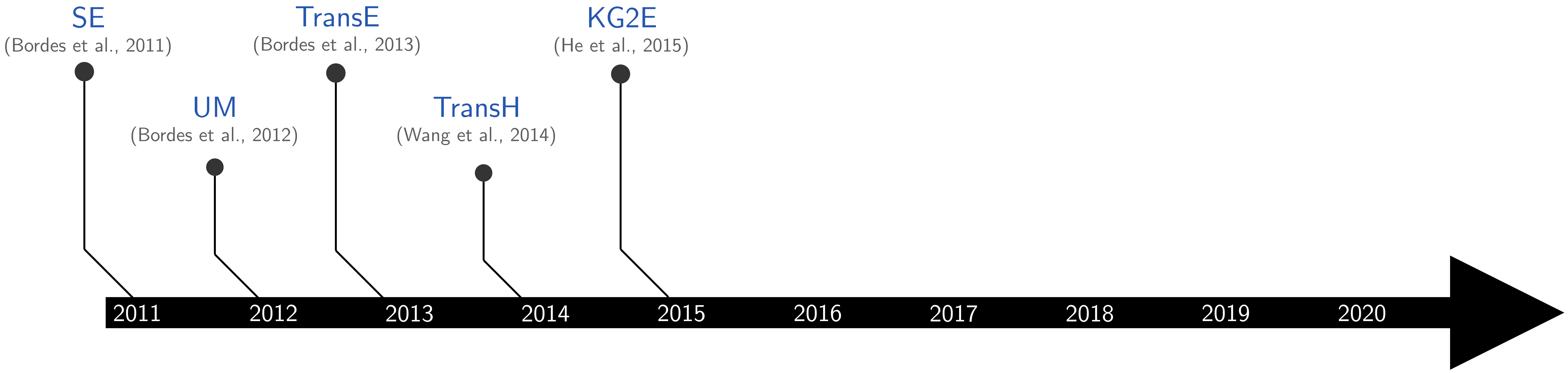
A Glimpse at Embedding Models



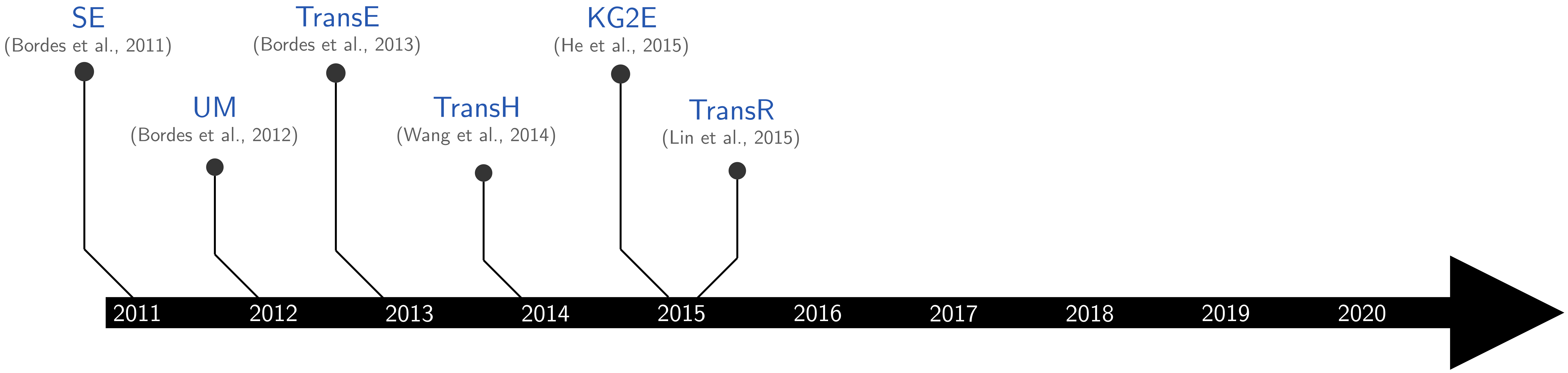
A Glimpse at Embedding Models



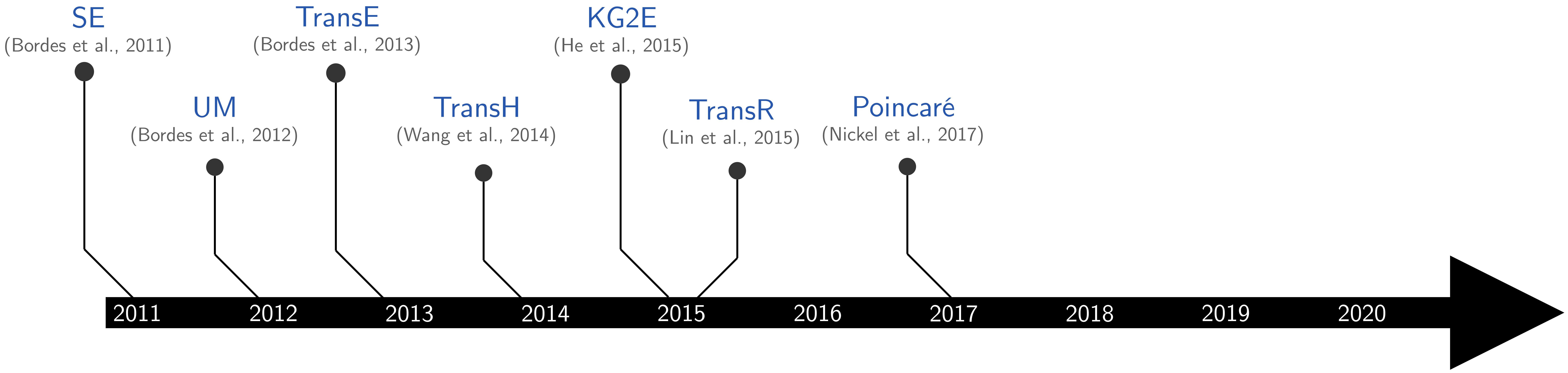
A Glimpse at Embedding Models



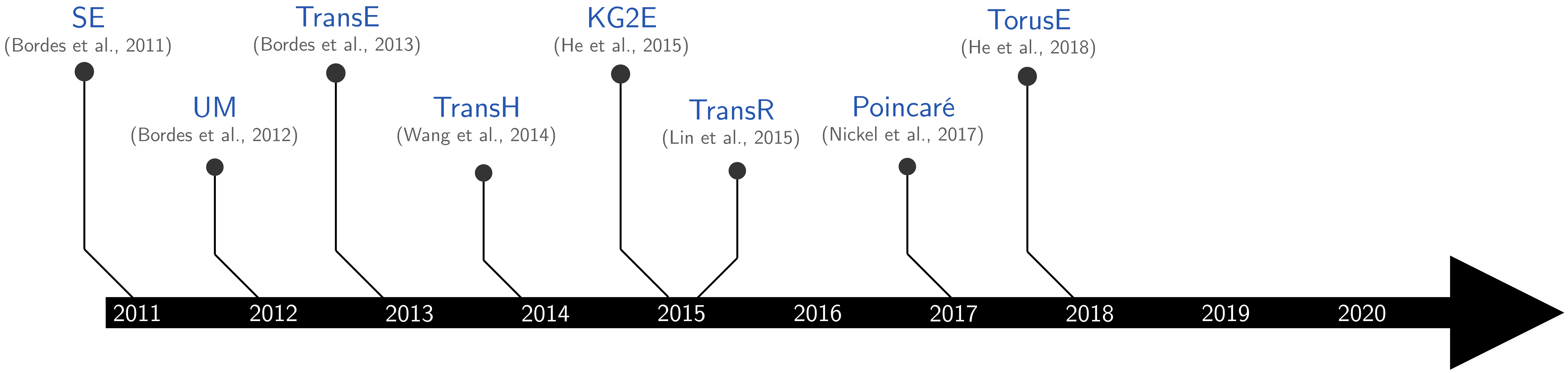
A Glimpse at Embedding Models



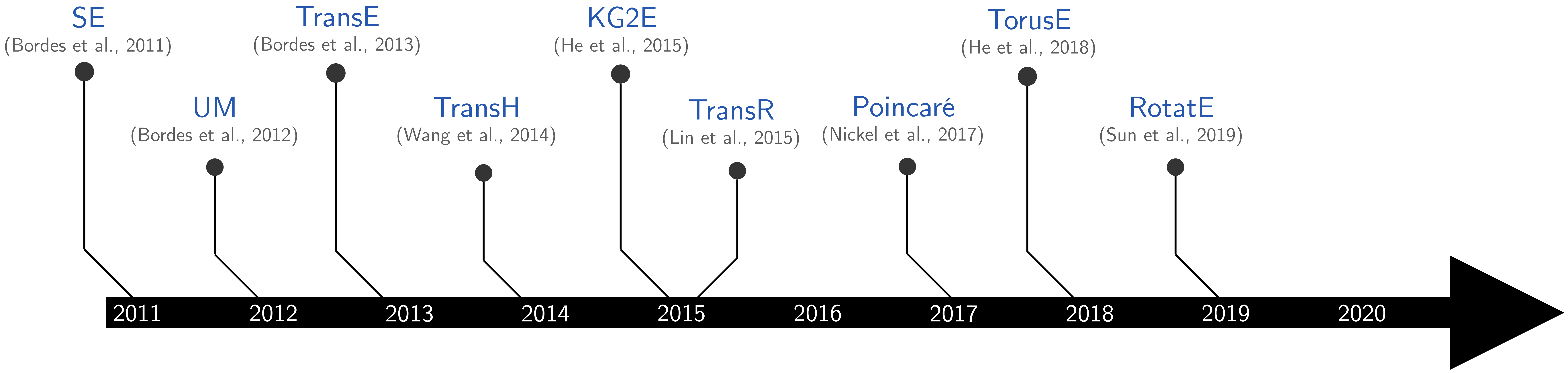
A Glimpse at Embedding Models



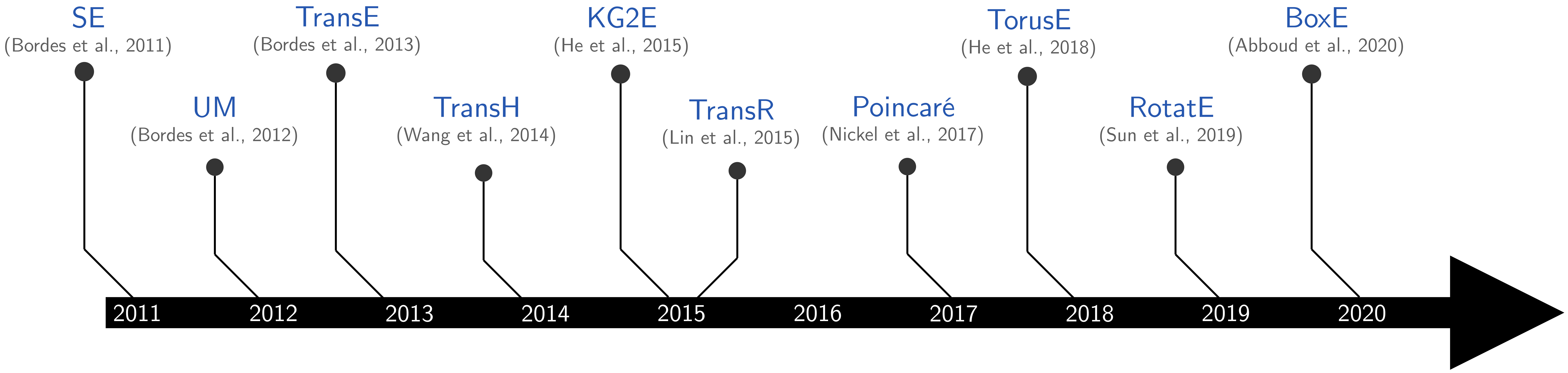
A Glimpse at Embedding Models



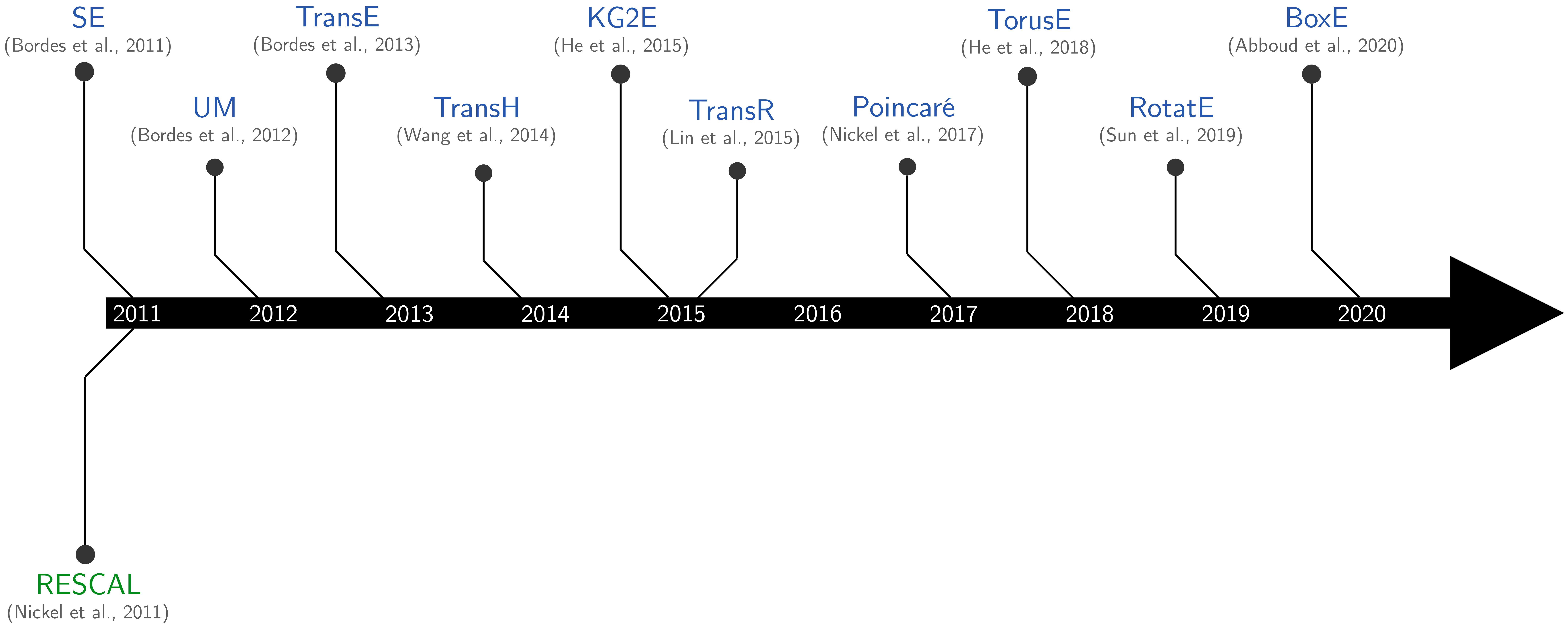
A Glimpse at Embedding Models



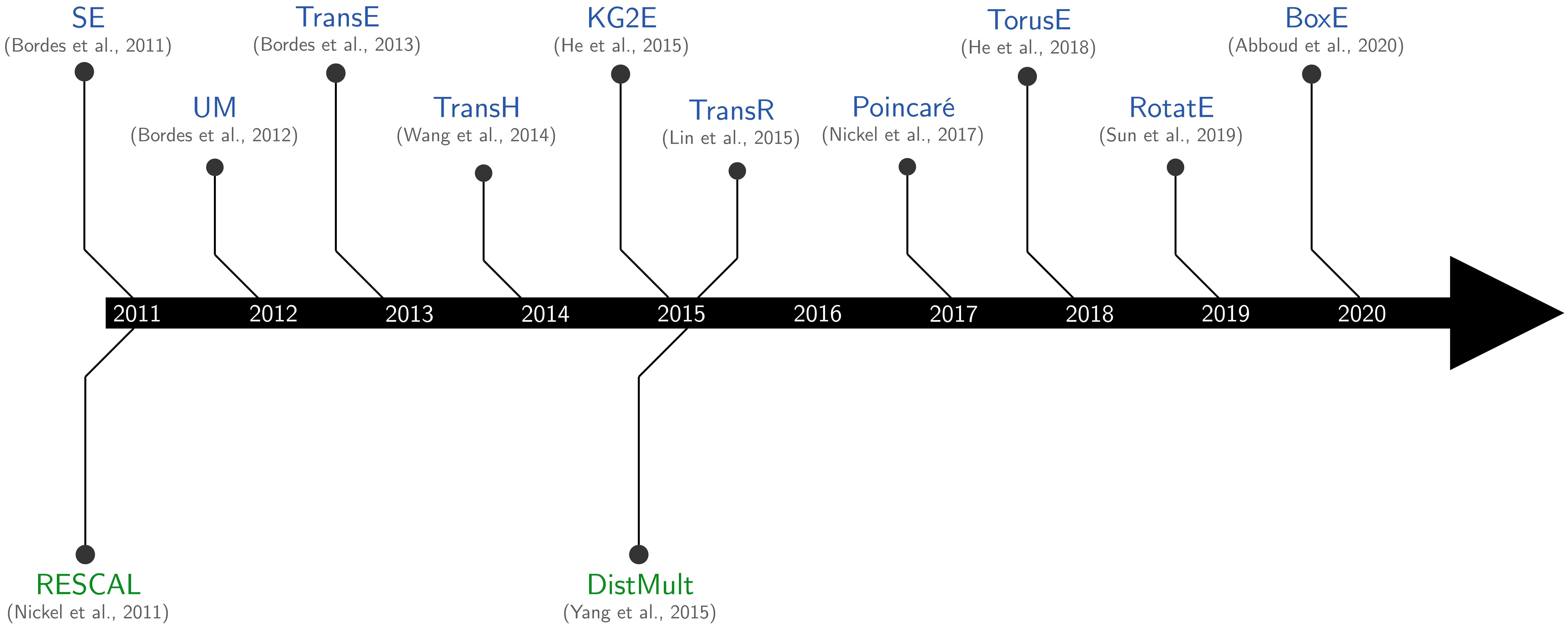
A Glimpse at Embedding Models



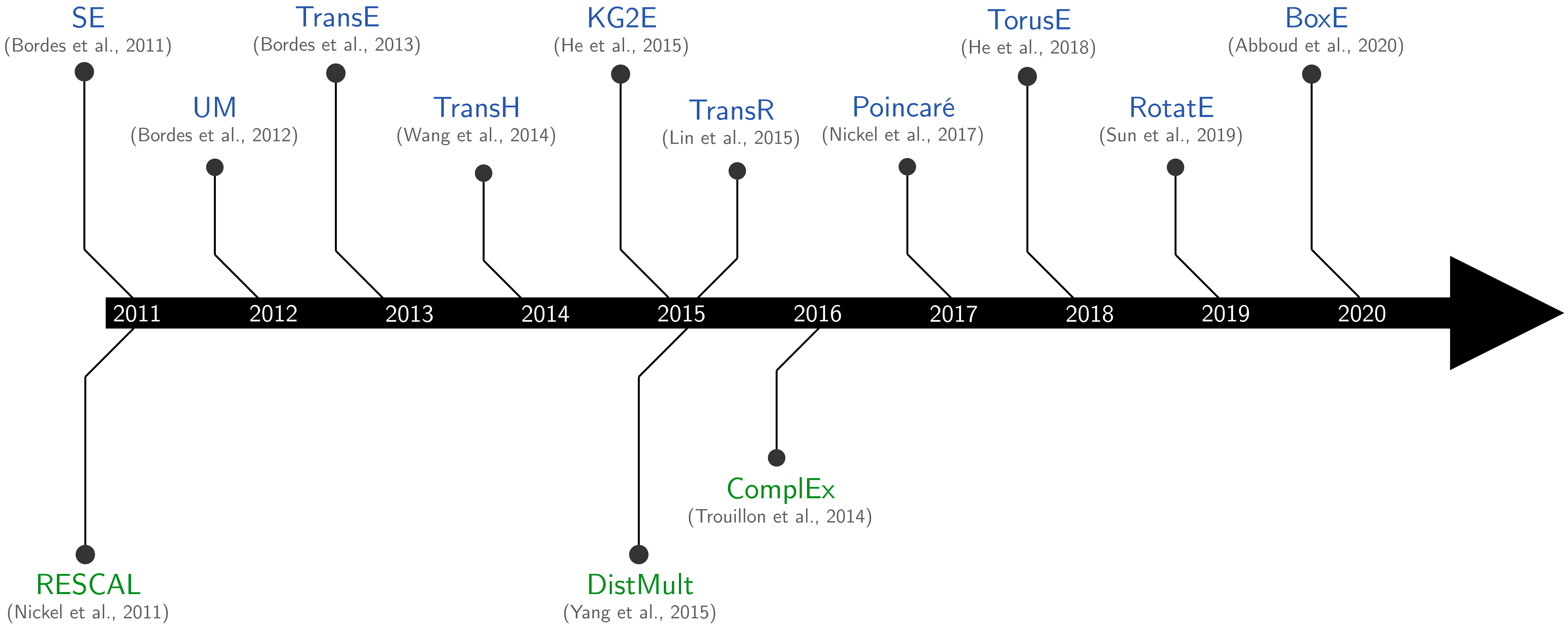
A Glimpse at Embedding Models



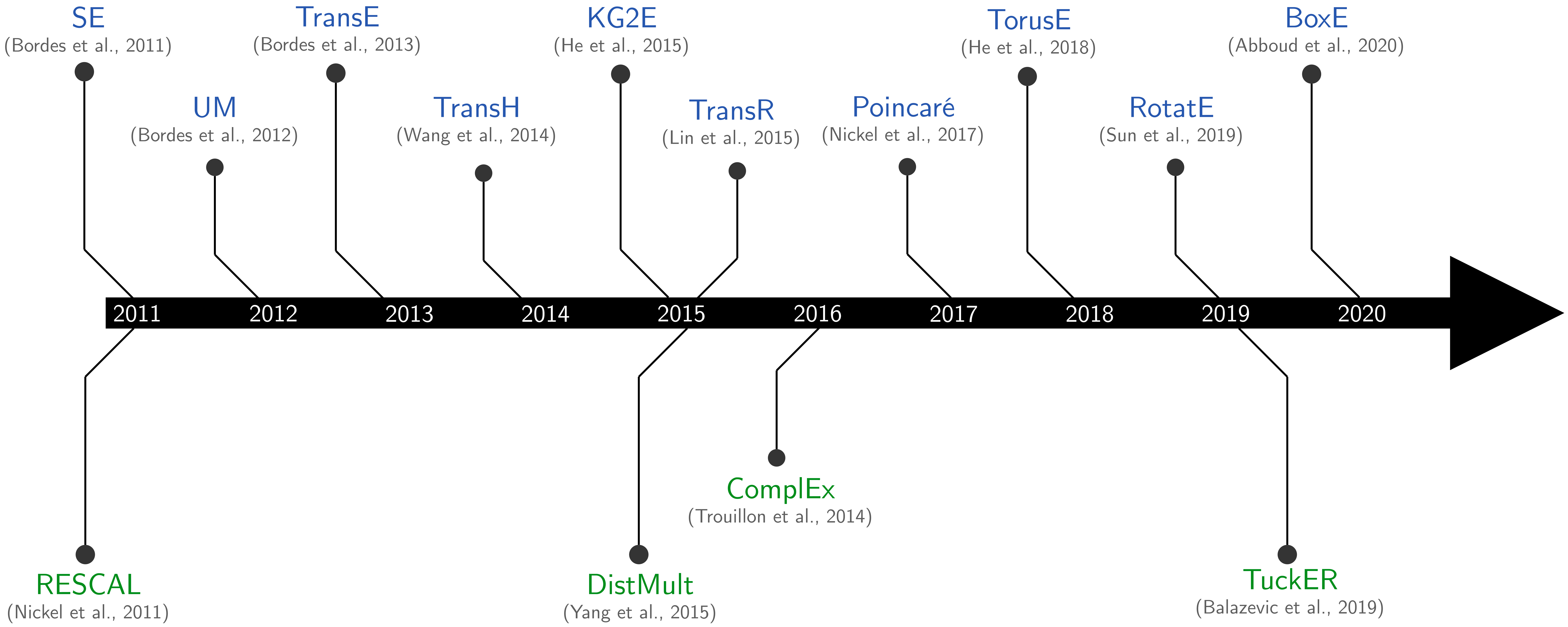
A Glimpse at Embedding Models



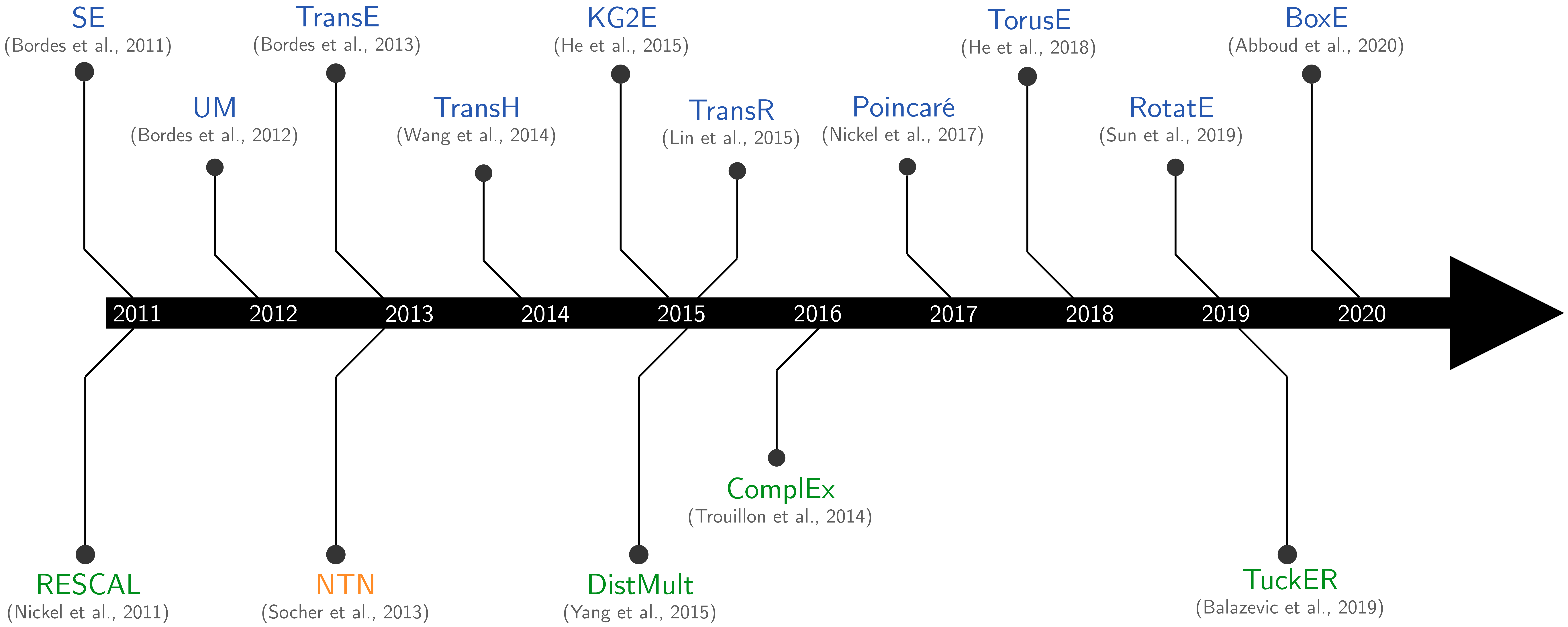
A Glimpse at Embedding Models



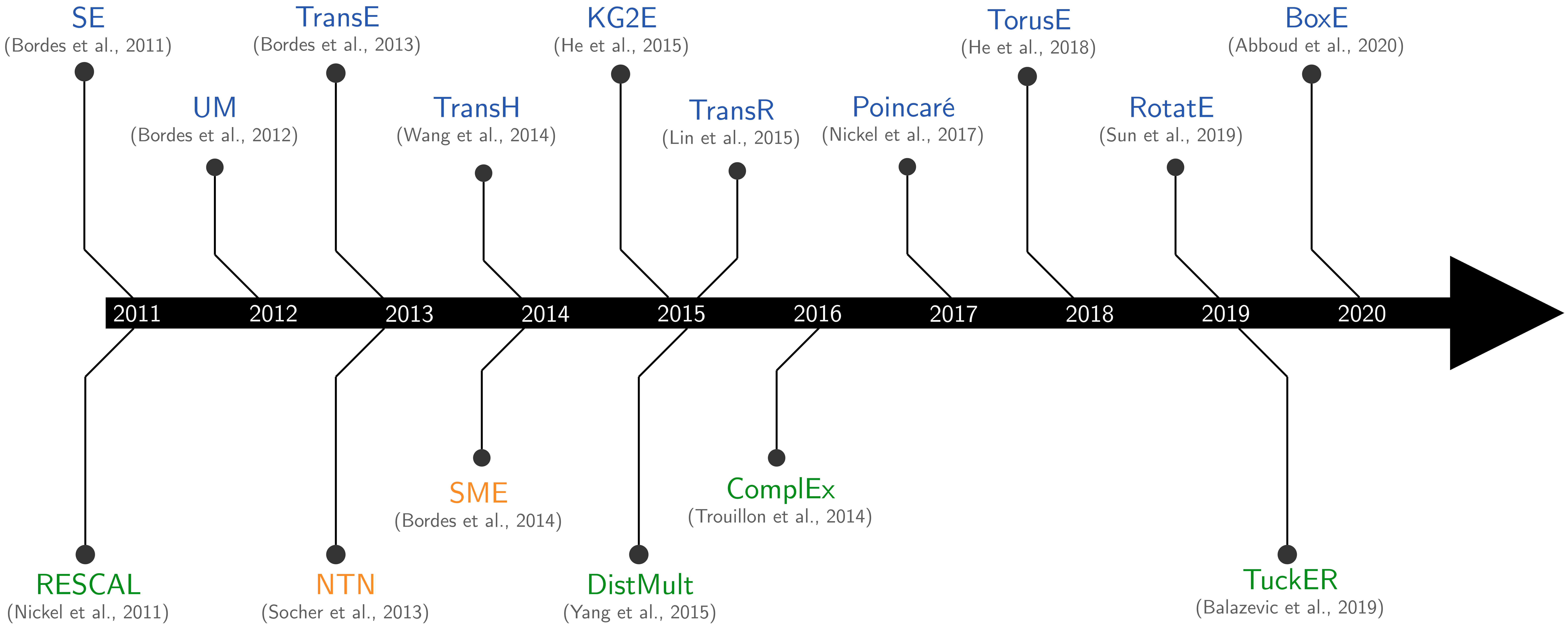
A Glimpse at Embedding Models



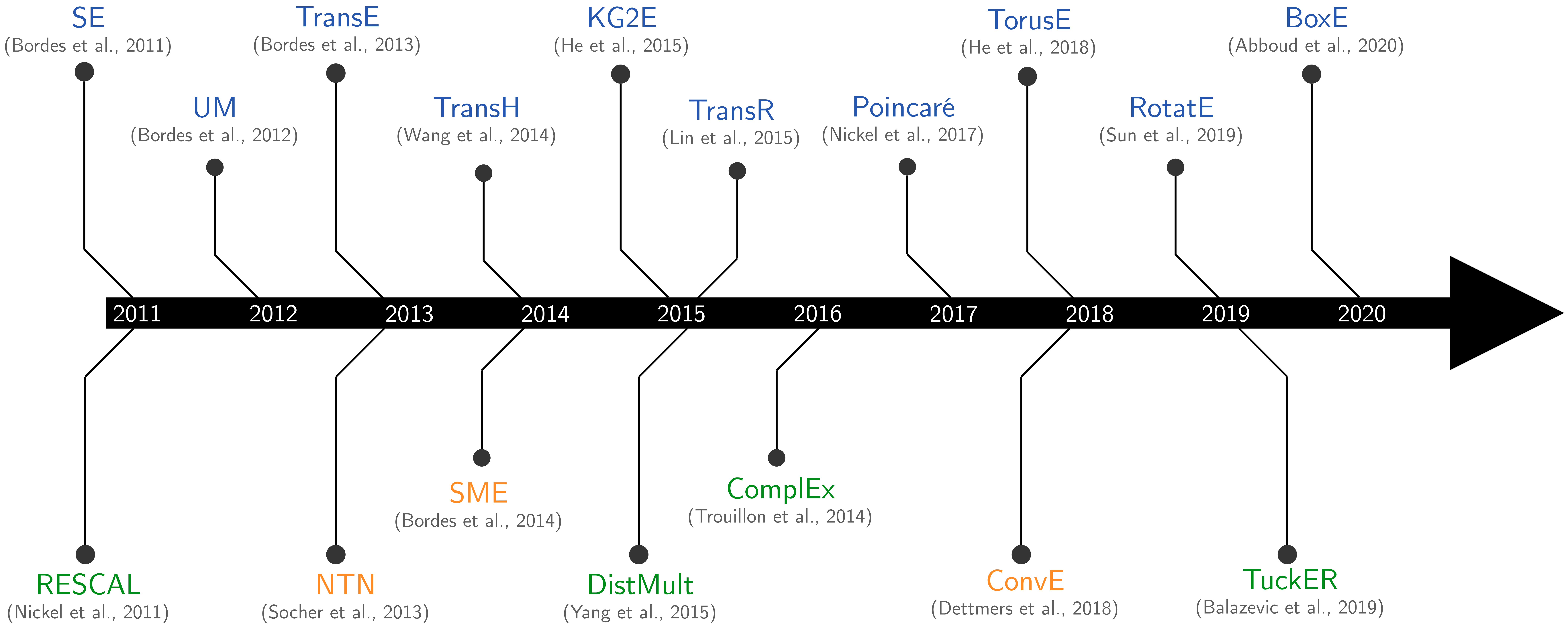
A Glimpse at Embedding Models



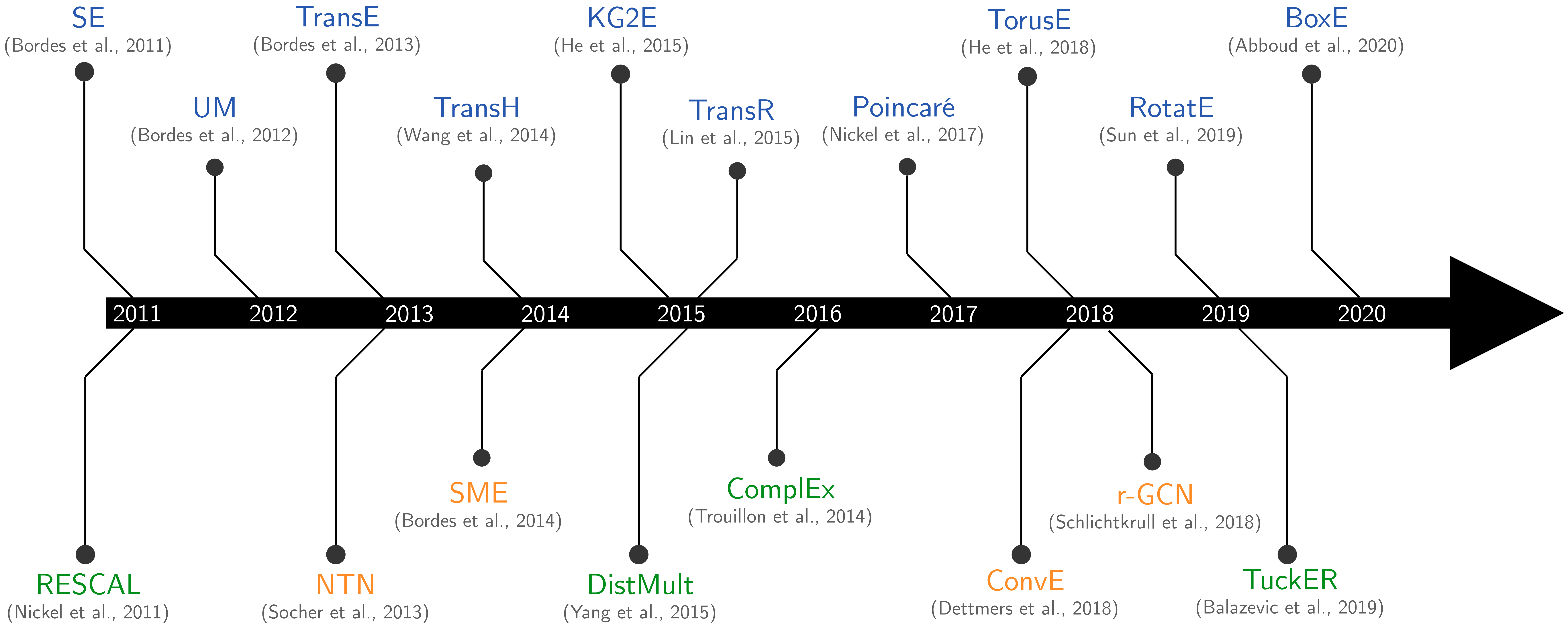
A Glimpse at Embedding Models



A Glimpse at Embedding Models



A Glimpse at Embedding Models

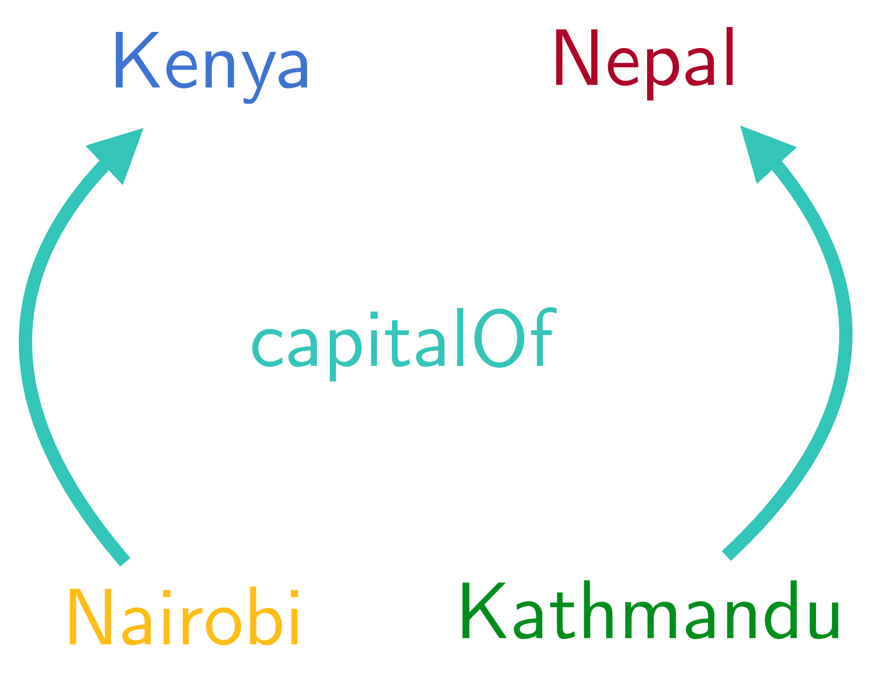


Translational Models

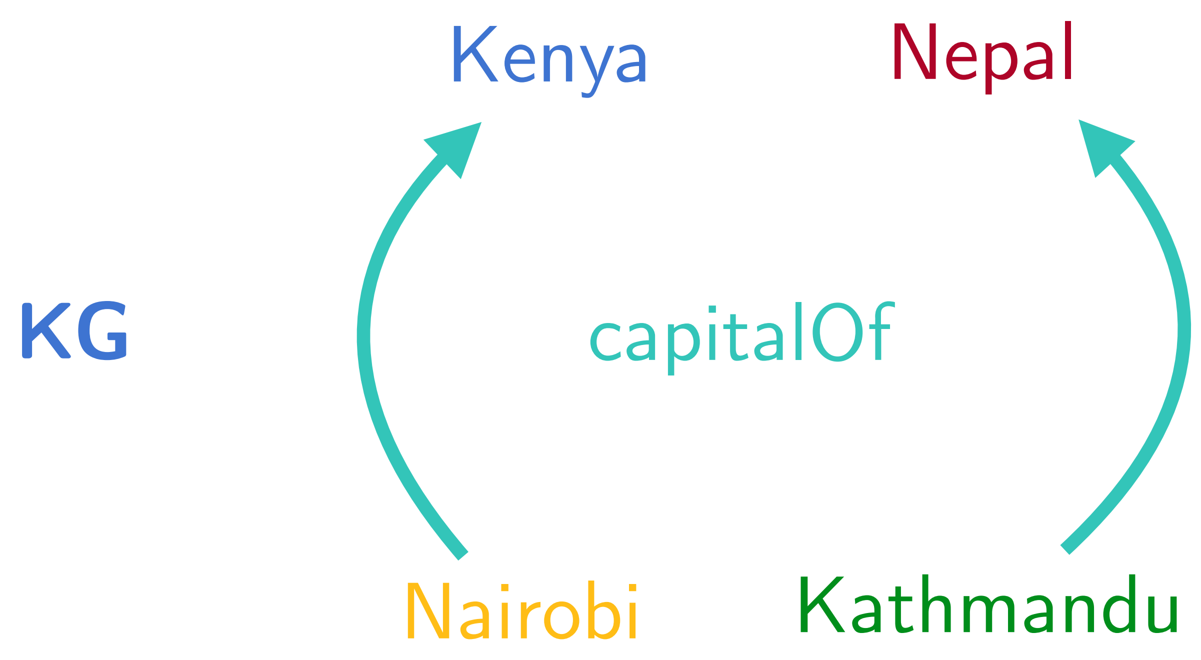
TransE: Representation

TransE: Representation

KG

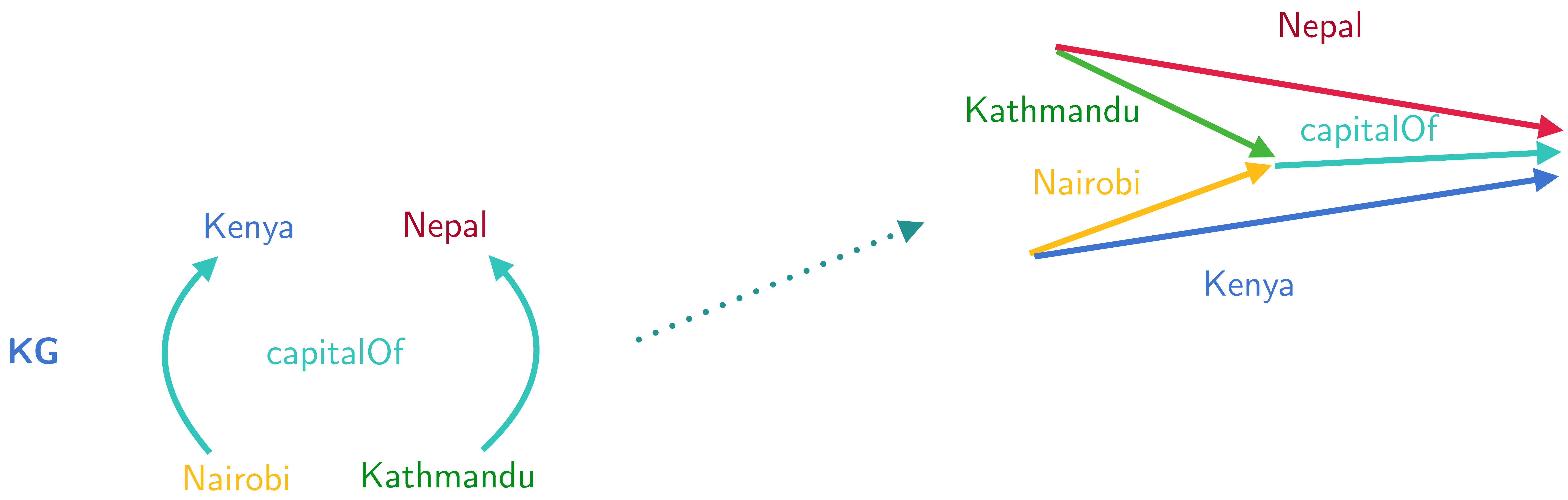


TransE: Representation



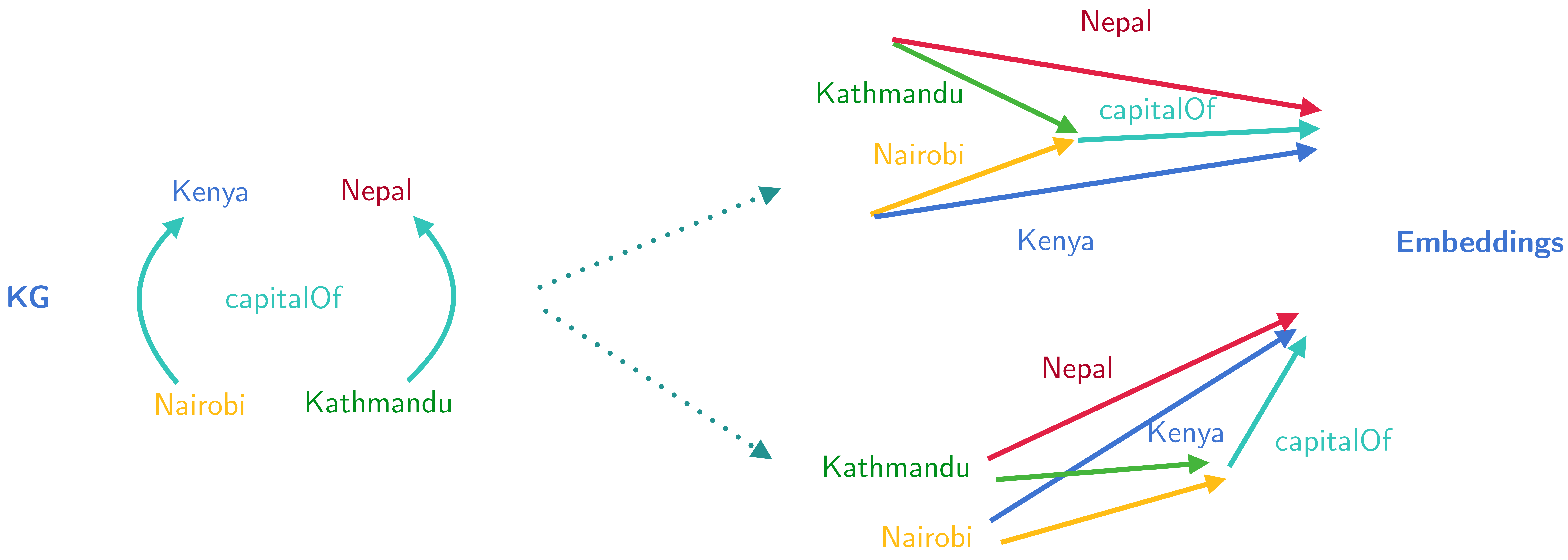
- TransE encodes entities $h, t \in E$ and relations $r \in R$, through d-dimensional vectors $\mathbf{h}, \mathbf{t}, \mathbf{r} \in \mathbb{R}^d$.

TransE: Representation



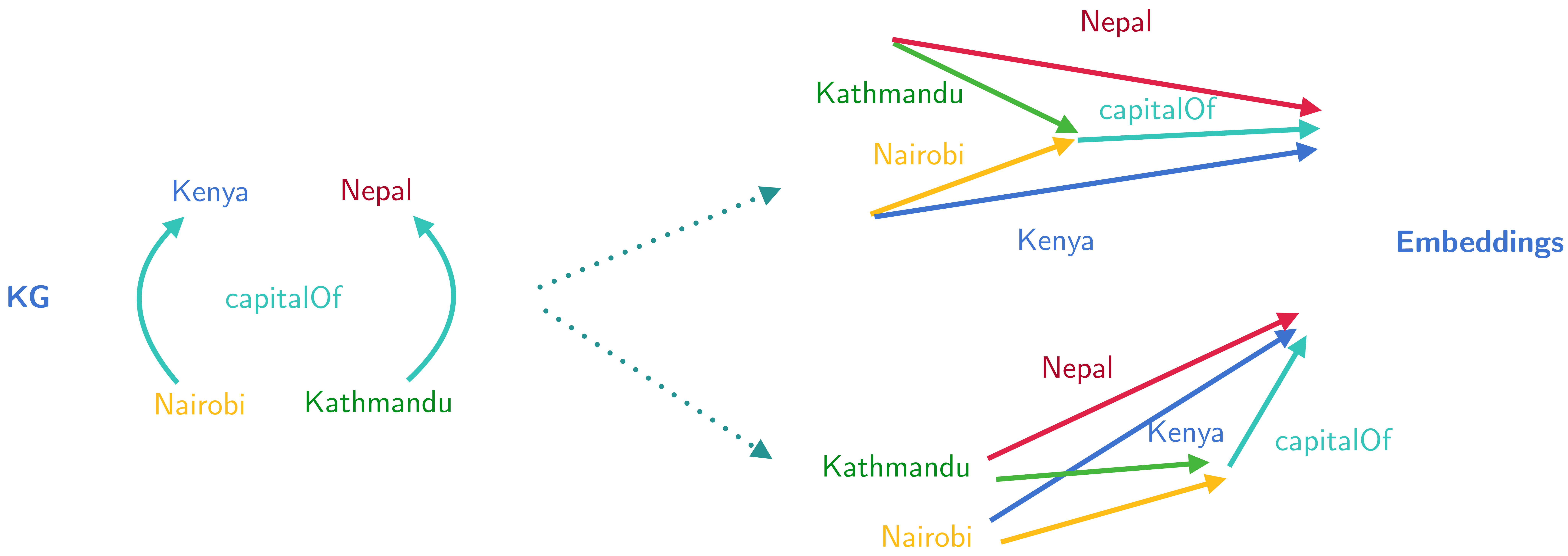
- TransE encodes entities $h, t \in E$ and relations $r \in R$, through d-dimensional vectors $\mathbf{h}, \mathbf{t}, \mathbf{r} \in \mathbb{R}^d$.

TransE: Representation



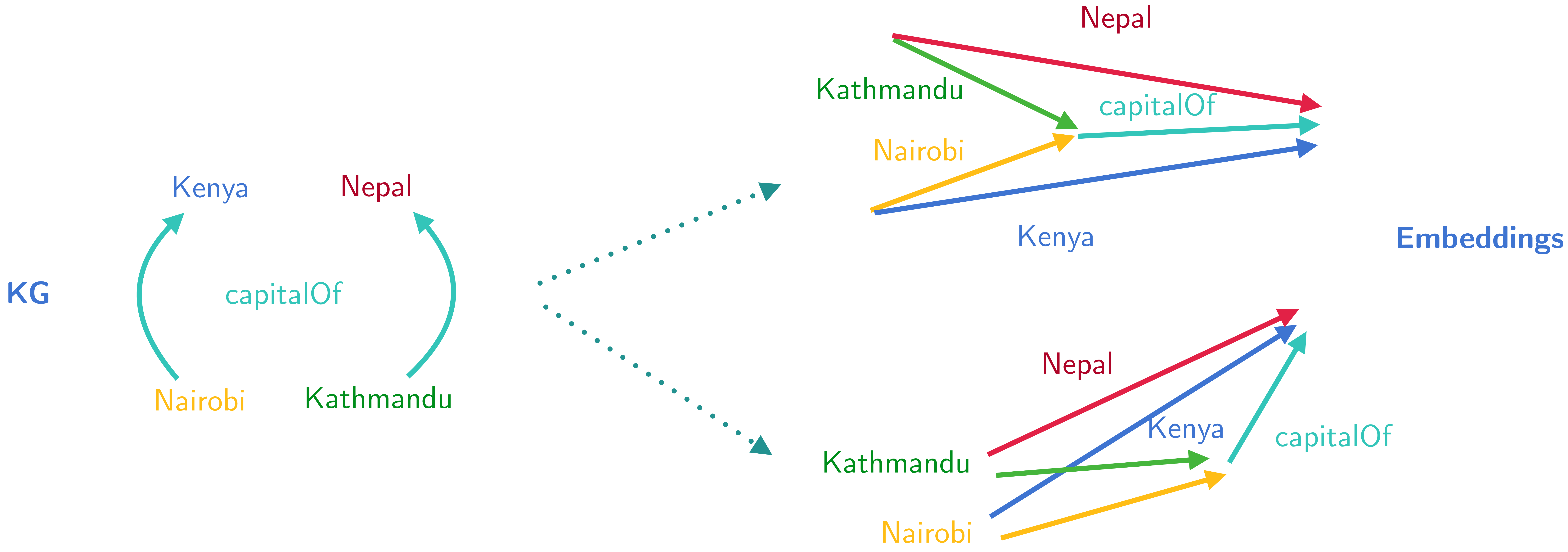
- TransE encodes entities $h, t \in E$ and relations $r \in R$, through d-dimensional vectors $\mathbf{h}, \mathbf{t}, \mathbf{r} \in \mathbb{R}^d$.

TransE: Representation



- TransE encodes entities $h, t \in E$ and relations $r \in R$, through d-dimensional vectors $\mathbf{h}, \mathbf{t}, \mathbf{r} \in \mathbb{R}^d$.
- TransE scores a fact $r(h, t)$ depending how similar $\mathbf{h} + \mathbf{r}$ and $\mathbf{t}$ are, i.e., $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$.

TransE: Representation



- TransE encodes entities $h, t \in E$ and relations $r \in R$, through d-dimensional vectors $\mathbf{h}, \mathbf{t}, \mathbf{r} \in \mathbb{R}^d$.
- TransE scores a fact $r(h, t)$ depending how similar $\mathbf{h} + \mathbf{r}$ and $\mathbf{t}$ are, i.e., $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$.
- TransE is optimised to minimise (resp., maximise) the dissimilarity of true facts (resp., negative facts).

TransE: Optimisation, Loss, Training

TransE: Optimisation, Loss, Training

Scoring: Consider a dissimilarity measure d, such as L_1 or L_2 norm, where $d(\mathbf{h} + \mathbf{r}, \mathbf{t})$ represents how dissimilar $\mathbf{h} + \mathbf{r}$, and $\mathbf{t}$ are, e.g., $d(\mathbf{h} + \mathbf{r}, \mathbf{t}) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|$.

TransE: Optimisation, Loss, Training

Scoring: Consider a dissimilarity measure d, such as L_1 or L_2 norm, where $d(\mathbf{h}+\mathbf{r},\mathbf{t})$ represents how dissimilar $\mathbf{h}+\mathbf{r}$, and $\mathbf{t}$ are, e.g., $d(\mathbf{h}+\mathbf{r},\mathbf{t}) = \|\mathbf{h}+\mathbf{r}-\mathbf{t}\|$.

Loss: For every fact $r(h,t)$, TransE defines the loss function:

TransE: Optimisation, Loss, Training

Scoring: Consider a dissimilarity measure d, such as L_1 or L_2 norm, where $d(\mathbf{h}+\mathbf{r},\mathbf{t})$ represents how dissimilar $\mathbf{h}+\mathbf{r}$, and $\mathbf{t}$ are, e.g., $d(\mathbf{h}+\mathbf{r},\mathbf{t}) = \|\mathbf{h}+\mathbf{r}-\mathbf{t}\|$.

Loss: For every fact $r(h,t)$, TransE defines the loss function:

$$\mathscr{L} = \gamma + d(\mathbf{h}+\mathbf{r},\mathbf{t}) + \sum_{r(h',t')\in N^{r(h,t)}} - d(\mathbf{h}'+\mathbf{r},\mathbf{t}') - \gamma \ ,$$

TransE: Optimisation, Loss, Training

Scoring: Consider a dissimilarity measure d, such as L_1 or L_2 norm, where $d(\mathbf{h} + \mathbf{r}, \mathbf{t})$ represents how dissimilar $\mathbf{h} + \mathbf{r}$, and $\mathbf{t}$ are, e.g., $d(\mathbf{h} + \mathbf{r}, \mathbf{t}) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|$.

Loss: For every fact $r(h, t)$, TransE defines the loss function:

$$\mathscr{L} = \gamma + d(\mathbf{h} + \mathbf{r}, \mathbf{t}) + \sum_{r(h',t') \in N^{r(h,t)}} - d(\mathbf{h}' + \mathbf{r}, \mathbf{t}') - \gamma \ ,$$

where γ is a margin hyper-parameter, and $N^{r(h,t)}$ is a set of negative samples for $r(h, t)$.

TransE: Optimisation, Loss, Training

Scoring: Consider a dissimilarity measure d, such as L_1 or L_2 norm, where $d(\mathbf{h}+\mathbf{r},\mathbf{t})$ represents how dissimilar $\mathbf{h}+\mathbf{r}$, and $\mathbf{t}$ are, e.g., $d(\mathbf{h}+\mathbf{r},\mathbf{t}) = \|\mathbf{h}+\mathbf{r}-\mathbf{t}\|$.

Loss: For every fact $r(h,t)$, TransE defines the loss function:

$$\mathscr{L} = \gamma + d(\mathbf{h}+\mathbf{r},\mathbf{t}) + \sum_{r(h',t')\in N^{r(h,t)}} - d(\mathbf{h}'+\mathbf{r},\mathbf{t}') - \gamma \ ,$$

where γ is a margin hyper-parameter, and $N^{r(h,t)}$ is a set of negative samples for $r(h,t)$.

The loss function favours lower values of dissimilarity for true facts than for negative facts, and is thus a natural implementation of the intended criterion.

TransE: Optimisation, Loss, Training

Scoring: Consider a dissimilarity measure d, such as L_1 or L_2 norm, where $d(\mathbf{h}+\mathbf{r},\mathbf{t})$ represents how dissimilar $\mathbf{h}+\mathbf{r}$, and $\mathbf{t}$ are, e.g., $d(\mathbf{h}+\mathbf{r},\mathbf{t}) = \|\mathbf{h}+\mathbf{r}-\mathbf{t}\|$.

Loss: For every fact $r(h,t)$, TransE defines the loss function:

$$\mathscr{L} = \gamma + d(\mathbf{h}+\mathbf{r},\mathbf{t}) + \sum_{r(h',t')\in N^{r(h,t)}} - d(\mathbf{h}'+\mathbf{r},\mathbf{t}') - \gamma \ ,$$

where γ is a margin hyper-parameter, and $N^{r(h,t)}$ is a set of negative samples for $r(h,t)$.

The loss function favours lower values of dissimilarity for true facts than for negative facts, and is thus a natural implementation of the intended criterion.

Optimisation: The optimisation is carried out by stochastic gradient descent, where all embeddings for entities and relationships are first initialised randomly; at each iteration, the parameters are updated by taking a gradient step with constant learning rate. The algorithm is stopped based on its performance on a validation set.

How expressive is TransE?

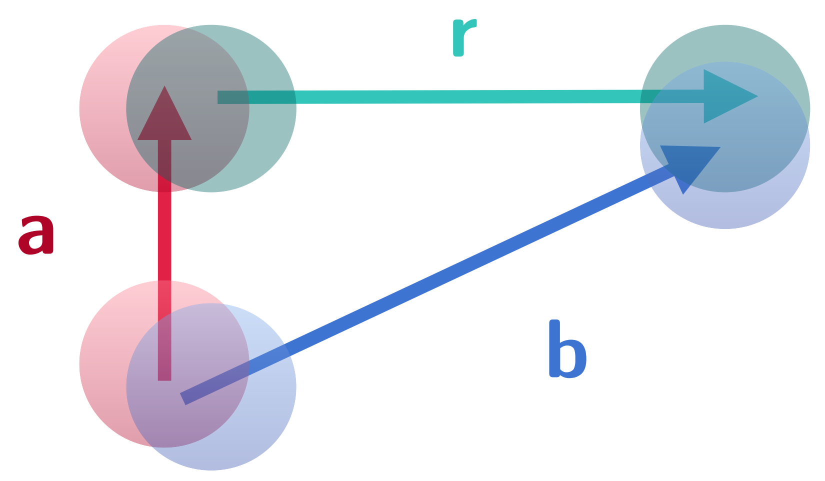
How expressive is TransE?

Let us realise the set of true facts $\{r(a,b), r(b,a)\}$ in TransE.
These facts can be clearly be realised independently, e.g. (i) & (ii).

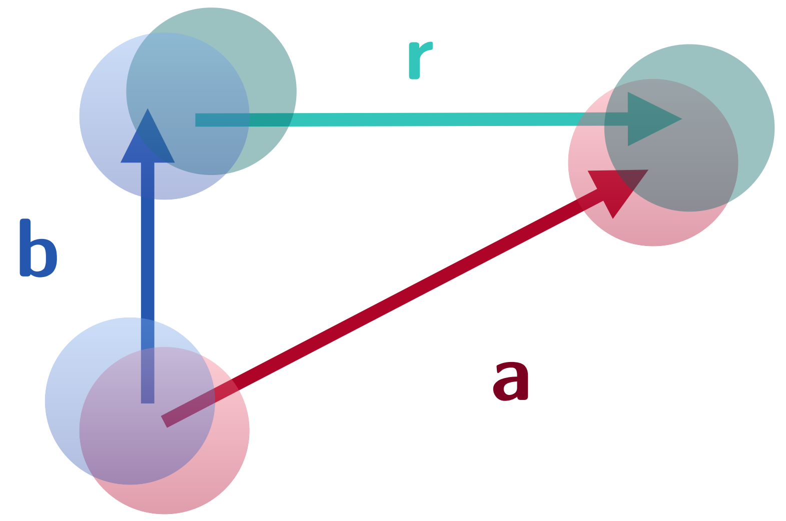
How expressive is TransE?

Let us realise the set of true facts $\{r(a,b), r(b,a)\}$ in TransE.
These facts can be clearly be realised independently, e.g. (i) & (ii).

(i) $r(a,b)$



(ii) $r(b,a)$

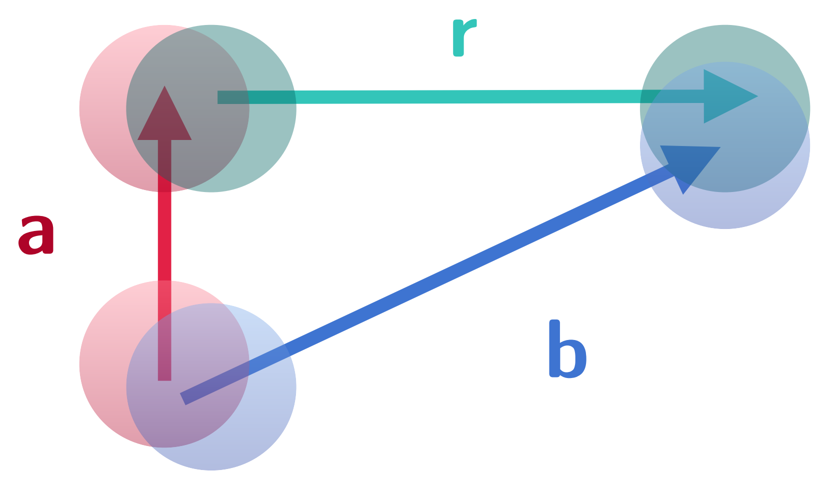


How expressive is TransE?

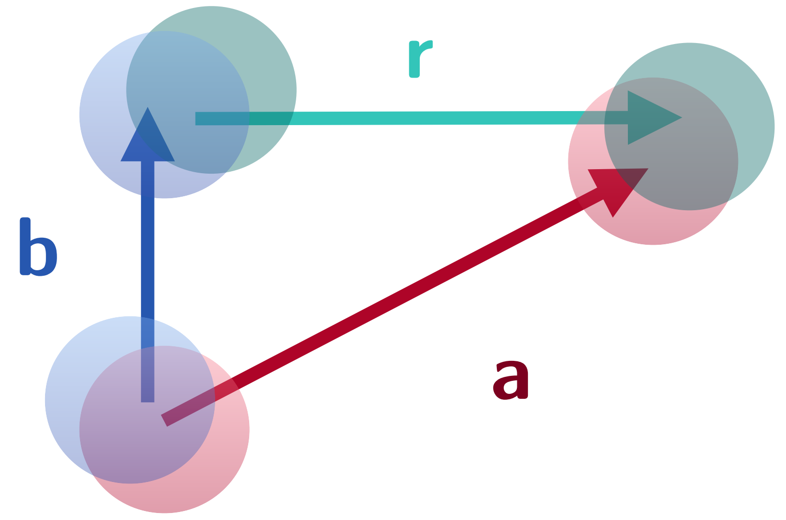
Let us realise the set of true facts $\{r(a,b), r(b,a)\}$ in TransE. These facts can be clearly be realised independently, e.g. (i) & (ii).

To realise these facts jointly, we need $r = 0$, as shown in (iii), but then, the facts $\{r(a,a), r(b,b)\}$, are necessarily classified as true facts, although these could well be false facts.

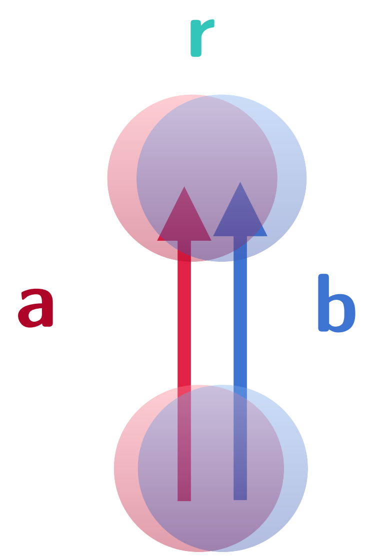
(i) $r(a,b)$



(ii) $r(b,a)$



(iii) $r(a,b)$ & $r(b,a)$



How expressive is TransE?

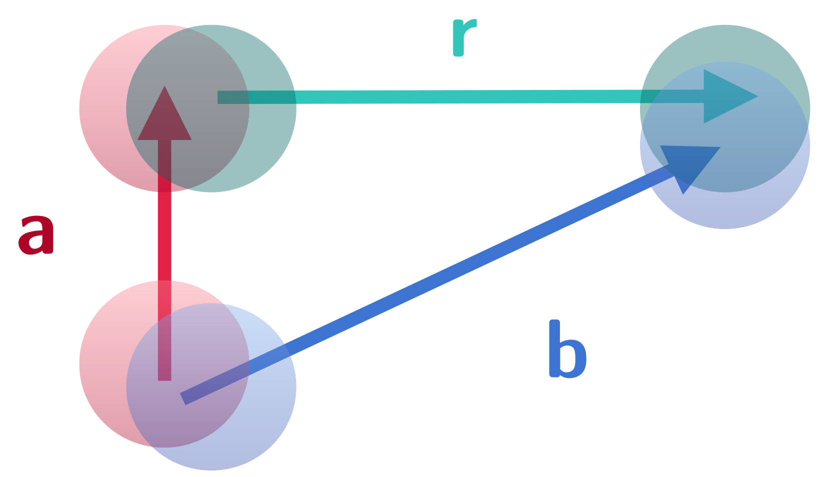
Let us realise the set of true facts $\{r(a,b), r(b,a)\}$ in TransE. These facts can be clearly be realised independently, e.g. (i) & (ii).

To realise these facts jointly, we need $r = 0$, as shown in (iii), but then, the facts $\{r(a,a), r(b,b)\}$, are necessarily classified as true facts, although these could well be false facts.

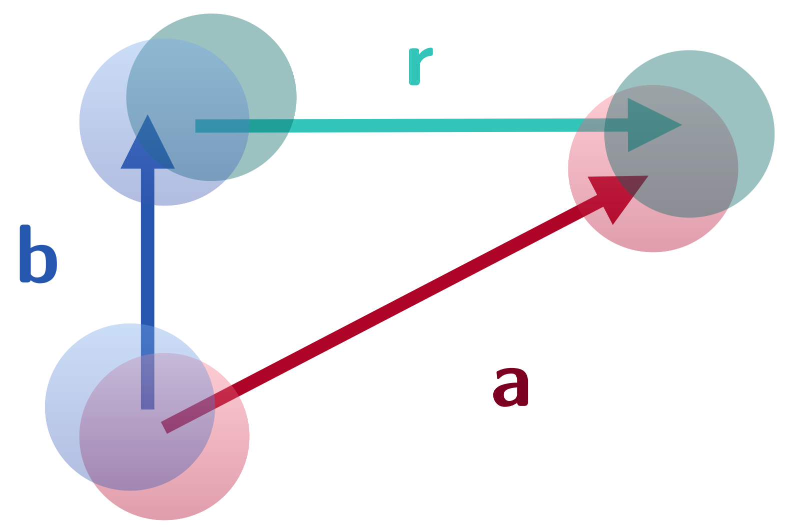
This also means that the relation r can be made symmetric only by additionally forcing r to be reflexive, hence leading to loss of generality!

TransE is not fully expressive, as it cannot encode the set of true facts $\{r(a,b), r(b,a)\}$ and the set of false facts $\{r(a,a), r(b,b)\}$ simultaneously.

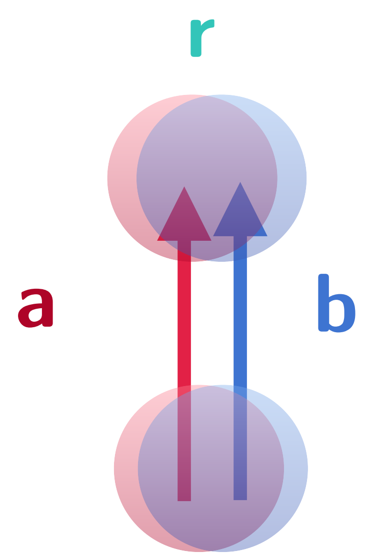
(i) $r(a,b)$



(ii) $r(b,a)$



(iii) $r(a,b)$ & $r(b,a)$



How expressive is TransE?

Let us realise the set of true facts $\{r(a,b), r(b,a)\}$ in TransE. These facts can be clearly be realised independently, e.g. (i) & (ii).

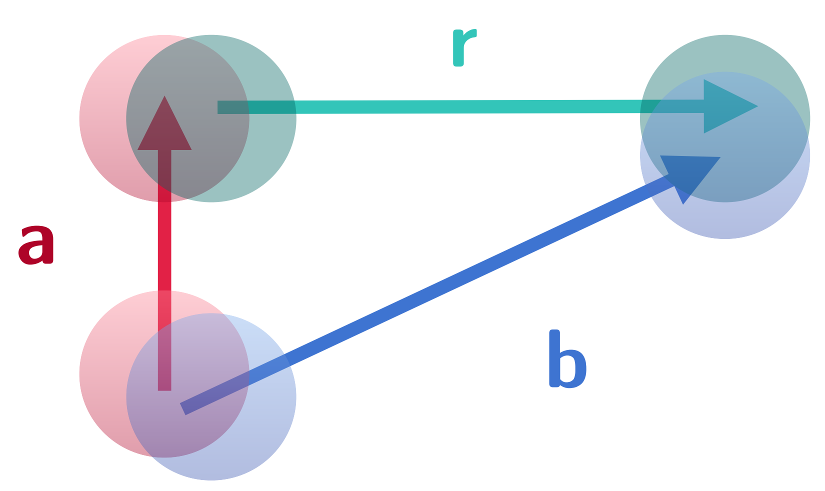
To realise these facts jointly, we need $r = 0$, as shown in (iii), but then, the facts $\{r(a,a), r(b,b)\}$, are necessarily classified as true facts, although these could well be false facts.

This also means that the relation r can be made symmetric only by additionally forcing r to be reflexive, hence leading to loss of generality!

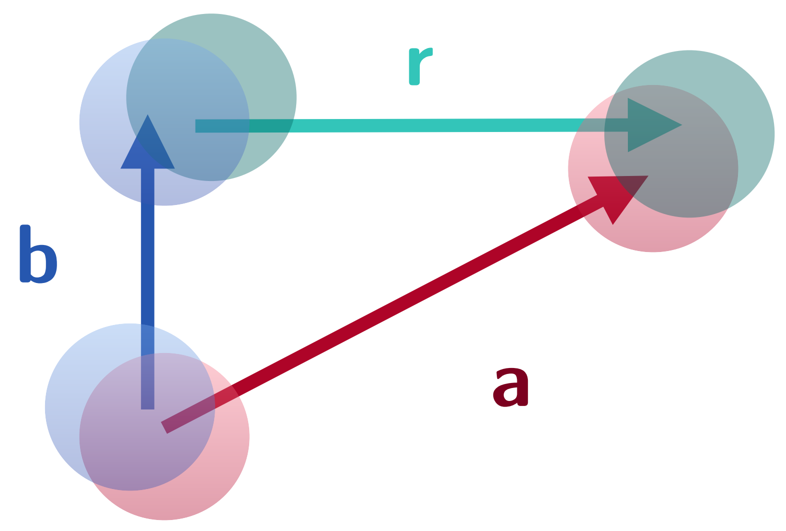
TransE is not fully expressive, as it cannot encode the set of true facts $\{r(a,b), r(b,a)\}$ and the set of false facts $\{r(a,a), r(b,b)\}$ simultaneously.

Consider a relation such as cousinOf with entities alice, bob to see a problematic example. TransE is limited in various other ways, as we shall see later.

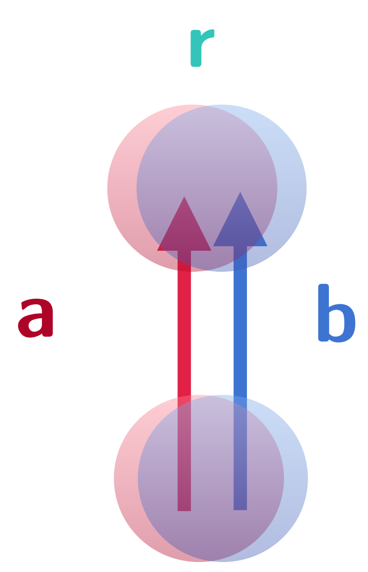
(i) $r(a,b)$



(ii) $r(b,a)$



(iii) $r(a,b)$ & $r(b,a)$

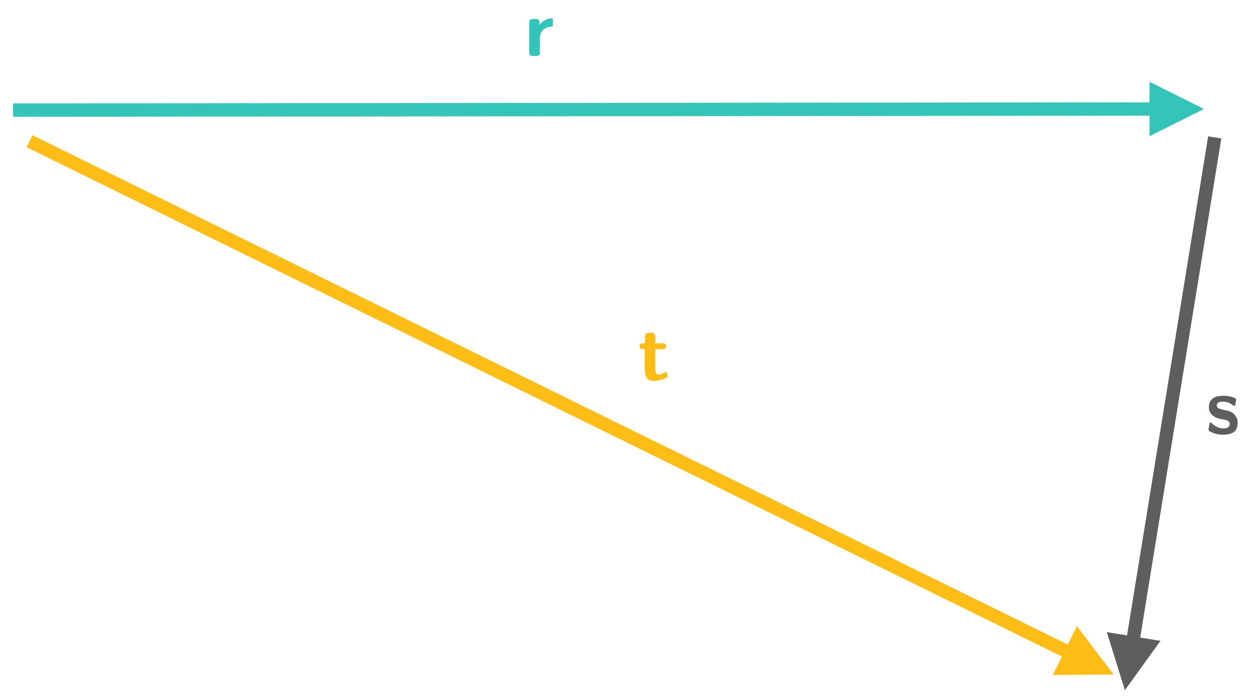


Which inference patterns can TransE capture?

Which inference patterns can TransE capture?

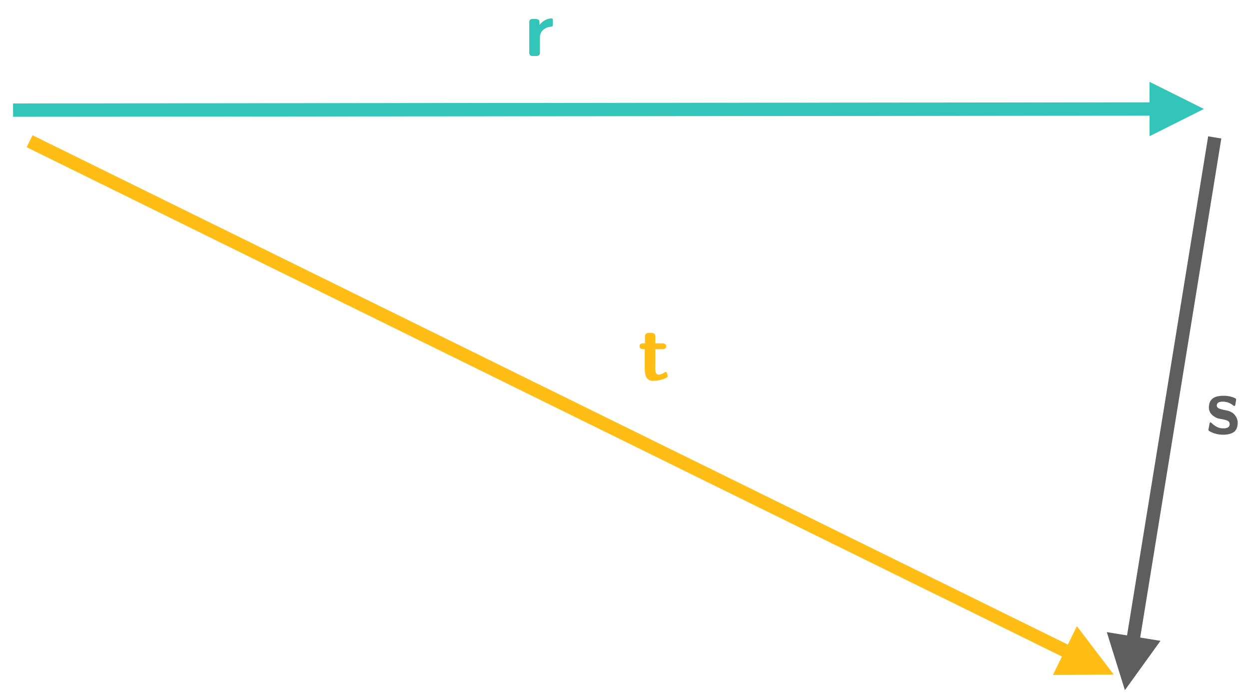
Let us consider the composition pattern: $\forall x, y, z \; r(x,y) \wedge s(y,z) \Rightarrow t(x,z)$. Can we find a model configuration for TransE that captures this pattern?

Which inference patterns can TransE capture?



Let us consider the composition pattern: $\forall x, y, z\ r(x, y) \wedge s(y, z) \Rightarrow t(x, z)$. Can we find a model configuration for TransE that captures this pattern?

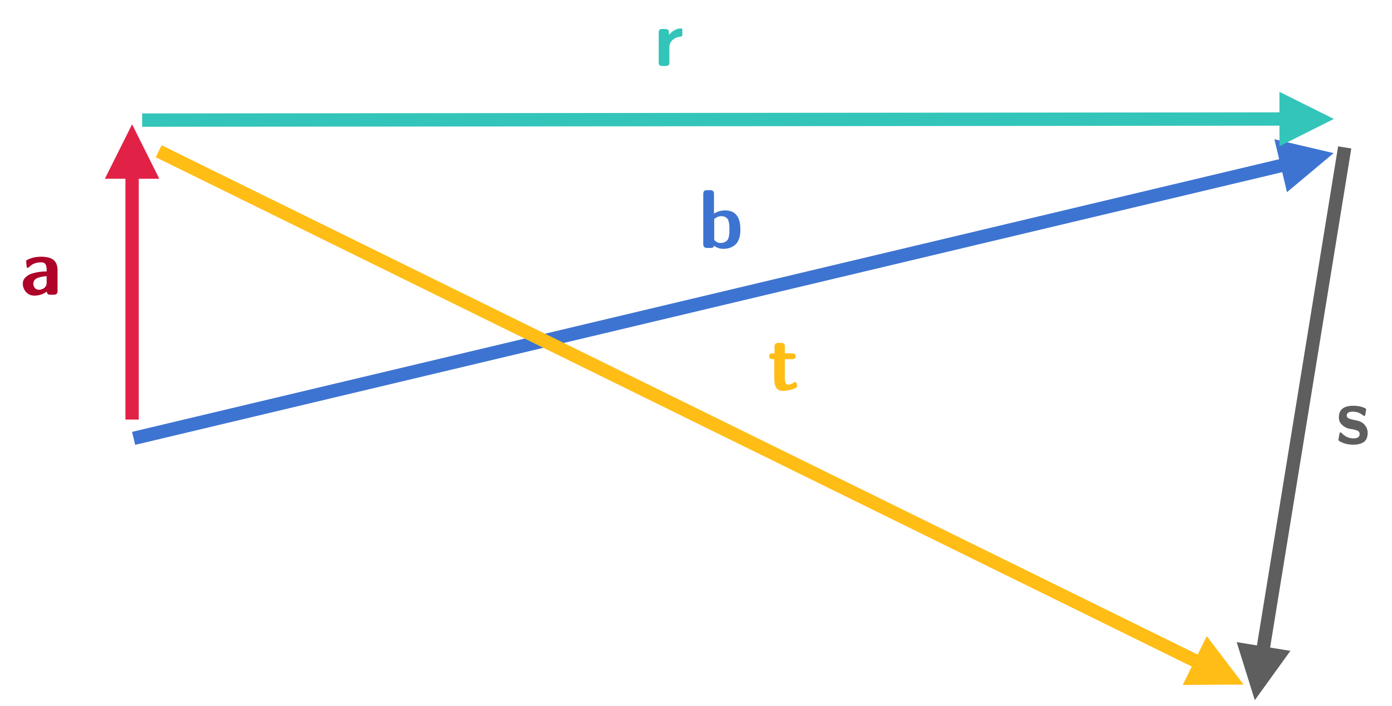
Which inference patterns can TransE capture?



Let us consider the composition pattern: $\forall x, y, z \; r(x, y) \wedge s(y, z) \Rightarrow t(x, z)$. Can we find a model configuration for TransE that captures this pattern?

Consider a model configuration, where $\mathbf{r} + \mathbf{s} \approx \mathbf{t}$. Then, for any entity $a, b, c \in E$, whenever $r(a, b), s(b, c)$ hold, so does $t(a, c)$.

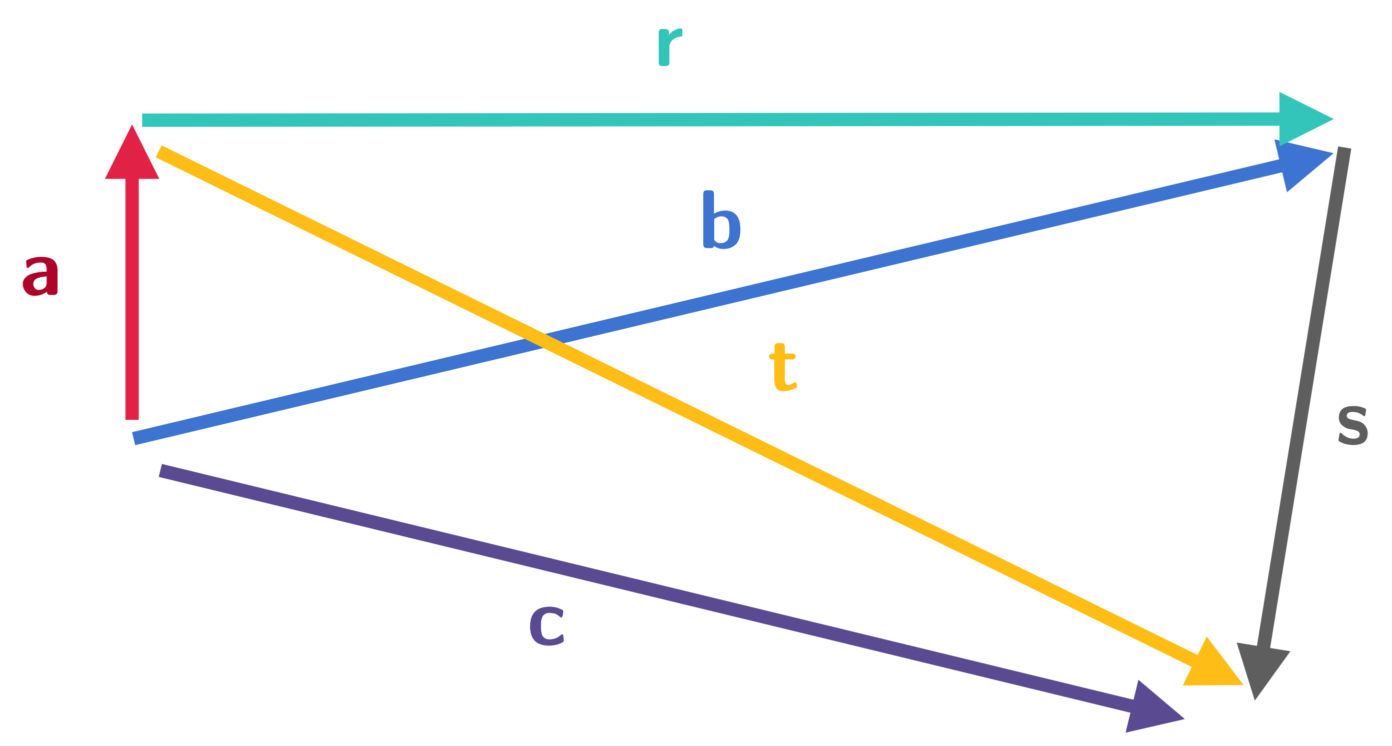
Which inference patterns can TransE capture?



Let us consider the composition pattern: $\forall x, y, z \; r(x, y) \wedge s(y, z) \Rightarrow t(x, z)$. Can we find a model configuration for TransE that captures this pattern?

Consider a model configuration, where $\mathbf{r} + \mathbf{s} \approx \mathbf{t}$. Then, for any entity $a, b, c \in E$, whenever $r(a, b), s(b, c)$ hold, so does $t(a, c)$.

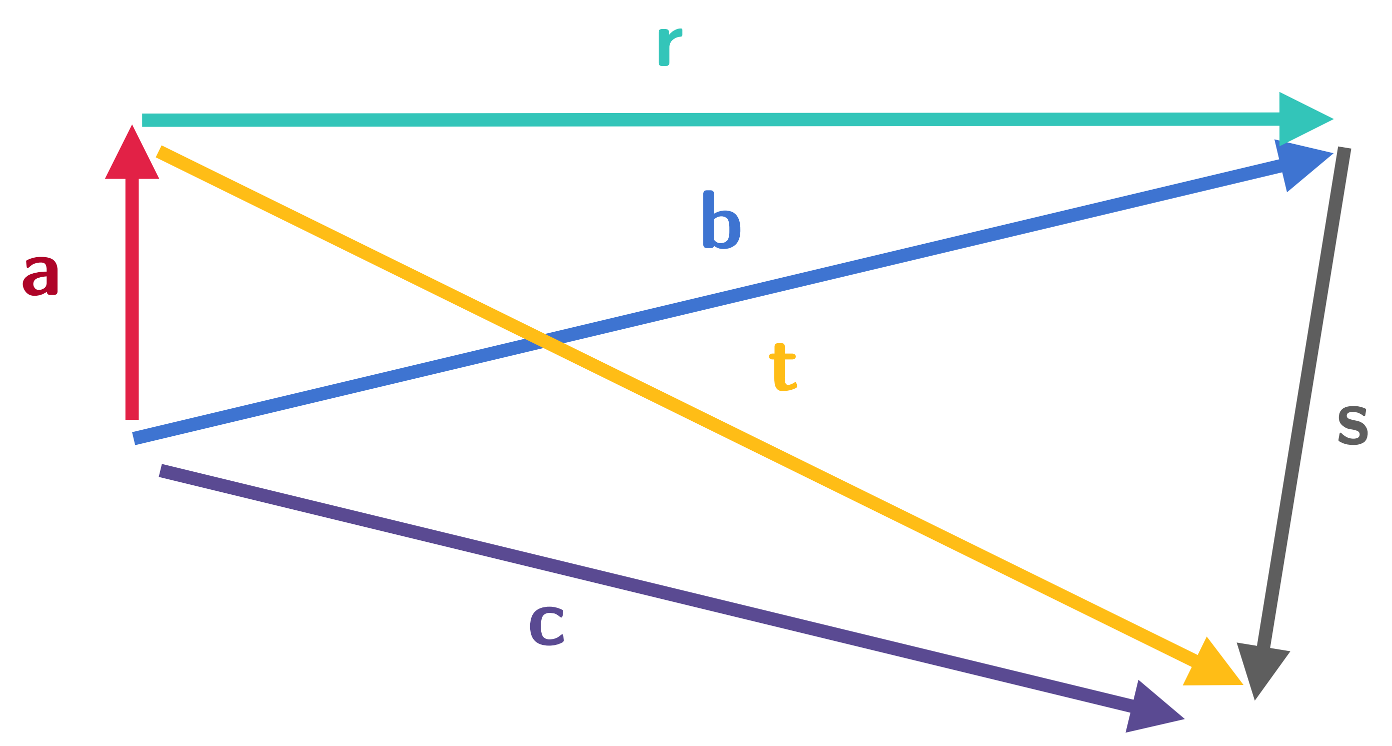
Which inference patterns can TransE capture?



Let us consider the composition pattern: $\forall x, y, z \; r(x, y) \wedge s(y, z) \Rightarrow t(x, z)$. Can we find a model configuration for TransE that captures this pattern?

Consider a model configuration, where $\mathbf{r} + \mathbf{s} \approx \mathbf{t}$. Then, for any entity $a, b, c \in E$, whenever $r(a, b), s(b, c)$ hold, so does $t(a, c)$.

Which inference patterns can TransE capture?

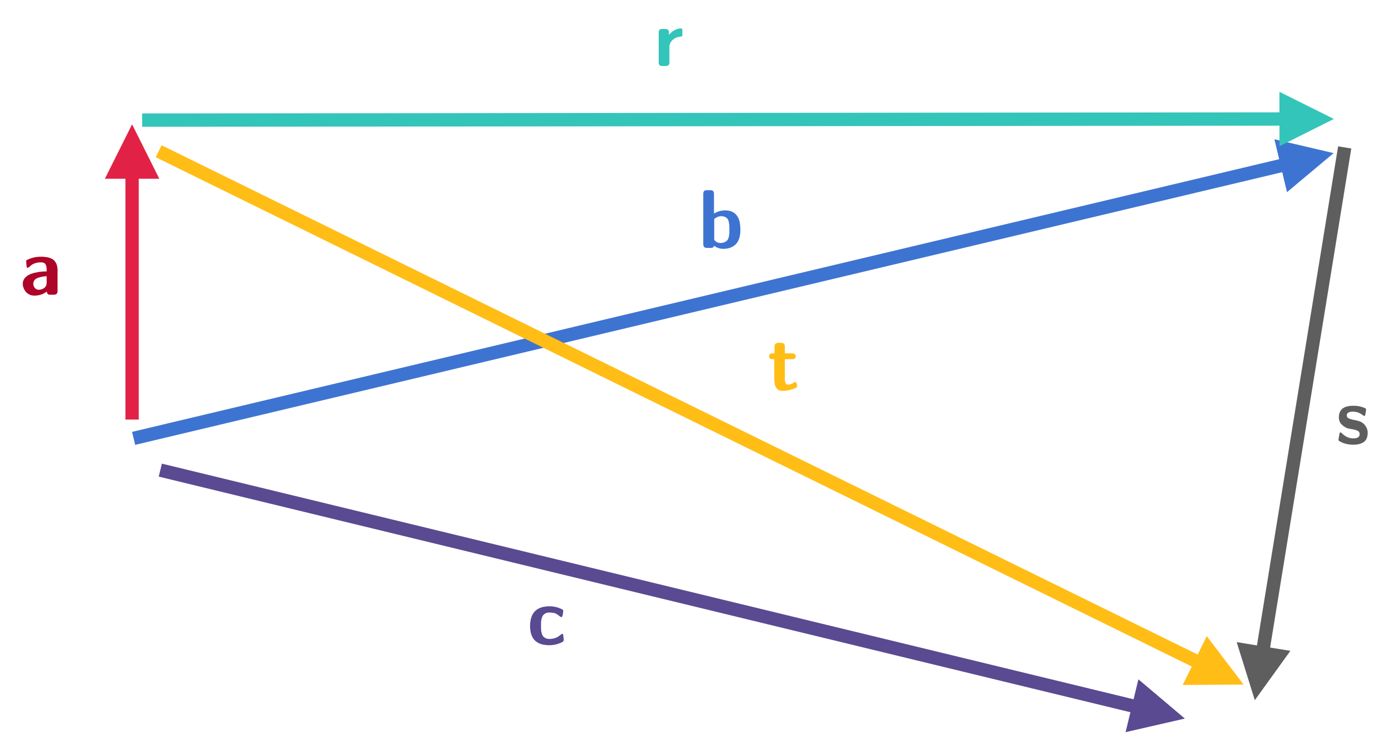


Let us consider the composition pattern: $\forall x, y, z \; r(x, y) \wedge s(y, z) \Rightarrow t(x, z)$. Can we find a model configuration for TransE that captures this pattern?

Consider a model configuration, where $\mathbf{r} + \mathbf{s} \approx \mathbf{t}$. Then, for any entity $a, b, c \in E$, whenever $r(a, b), s(b, c)$ hold, so does $t(a, c)$.

Hence, TransE can capture the composition pattern.

Which inference patterns can TransE capture?



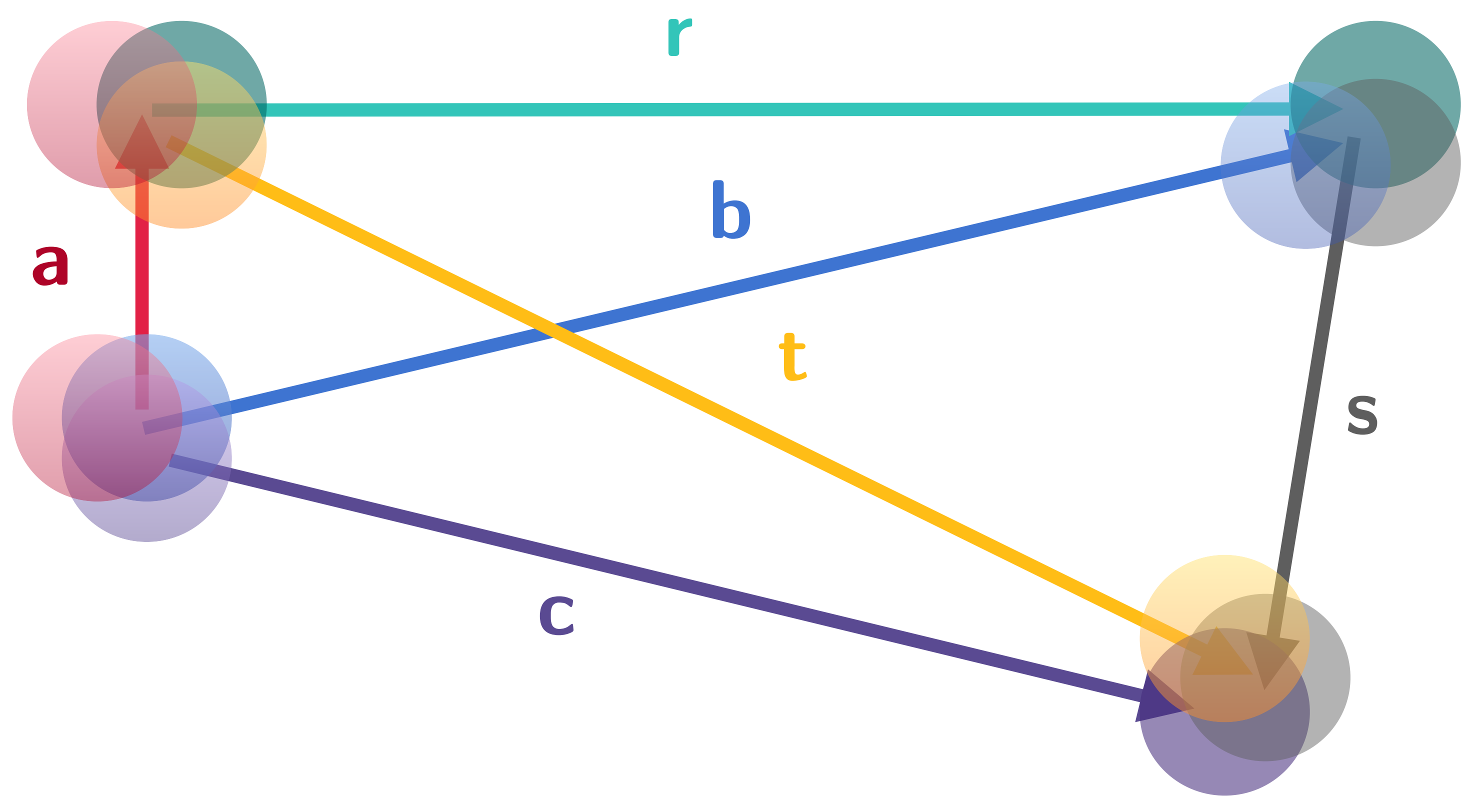
Let us consider the composition pattern: $\forall x, y, z \; r(x, y) \wedge s(y, z) \Rightarrow t(x, z)$. Can we find a model configuration for TransE that captures this pattern?

Consider a model configuration, where $\mathbf{r} + \mathbf{s} \approx \mathbf{t}$. Then, for any entity $a, b, c \in E$, whenever $r(a, b), s(b, c)$ hold, so does $t(a, c)$.

Hence, TransE can capture the composition pattern.

It is easy to see that TransE can also capture, e.g., anti-symmetry and inversion. It can capture intersection only by tweaking the margins.

Which inference patterns can TransE capture?



Let us consider the composition pattern: $\forall x, y, z \; r(x, y) \wedge s(y, z) \Rightarrow t(x, z)$. Can we find a model configuration for TransE that captures this pattern?

Consider a model configuration, where $\mathbf{r} + \mathbf{s} \approx \mathbf{t}$. Then, for any entity $a, b, c \in E$, whenever $r(a, b), s(b, c)$ hold, so does $t(a, c)$.

Hence, TransE can capture the composition pattern.

It is easy to see that TransE can also capture, e.g., anti-symmetry and inversion. It can capture intersection only by tweaking the margins.

Which inference patterns can TransE not capture?

Which inference patterns can TransE not capture?

We have already shown a relation r can be made symmetric in TransE, only by additionally forcing r to be reflexive, so TransE cannot capture symmetry.

Which inference patterns can TransE not capture?

We have already shown a relation r can be made symmetric in TransE, only by additionally forcing r to be reflexive, so TransE cannot capture symmetry.

What about the hierarchy pattern: $\forall x, y \; r(x, y) \Rightarrow s(x, y)$?

Which inference patterns can TransE not capture?

We have already shown a relation r can be made symmetric in TransE, only by additionally forcing r to be reflexive, so TransE cannot capture symmetry.

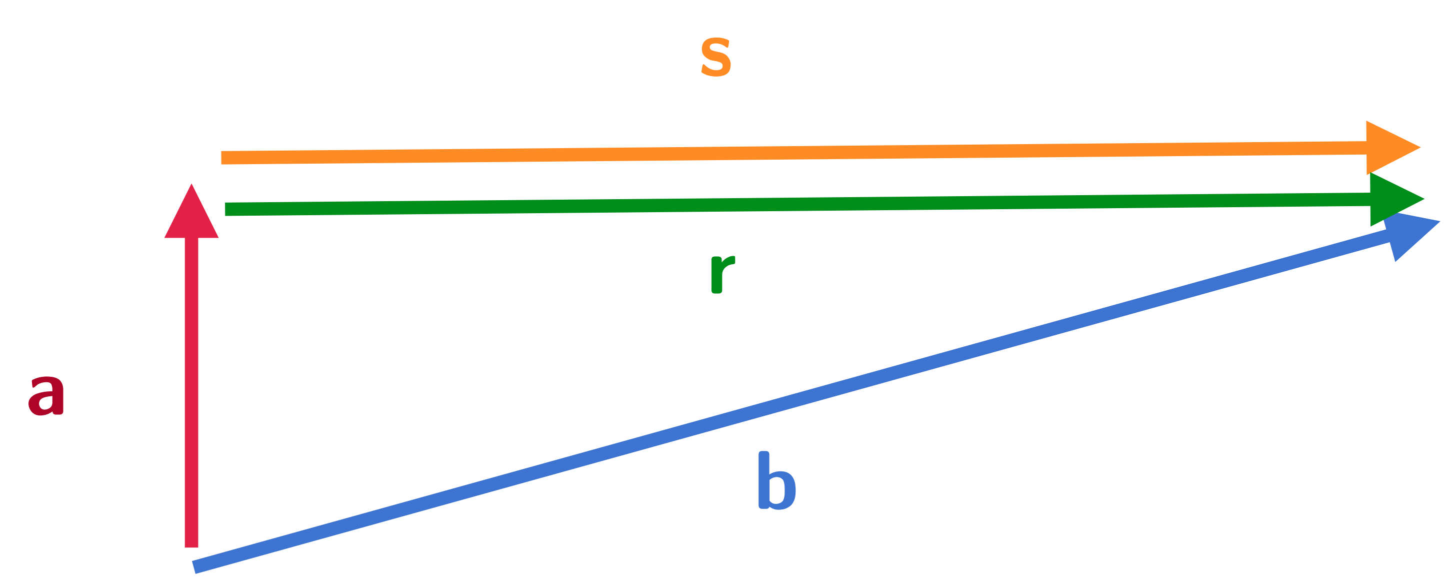
What about the hierarchy pattern: $\forall x, y\ r(x, y) \Rightarrow s(x, y)$?

Observe that this pattern can be realised only by setting $\mathbf{r} \approx \mathbf{s}$, and, this would further imply relation equivalence: TransE cannot capture hierarchy either.

Which inference patterns can TransE not capture?

We have already shown a relation r can be made symmetric in TransE, only by additionally forcing r to be reflexive, so TransE cannot capture symmetry.

What about the hierarchy pattern: $\forall x, y\ r(x, y) \Rightarrow s(x, y)$?

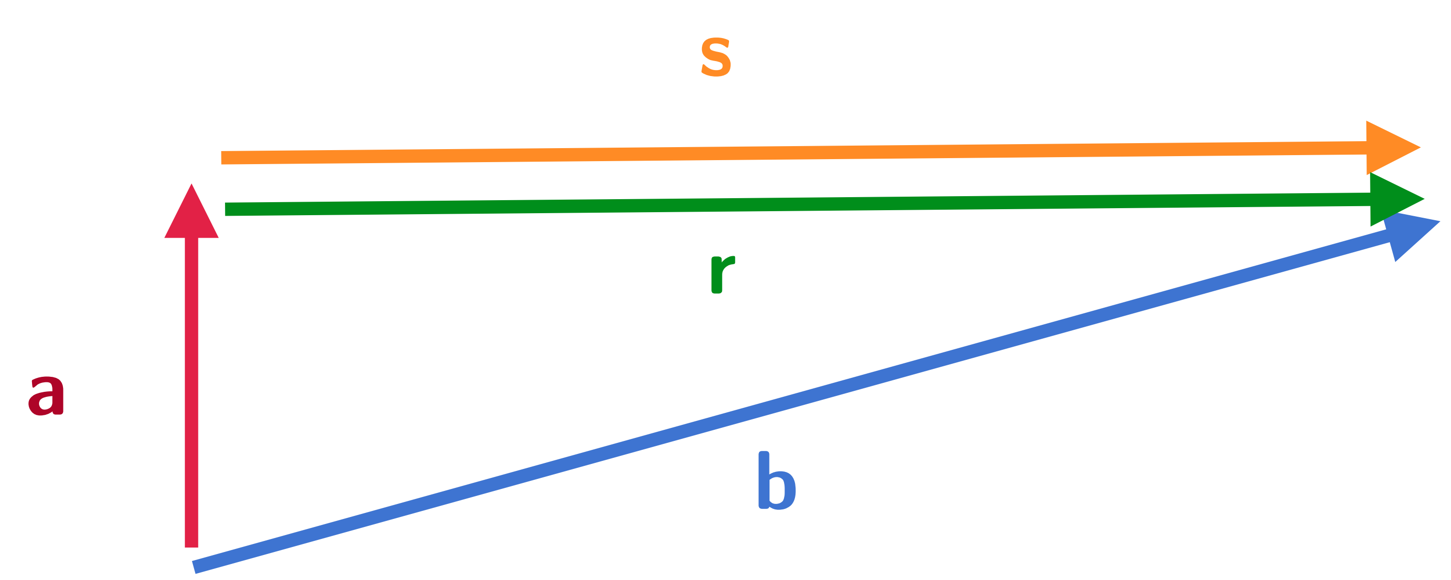


Observe that this pattern can be realised only by setting $\mathbf{r} \approx \mathbf{s}$, and, this would further imply relation equivalence: TransE cannot capture hierarchy either.

Which inference patterns can TransE not capture?

We have already shown a relation r can be made symmetric in TransE, only by additionally forcing r to be reflexive, so TransE cannot capture symmetry.

What about the hierarchy pattern: $\forall x, y\ r(x, y) \Rightarrow s(x, y)$?



Observe that this pattern can be realised only by setting $\mathbf{r} \approx \mathbf{s}$, and, this would further imply relation equivalence: TransE cannot capture hierarchy either.

Similarly to the symmetry pattern, the lack of ability to capture the hierarchy pattern is a serious limitation, as it is also prevalent in datasets (e.g., the relation capitalOf implies the relation cityIn).

Other Limitations of TransE

Other Limitations of TransE

Relations are sometimes referred as 1-to-n, n-to-1, or n-to-n, referring to the cardinality of the relation in terms of the head and tail entities.

Other Limitations of TransE

Relations are sometimes referred as 1-to-n, n-to-1, or n-to-n, referring to the cardinality of the relation in terms of the head and tail entities.

TransE does not efficiently learn the representations for 1-to-n, or n-to-n, relationships in a knowledge graph. This comes from how the scoring function is defined, e.g., consider the facts:

locatedIn(Oxford, Oxfordshire)

locatedIn(Oxford, UK)

Other Limitations of TransE

Relations are sometimes referred as 1-to-n, n-to-1, or n-to-n, referring to the cardinality of the relation in terms of the head and tail entities.

TransE does not efficiently learn the representations for 1-to-n, or n-to-n, relationships in a knowledge graph. This comes from how the scoring function is defined, e.g., consider the facts:

locatedIn(Oxford, Oxfordshire)

locatedIn(Oxford, UK)

The scoring function enforces entities Oxfordshire and UK to be similar, since the other elements locatedIn, Oxford in the function are equivalent.

Other Limitations of TransE

Relations are sometimes referred as 1-to-n, n-to-1, or n-to-n, referring to the cardinality of the relation in terms of the head and tail entities.

TransE does not efficiently learn the representations for 1-to-n, or n-to-n, relationships in a knowledge graph. This comes from how the scoring function is defined, e.g., consider the facts:

locatedIn(Oxford, Oxfordshire)

locatedIn(Oxford, UK)

The scoring function enforces entities Oxfordshire and UK to be similar, since the other elements locatedIn, Oxford in the function are equivalent.

Other translational models are proposed to reduce the effect of this problem; see, e.g., TransH and TransR.

RotatE

RotatE

RotatE is a popular translational model, which defines each relation r as a rotation from an entity h to an entity t in the complex vector space. The main intuition comes from Euler's identity: $e^{i\theta} = cos\theta + i\ sin\theta$, i.e., that a unitary complex number can be regarded as a rotation in the complex plane.

RotatE

RotatE is a popular translational model, which defines each relation r as a rotation from an entity h to an entity t in the complex vector space. The main intuition comes from Euler's identity: $e^{i\theta} = cos\theta + i\ sin\theta$, i.e., that a unitary complex number can be regarded as a rotation in the complex plane.

Representation: RotatE encodes entities $h, t \in E$ and relations $r \in R$, through d-dimensional complex vectors $\mathbf{h}, \mathbf{t}, \mathbf{r} \in \mathbb{C}^d$, where $\mathbf{r}$ corresponds to a rotation with modulus $|\ r_i\ | = 1$ in every dimension i.

RotatE

RotatE is a popular translational model, which defines each relation r as a rotation from an entity h to an entity t in the complex vector space. The main intuition comes from Euler's identity: $e^{i\theta} = cos\theta + i\ sin\theta$, i.e., that a unitary complex number can be regarded as a rotation in the complex plane.

Representation: RotatE encodes entities $h, t \in E$ and relations $r \in R$, through d-dimensional complex vectors $\mathbf{h}, \mathbf{t}, \mathbf{r} \in \mathbb{C}^d$, where $\mathbf{r}$ corresponds to a rotation with modulus $| r_i | = 1$ in every dimension i.

Scoring: RotatE scores a fact $r(h, t)$ in accordance to a distance measure $d(\mathbf{h} \odot \mathbf{r}, \mathbf{t}) = ||\mathbf{h} \odot \mathbf{r} - \mathbf{t}||$, where $\odot$ denotes element-wise product. Then, each element r_i of $\mathbf{r}$ is of the form $e^{i\theta_{r,i}}$, corresponding to a counterclockwise rotation by $\theta^{i\theta_{r,i}}$ radians about the origin of the complex plane.

RotatE

RotatE is a popular translational model, which defines each relation r as a rotation from an entity h to an entity t in the complex vector space. The main intuition comes from Euler's identity: $e^{i\theta} = cos\theta + i\ sin\theta$, i.e., that a unitary complex number can be regarded as a rotation in the complex plane.

Representation: RotatE encodes entities $h, t \in E$ and relations $r \in R$, through d-dimensional complex vectors $\mathbf{h}, \mathbf{t}, \mathbf{r} \in \mathbb{C}^d$, where $\mathbf{r}$ corresponds to a rotation with modulus $|\ r_i\ | = 1$ in every dimension i.

Scoring: RotatE scores a fact $r(h, t)$ in accordance to a distance measure $d(\mathbf{h} \odot \mathbf{r}, \mathbf{t}) = ||\mathbf{h} \odot \mathbf{r} - \mathbf{t}||$, where $\odot$ denotes element-wise product. Then, each element r_i of $\mathbf{r}$ is of the form $e^{i\theta_{r,i}}$, corresponding to a counterclockwise rotation by $\theta^{i\theta_{r,i}}$ radians about the origin of the complex plane.

Loss: For every fact $r(h, t)$, RotatE minimises the following loss function:

RotatE

RotatE is a popular translational model, which defines each relation r as a rotation from an entity h to an entity t in the complex vector space. The main intuition comes from Euler's identity: $e^{i\theta} = cos\theta + i\ sin\theta$, i.e., that a unitary complex number can be regarded as a rotation in the complex plane.

Representation: RotatE encodes entities $h, t \in E$ and relations $r \in R$, through d-dimensional complex vectors $\mathbf{h}, \mathbf{t}, \mathbf{r} \in \mathbb{C}^d$, where $\mathbf{r}$ corresponds to a rotation with modulus $|\ r_i\ | = 1$ in every dimension i.

Scoring: RotatE scores a fact $r(h, t)$ in accordance to a distance measure $d(\mathbf{h} \odot \mathbf{r}, \mathbf{t}) = ||\mathbf{h} \odot \mathbf{r} - \mathbf{t}||$, where $\odot$ denotes element-wise product. Then, each element r_i of $\mathbf{r}$ is of the form $e^{i\theta_{r,i}}$, corresponding to a counterclockwise rotation by $\theta^{i\theta_{r,i}}$ radians about the origin of the complex plane.

Loss: For every fact $r(h, t)$, RotatE minimises the following loss function:

$$\mathscr{L} = -\log \sigma(\gamma - d(\mathbf{h} \odot \mathbf{r}, \mathbf{t})) - \sum_{r(h',t') \in N^{r(h,t)}} \frac{1}{k} \log \sigma(d(\mathbf{h}' \odot \mathbf{r}, \mathbf{t}') - \gamma),$$

RotatE

RotatE is a popular translational model, which defines each relation r as a rotation from an entity h to an entity t in the complex vector space. The main intuition comes from Euler's identity: $e^{i\theta} = cos\theta + i\ sin\theta$, i.e., that a unitary complex number can be regarded as a rotation in the complex plane.

Representation: RotatE encodes entities $h, t \in E$ and relations $r \in R$, through d-dimensional complex vectors $\mathbf{h}, \mathbf{t}, \mathbf{r} \in \mathbb{C}^d$, where $\mathbf{r}$ corresponds to a rotation with modulus $| r_i | = 1$ in every dimension i.

Scoring: RotatE scores a fact $r(h, t)$ in accordance to a distance measure $d(\mathbf{h} \odot \mathbf{r}, \mathbf{t}) = ||\mathbf{h} \odot \mathbf{r} - \mathbf{t}||$, where $\odot$ denotes element-wise product. Then, each element r_i of $\mathbf{r}$ is of the form $e^{i\theta_{r,i}}$, corresponding to a counterclockwise rotation by $\theta^{i\theta_{r,i}}$ radians about the origin of the complex plane.

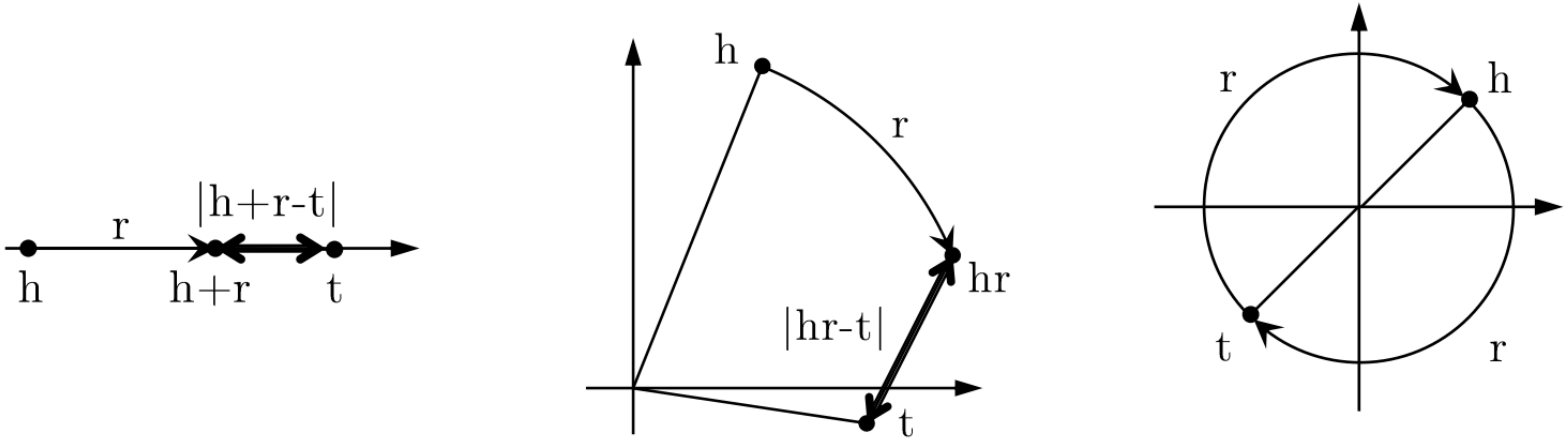
Loss: For every fact $r(h, t)$, RotatE minimises the following loss function:

$$\mathscr{L} = -\log \sigma(\gamma - d(\mathbf{h} \odot \mathbf{r}, \mathbf{t})) - \sum_{r(h',t') \in N^{r(h,t)}} \frac{1}{k} \log \sigma(d(\mathbf{h}' \odot \mathbf{r}, \mathbf{t}') - \gamma),$$

where γ is a fixed margin, σ is the sigmoid function, and $N^{r(h,t)}$ is a set of k negative samples for $r(h, t)$.

RotatE vs TransE

RotatE vs TransE



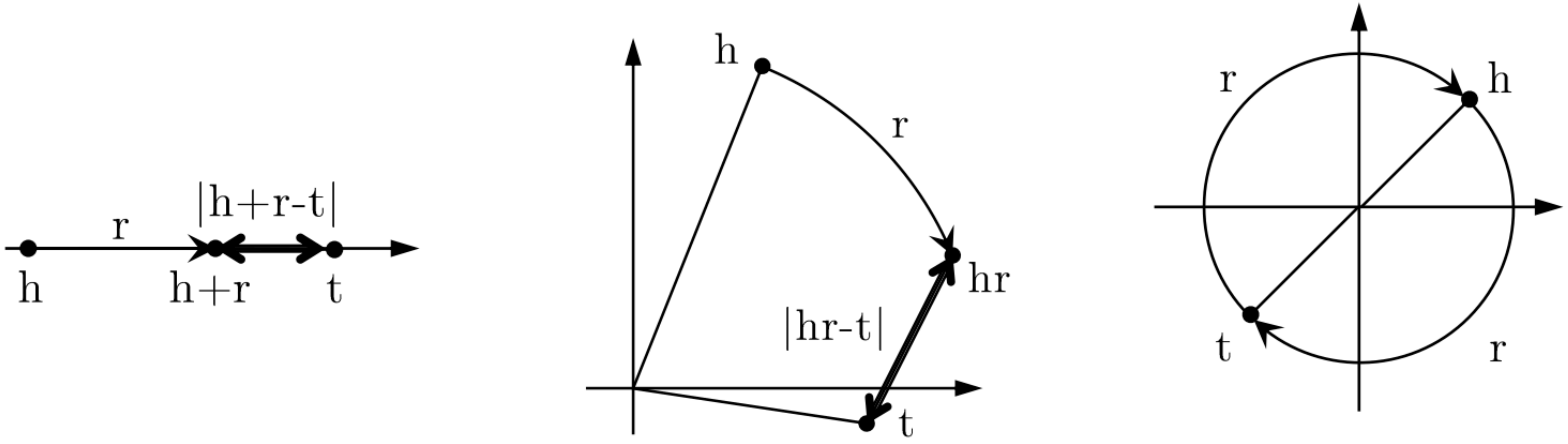
(a) TransE models **r** as translation in real line.

(b) RotatE models **r** as rotation in complex plane.

(c) RotatE: an example of modeling symmetric relations **r** with $r_i = -1$

Figure taken from (Sun et al), showing a comparative 1-dimensional embedding of the models TransE and RotatE. Rotations in each individual dimension enable RotatE to capture symmetry.

RotatE vs TransE



(a) TransE models **r** as translation in real line.

(b) RotatE models **r** as rotation in complex plane.

(c) RotatE: an example of modeling symmetric relations **r** with $r_i = -1$

Figure taken from (Sun et al), showing a comparative 1-dimensional embedding of the models TransE and RotatE. Rotations in each individual dimension enable RotatE to capture symmetry.

Question: Does RotatE capture TransE as a special case?

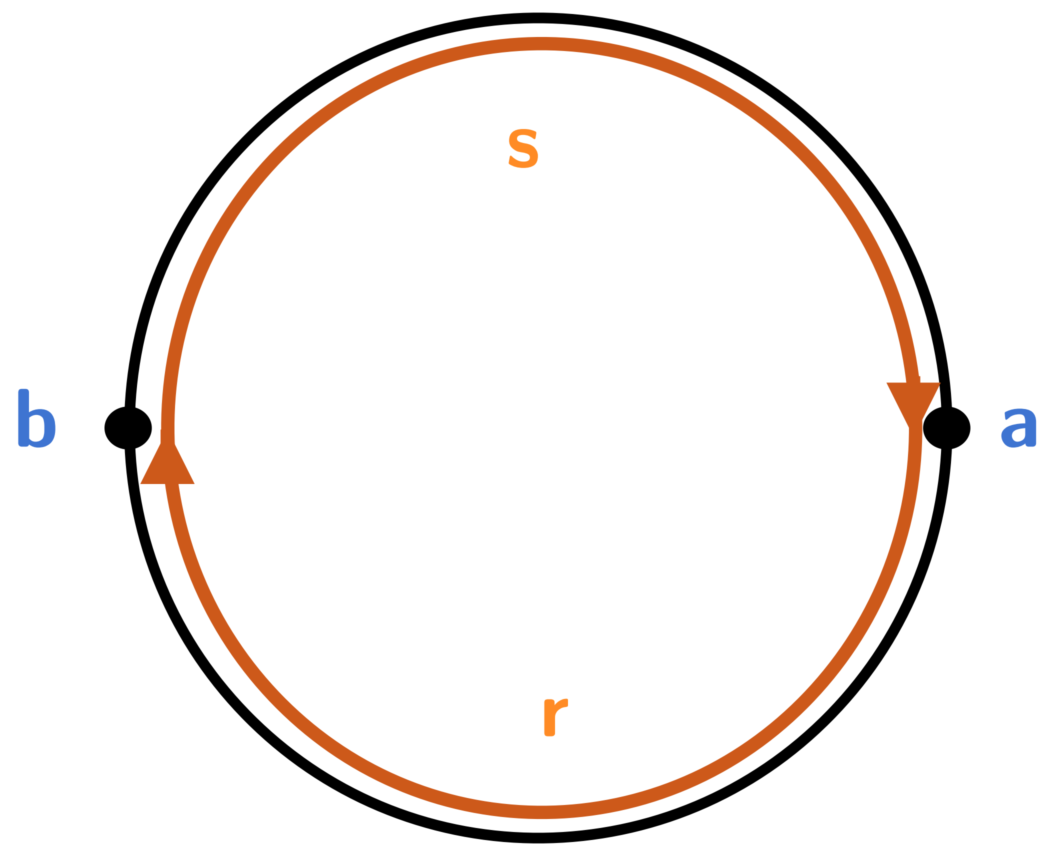
How expressive is RotatE?

How expressive is RotatE?

Consider the set of true facts $T = \{r(a,b), s(b,c), s(b,a)\}$. We can realise the facts$\{r(a,b), s(b,a)\}$ in RotatE by the following configuration:

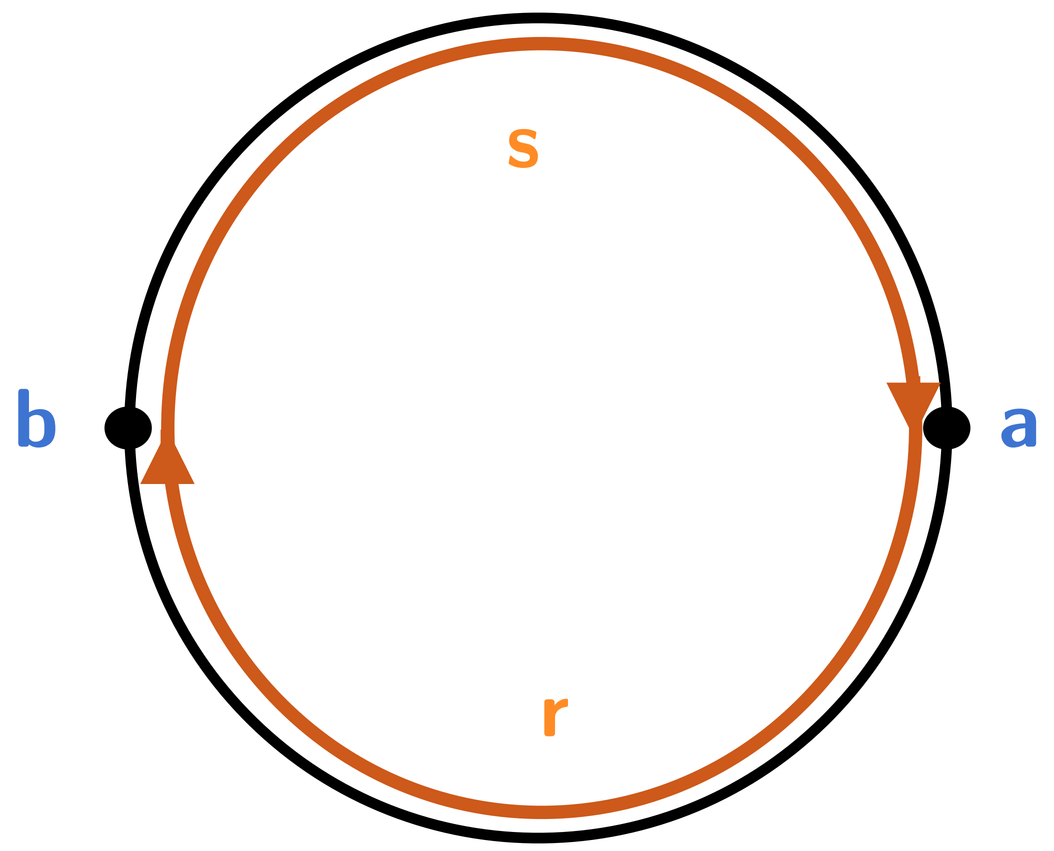
How expressive is RotatE?

Consider the set of true facts $T = \{r(a,b), s(b,c), s(b,a)\}$. We can realise the facts$\{r(a,b), s(b,a)\}$ in RotatE by the following configuration:



How expressive is RotatE?

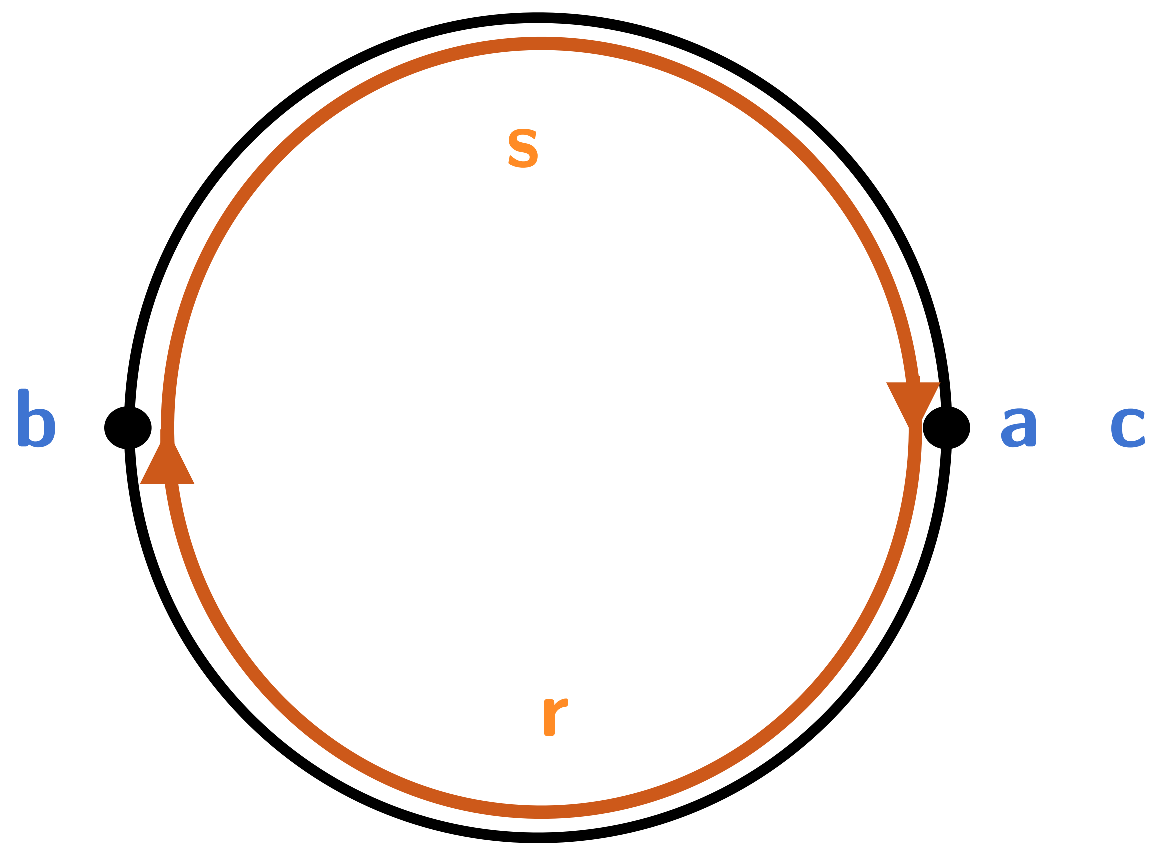
Consider the set of true facts $T = \{r(a, b), s(b, c), s(b, a)\}$. We can realise the facts$\{r(a, b), s(b, a)\}$ in RotatE by the following configuration:



To additionally realise the fact $s(b, c)$, we need $\mathbf{a} \approx \mathbf{c}$. But then this would additionally imply the fact $r(c, b)$ since the rotation $\mathbf{r}$ from $\mathbf{c}$ will result in $\mathbf{b}$.

How expressive is RotatE?

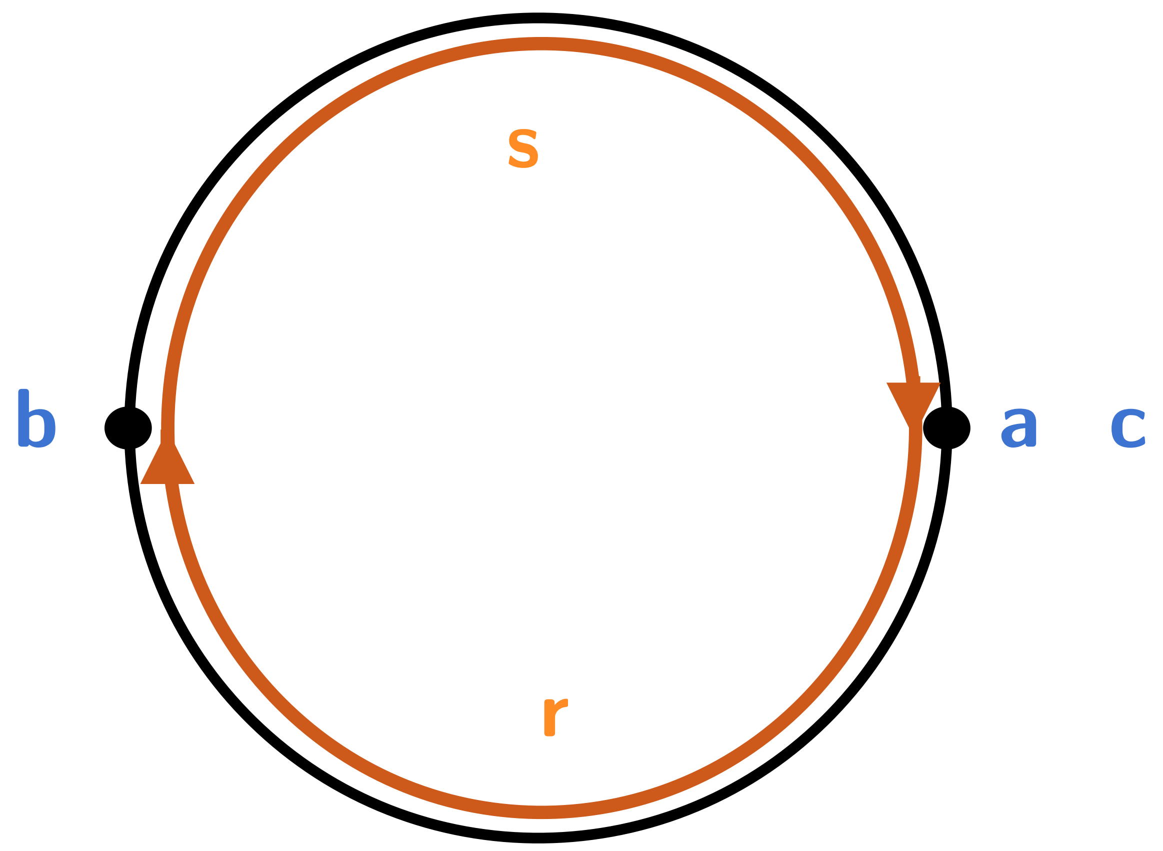
Consider the set of true facts $T = \{r(a,b), s(b,c), s(b,a)\}$. We can realise the facts $\{r(a,b), s(b,a)\}$ in RotatE by the following configuration:



To additionally realise the fact $s(b,c)$, we need $\mathbf{a} \approx \mathbf{c}$. But then this would additionally imply the fact $r(c,b)$ since the rotation $\mathbf{r}$ from $\mathbf{c}$ will result in $\mathbf{b}$.

How expressive is RotatE?

Consider the set of true facts $T = \{r(a,b), s(b,c), s(b,a)\}$. We can realise the facts$\{r(a,b), s(b,a)\}$ in RotatE by the following configuration:



To additionally realise the fact $s(b,c)$, we need $\mathbf{a} \approx \mathbf{c}$. But then this would additionally imply the fact $r(c,b)$ since the rotation $\mathbf{r}$ from $\mathbf{c}$ will result in $\mathbf{b}$.

This observation is not limited to this configuration: RotatE sets $\mathbf{r}$ and $\mathbf{s}$ symmetric to capture the initial two facts, though the relations need not be symmetric. If we consider the set $F = \{r(c,b)\}$ as the set of false facts, and then it is easy to see that RotatE cannot fit these facts simultaneously.

Which inference patterns can RotatE capture?

Which inference patterns can RotatE capture?

It is rather easy to see that all patterns captured by TransE can be captured by RotatE.

Which inference patterns can RotatE capture?

It is rather easy to see that all patterns captured by TransE can be captured by RotatE.

Unlike TransE, RotatE can also capture symmetry, as explained earlier.

Which inference patterns can RotatE capture?

It is rather easy to see that all patterns captured by TransE can be captured by RotatE.

Unlike TransE, RotatE can also capture symmetry, as explained earlier.

What about the hierarchy pattern $\forall x, y \; r(x, y) \Rightarrow s(x, y)$ which is not captured by TransE?

Which inference patterns can RotatE capture?

It is rather easy to see that all patterns captured by TransE can be captured by RotatE.

Unlike TransE, RotatE can also capture symmetry, as explained earlier.

What about the hierarchy pattern $\forall x, y \; r(x, y) \Rightarrow s(x, y)$ which is not captured by TransE?

The hierarchy pattern cannot be captured by RotatE for very similar reasons as TransE.

Which inference patterns can RotatE capture?

It is rather easy to see that all patterns captured by TransE can be captured by RotatE.

Unlike TransE, RotatE can also capture symmetry, as explained earlier.

What about the hierarchy pattern $\forall x, y \; r(x, y) \Rightarrow s(x, y)$ which is not captured by TransE?

The hierarchy pattern cannot be captured by RotatE for very similar reasons as TransE.

To capture facts of the form $r(a, b), s(a, b), \ldots$ we need the rotations from a to b need to be similar, i.e., $\mathbf{r} \approx \mathbf{s}$, effectively enforcing relation equivalence.

Bilinear models

Tensor Representation of a KG

Tensor Representation of a KG

Given a KG G over a relational vocabulary R and E, we can represent G, by defining, for every relation $r \in R$ an adjacency matrix $M_r \in \mathbb{R}^{|E| \times |E|}$:

$$M_{r[i,j]} = \begin{cases} 1 & \text{if } r(e_i, e_j) \in G, \\ 0 & \text{otherwise.} \end{cases}$$

Tensor Representation of a KG

Given a KG G over a relational vocabulary R and E, we can represent G, by defining, for every relation $r \in R$ an adjacency matrix $M_r \in \mathbb{R}^{|E| \times |E|}$:

$$M_{r[i,j]} = \begin{cases} 1 & \text{if } r(e_i, e_j) \in G, \\ 0 & \text{otherwise.} \end{cases}$$

Similarly, we can represent G in terms of a tensor $T \in \mathbb{R}^{|E| \times |E| \times |R|}$:

$$T_{i,j,k} = \begin{cases} 1 & \text{if } r_k(e_i, e_j) \in G, \\ 0 & \text{otherwise.} \end{cases}$$

Tensor Representation of a KG

Given a KG G over a relational vocabulary R and E, we can represent G, by defining, for every relation $r \in R$ an adjacency matrix $M_r \in \mathbb{R}^{|E| \times |E|}$:

$$M_{r[i,j]} = \begin{cases} 1 & \text{if } r(e_i, e_j) \in G, \\ 0 & \text{otherwise.} \end{cases}$$

Similarly, we can represent G in terms of a tensor $T \in \mathbb{R}^{|E| \times |E| \times |R|}$:

$$T_{i,j,k} = \begin{cases} 1 & \text{if } r_k(e_i, e_j) \in G, \\ 0 & \text{otherwise.} \end{cases}$$

Many bilinear models use tensor/matrix representation for relations and so they are also referred as tensor factorisation methods.

Tensor Representation of a KG

Given a KG G over a relational vocabulary R and E, we can represent G, by defining, for every relation $r \in R$ an adjacency matrix $M_r \in \mathbb{R}^{|E| \times |E|}$:

$$M_{r[i,j]} = \begin{cases} 1 & \text{if } r(e_i, e_j) \in G, \\ 0 & \text{otherwise.} \end{cases}$$

Similarly, we can represent G in terms of a tensor $T \in \mathbb{R}^{|E| \times |E| \times |R|}$:

$$T_{i,j,k} = \begin{cases} 1 & \text{if } r_k(e_i, e_j) \in G, \\ 0 & \text{otherwise.} \end{cases}$$

Many bilinear models use tensor/matrix representation for relations and so they are also referred as tensor factorisation methods.

Differently from translational models, bilinear models typically use a multiplicative approach, i.e., a bilinear product, to represent the relationships, hence the name "bilinear".

RESCAL

RESCAL

RESCAL is the first bilinear model and has inspired a line of research.

RESCAL

RESCAL is the first bilinear model and has inspired a line of research.

Representation: RESCAL encodes entities $h, t \in E$ through d-dimensional vectors $\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$, and relations $r \in R$, as a matrix $\mathbf{M}_r \in \mathbb{R}^d \times \mathbb{R}^d$.

RESCAL

RESCAL is the first bilinear model and has inspired a line of research.

Representation: RESCAL encodes entities $h, t \in E$ through d-dimensional vectors $\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$, and relations $r \in R$, as a matrix $\mathbf{M}_r \in \mathbb{R}^d \times \mathbb{R}^d$.

Scoring: RESCAL scores a fact $r(h, t)$ in accordance to the function: $\mathbf{h}^\top \mathbf{M}_r \mathbf{t}$, which captures all pairwise interactions between the components of $\mathbf{h}$ and $\mathbf{t}$.

RESCAL

RESCAL is the first bilinear model and has inspired a line of research.

Representation: RESCAL encodes entities $h, t \in E$ through d-dimensional vectors $\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$, and relations $r \in R$, as a matrix $\mathbf{M}_r \in \mathbb{R}^d \times \mathbb{R}^d$.

Scoring: RESCAL scores a fact $r(h, t)$ in accordance to the function: $\mathbf{h}^\top \mathbf{M}_r \mathbf{t}$, which captures all pairwise interactions between the components of $\mathbf{h}$ and $\mathbf{t}$.

Loss: Exact formulation of the loss function can vary, depending on the considered parameters, e.g., it is possible to extend a loss function by adding regularisation which is common in bilinear models.

RESCAL

RESCAL is the first bilinear model and has inspired a line of research.

Representation: RESCAL encodes entities $h, t \in E$ through d-dimensional vectors $\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$, and relations $r \in R$, as a matrix $\mathbf{M}_r \in \mathbb{R}^d \times \mathbb{R}^d$.

Scoring: RESCAL scores a fact $r(h, t)$ in accordance to the function: $\mathbf{h}^\top \mathbf{M}_r \mathbf{t}$, which captures all pairwise interactions between the components of $\mathbf{h}$ and $\mathbf{t}$.

Loss: Exact formulation of the loss function can vary, depending on the considered parameters, e.g., it is possible to extend a loss function by adding regularisation which is common in bilinear models.

Expressiveness: It is easy to see that RESCAL is fully expressive, as it is possible to fit arbitrary set of true and false facts using the power of full rank matrix. However, this requires $O(d^2)$ parameters per relation, and this is impractical for large-scale KGs.

RESCAL

RESCAL is the first bilinear model and has inspired a line of research.

Representation: RESCAL encodes entities $h, t \in E$ through d-dimensional vectors $\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$, and relations $r \in R$, as a matrix $\mathbf{M}_r \in \mathbb{R}^d \times \mathbb{R}^d$.

Scoring: RESCAL scores a fact $r(h, t)$ in accordance to the function: $\mathbf{h}^\top \mathbf{M}_r \mathbf{t}$, which captures all pairwise interactions between the components of $\mathbf{h}$ and $\mathbf{t}$.

Loss: Exact formulation of the loss function can vary, depending on the considered parameters, e.g., it is possible to extend a loss function by adding regularisation which is common in bilinear models.

Expressiveness: It is easy to see that RESCAL is fully expressive, as it is possible to fit arbitrary set of true and false facts using the power of full rank matrix. However, this requires $O(d^2)$ parameters per relation, and this is impractical for large-scale KGs.

Though expressive, using a full rank matrix is prone to overfitting, and this has motivated a line of research, where several restrictions are imposed on the representation.

DistMult

DistMult

DistMult is a bilinear model that restricts RESCAL to a diagonal matrix instead of a full rank matrix.

DistMult

DistMult is a bilinear model that restricts RESCAL to a diagonal matrix instead of a full rank matrix.

Representation: DistMult does not allow an arbitrary matrix $\mathbf{M}_r \in \mathbb{R}^d \times \mathbb{R}^d$ for a relation $r \in R$ and restricts this to be the diagonal matrix $\mathbf{D}_r$.

DistMult

DistMult is a bilinear model that restricts RESCAL to a diagonal matrix instead of a full rank matrix.

Representation: DistMult does not allow an arbitrary matrix $\mathbf{M}_r \in \mathbb{R}^d \times \mathbb{R}^d$ for a relation $r \in R$ and restricts this to be the diagonal matrix $\mathbf{D}_r$.

Scoring: DistMult scores a fact $r(h, t)$ similar to RESCAL, with the restriction to the diagonal matrix: $\mathbf{h}^\top \mathbf{D}_r \mathbf{t}$.

DistMult

DistMult is a bilinear model that restricts RESCAL to a diagonal matrix instead of a full rank matrix.

Representation: DistMult does not allow an arbitrary matrix $\mathbf{M}_r \in \mathbb{R}^d \times \mathbb{R}^d$ for a relation $r \in R$ and restricts this to be the diagonal matrix $\mathbf{D}_r$.

Scoring: DistMult scores a fact $r(h, t)$ similar to RESCAL, with the restriction to the diagonal matrix: $\mathbf{h}^\top \mathbf{D}_r \mathbf{t}$.

DistMult cannot capture all pairwise interactions between the components of $\mathbf{h}$ and $\mathbf{t}$ any more. Why?

DistMult

DistMult is a bilinear model that restricts RESCAL to a diagonal matrix instead of a full rank matrix.

Representation: DistMult does not allow an arbitrary matrix $\mathbf{M}_r \in \mathbb{R}^d \times \mathbb{R}^d$ for a relation $r \in R$ and restricts this to be the diagonal matrix $\mathbf{D}_r$.

Scoring: DistMult scores a fact $r(h, t)$ similar to RESCAL, with the restriction to the diagonal matrix: $\mathbf{h}^\top \mathbf{D}_r \mathbf{t}$.

DistMult cannot capture all pairwise interactions between the components of $\mathbf{h}$ and $\mathbf{t}$ any more. Why?

DistMult cannot differentiate between head entity and tail entity since $\mathbf{h}^\top \mathbf{D}_r \mathbf{t} = \mathbf{t}^\top \mathbf{D}_r \mathbf{h}$. This means that all relations are modelled as symmetric regardless, i.e., even anti-symmetric relations will be represented as symmetric.

DistMult

DistMult is a bilinear model that restricts RESCAL to a diagonal matrix instead of a full rank matrix.

Representation: DistMult does not allow an arbitrary matrix $\mathbf{M}_r \in \mathbb{R}^d \times \mathbb{R}^d$ for a relation $r \in R$ and restricts this to be the diagonal matrix $\mathbf{D}_r$.

Scoring: DistMult scores a fact $r(h, t)$ similar to RESCAL, with the restriction to the diagonal matrix: $\mathbf{h}^\top \mathbf{D}_r \mathbf{t}$.

DistMult cannot capture all pairwise interactions between the components of $\mathbf{h}$ and $\mathbf{t}$ any more. Why?

DistMult cannot differentiate between head entity and tail entity since $\mathbf{h}^\top \mathbf{D}_r \mathbf{t} = \mathbf{t}^\top \mathbf{D}_r \mathbf{h}$. This means that all relations are modelled as symmetric regardless, i.e., even anti-symmetric relations will be represented as symmetric.

Expressiveness: DistMult is not fully expressive, i.e., clearly underfitting any dataset with facts from an asymmetric relation.

DistMult

DistMult is a bilinear model that restricts RESCAL to a diagonal matrix instead of a full rank matrix.

Representation: DistMult does not allow an arbitrary matrix $\mathbf{M}_r \in \mathbb{R}^d \times \mathbb{R}^d$ for a relation $r \in R$ and restricts this to be the diagonal matrix $\mathbf{D}_r$.

Scoring: DistMult scores a fact $r(h, t)$ similar to RESCAL, with the restriction to the diagonal matrix: $\mathbf{h}^\top \mathbf{D}_r \mathbf{t}$.

DistMult cannot capture all pairwise interactions between the components of $\mathbf{h}$ and $\mathbf{t}$ any more. Why?

DistMult cannot differentiate between head entity and tail entity since $\mathbf{h}^\top \mathbf{D}_r \mathbf{t} = \mathbf{t}^\top \mathbf{D}_r \mathbf{h}$. This means that all relations are modelled as symmetric regardless, i.e., even anti-symmetric relations will be represented as symmetric.

Expressiveness: DistMult is not fully expressive, i.e., clearly underfitting any dataset with facts from an asymmetric relation.

While very inexpressive, DistMult is scalable, i.e., linear in d.

ComplEx

ComplEx

ComplEx is another bilinear model which extends DistMult to the complex domain.

ComplEx

ComplEx is another bilinear model which extends DistMult to the complex domain.

Representation: ComplEx encodes entities $h, t \in E$ through d-dimensional vectors $\mathbf{h}, \mathbf{t} \in \mathbb{C}^d$ in complex space, and relations $r \in R$, as a diagonal matrix $\mathbf{D}_r \in \mathbb{C}^d \times \mathbb{C}^d$ in this space.

ComplEx

ComplEx is another bilinear model which extends DistMult to the complex domain.

Representation: ComplEx encodes entities $h, t \in E$ through d-dimensional vectors $\mathbf{h}, \mathbf{t} \in \mathbb{C}^d$ in complex space, and relations $r \in R$, as a diagonal matrix $\mathbf{D}_r \in \mathbb{C}^d \times \mathbb{C}^d$ in this space.

Scoring: ComplEx scores a fact $r(h, t)$ as $\text{Re}(\mathbf{h}^\top \mathbf{D}_r \bar{\mathbf{t}})$, where $\bar{\mathbf{t}}$ defines the complex conjugate of $\mathbf{t}$, and Re denotes the real part of a complex vector.

ComplEx

ComplEx is another bilinear model which extends DistMult to the complex domain.

Representation: ComplEx encodes entities $h, t \in E$ through d-dimensional vectors $\mathbf{h}, \mathbf{t} \in \mathbb{C}^d$ in complex space, and relations $r \in R$, as a diagonal matrix $\mathbf{D}_r \in \mathbb{C}^d \times \mathbb{C}^d$ in this space.

Scoring: ComplEx scores a fact $r(h, t)$ as $\text{Re}(\mathbf{h}^\top \mathbf{D}_r \bar{\mathbf{t}})$, where $\bar{\mathbf{t}}$ defines the complex conjugate of $\mathbf{t}$, and Re denotes the real part of a complex vector.

Expressiveness: Intuitively, by modelling head and tail entity embeddings for the same entity as complex conjugates, ComplEx introduces asymmetry and thus can also model asymmetric relations. ComplEx is fully expressive for KGs.

ComplEx

ComplEx is another bilinear model which extends DistMult to the complex domain.

Representation: ComplEx encodes entities $h, t \in E$ through d-dimensional vectors $\mathbf{h}, \mathbf{t} \in \mathbb{C}^d$ in complex space, and relations $r \in R$, as a diagonal matrix $\mathbf{D}_r \in \mathbb{C}^d \times \mathbb{C}^d$ in this space.

Scoring: ComplEx scores a fact $r(h, t)$ as $\text{Re}(\mathbf{h}^\top \mathbf{D}_r \bar{\mathbf{t}})$, where $\bar{\mathbf{t}}$ defines the complex conjugate of $\mathbf{t}$, and Re denotes the real part of a complex vector.

Expressiveness: Intuitively, by modelling head and tail entity embeddings for the same entity as complex conjugates, ComplEx introduces asymmetry and thus can also model asymmetric relations. ComplEx is fully expressive for KGs.

ComplEx is an interesting trade-off, as it generalises DistMult to a fully expressive model, while still using diagonal matrices, which are less prone to overfitting.

What Inference Patterns can Bilinear Models capture?

What Inference Patterns can Bilinear Models capture?

DistMult is inherently symmetric and ComplEx can also capture symmetry.

What Inference Patterns can Bilinear Models capture?

DistMult is inherently symmetric and ComplEx can also capture symmetry.

ComplEx can additionally capture anti-symmetry and inversion with the help of complex conjugates.

What Inference Patterns can Bilinear Models capture?

DistMult is inherently symmetric and ComplEx can also capture symmetry.

ComplEx can additionally capture anti-symmetry and inversion with the help of complex conjugates.

Neither model can capture composition (or intersection): The main reason is that the scoring functions described by ComplEx or by DistMult are not injective, and injectivity is a necessary condition for capturing composition (Sun et al, 2019).

What Inference Patterns can Bilinear Models capture?

DistMult is inherently symmetric and ComplEx can also capture symmetry.

ComplEx can additionally capture anti-symmetry and inversion with the help of complex conjugates.

Neither model can capture composition (or intersection): The main reason is that the scoring functions described by ComplEx or by DistMult are not injective, and injectivity is a necessary condition for capturing composition (Sun et al, 2019).

Both ComplEx and DistMult can capture the hierarchy pattern: For DistMult, simply define the relation r as a scalar multiplication of a relation s, e.g., for $\lambda > 1$, set $s = \lambda r$. Then, any $\mathbf{h}^\top \mathbf{D_r t}$ implies $\mathbf{h}^\top \mathbf{D_s t}$, and and hence $\forall x, y \; r(x, y) \Rightarrow s(x, y)$. The argument for ComplEx is analogous.

What Inference Patterns can Bilinear Models capture?

DistMult is inherently symmetric and ComplEx can also capture symmetry.

ComplEx can additionally capture anti-symmetry and inversion with the help of complex conjugates.

Neither model can capture composition (or intersection): The main reason is that the scoring functions described by ComplEx or by DistMult are not injective, and injectivity is a necessary condition for capturing composition (Sun et al, 2019).

Both ComplEx and DistMult can capture the hierarchy pattern: For DistMult, simply define the relation r as a scalar multiplication of a relation s, e.g., for $\lambda > 1$, set $s = \lambda r$. Then, any $\mathbf{h}^\top \mathbf{D_r t}$ implies $\mathbf{h}^\top \mathbf{D_s t}$, and and hence $\forall x, y \; r(x, y) \Rightarrow s(x, y)$. The argument for ComplEx is analogous.

Note that this does not mean that bilinear models can capture relational hierarchies, i.e., it only means that one instance of such rule can be captured. Hierarchies captured in bilinear models are inherently linear, and this is an important limitation as we shall see later.

Box embedding models

Box Embeddings

Box Embeddings

Box Embeddings are first used in the context of entity classification (Vilnis et al.), where a probabilistic embedding model of KGs is proposed based on Box Lattice Measures.

Box Embeddings

Box Embeddings are first used in the context of entity classification (Vilnis et al.), where a probabilistic embedding model of KGs is proposed based on Box Lattice Measures.

Basic Idea: Every concept (i.e., unary relation) and entity in a KG are represented by a box. In this setup, entity class membership, as well as relation similarity, is captured by means of box intersection in the lattice representation space. For instance, Oxford being a City is captured by 2 boxes, one for the Oxford entity and another for the city class, such that the Oxford box fits entirely into the City box.

Box Embeddings

Box Embeddings are first used in the context of entity classification (Vilnis et al.), where a probabilistic embedding model of KGs is proposed based on Box Lattice Measures.

Basic Idea: Every concept (i.e., unary relation) and entity in a KG are represented by a box. In this setup, entity class membership, as well as relation similarity, is captured by means of box intersection in the lattice representation space. For instance, Oxford being a City is captured by 2 boxes, one for the Oxford entity and another for the city class, such that the Oxford box fits entirely into the City box.

Beyond entity classification: This representation is tailored towards entity classification and so is limited to classes, e.g., to unary relations. The model does not naturally scale to capture interactions between entities, which makes it difficult to apply in the KG completion setting.

Box Embeddings

Box Embeddings are first used in the context of entity classification (Vilnis et al.), where a probabilistic embedding model of KGs is proposed based on Box Lattice Measures.

Basic Idea: Every concept (i.e., unary relation) and entity in a KG are represented by a box. In this setup, entity class membership, as well as relation similarity, is captured by means of box intersection in the lattice representation space. For instance, Oxford being a City is captured by 2 boxes, one for the Oxford entity and another for the city class, such that the Oxford box fits entirely into the City box.

Beyond entity classification: This representation is tailored towards entity classification and so is limited to classes, e.g., to unary relations. The model does not naturally scale to capture interactions between entities, which makes it difficult to apply in the KG completion setting.

A naive solution to this problem is to represent entity pairs, as well as binary relations with boxes, but this results in a quadratic blow up in the representation space, and destroys any parameter sharing, which negatively affects learning.

Box Embeddings

Box Embeddings are first used in the context of entity classification (Vilnis et al.), where a probabilistic embedding model of KGs is proposed based on Box Lattice Measures.

Basic Idea: Every concept (i.e., unary relation) and entity in a KG are represented by a box. In this setup, entity class membership, as well as relation similarity, is captured by means of box intersection in the lattice representation space. For instance, Oxford being a City is captured by 2 boxes, one for the Oxford entity and another for the city class, such that the Oxford box fits entirely into the City box.

Beyond entity classification: This representation is tailored towards entity classification and so is limited to classes, e.g., to unary relations. The model does not naturally scale to capture interactions between entities, which makes it difficult to apply in the KG completion setting.

A naive solution to this problem is to represent entity pairs, as well as binary relations with boxes, but this results in a quadratic blow up in the representation space, and destroys any parameter sharing, which negatively affects learning.

Box embeddings have also been used for the task of query answering, see, e.g., Query2Box (Ren et al.).

Box Embeddings

Box Embeddings are first used in the context of entity classification (Vilnis et al.), where a probabilistic embedding model of KGs is proposed based on Box Lattice Measures.

Basic Idea: Every concept (i.e., unary relation) and entity in a KG are represented by a box. In this setup, entity class membership, as well as relation similarity, is captured by means of box intersection in the lattice representation space. For instance, Oxford being a City is captured by 2 boxes, one for the Oxford entity and another for the city class, such that the Oxford box fits entirely into the City box.

Beyond entity classification: This representation is tailored towards entity classification and so is limited to classes, e.g., to unary relations. The model does not naturally scale to capture interactions between entities, which makes it difficult to apply in the KG completion setting.

A naive solution to this problem is to represent entity pairs, as well as binary relations with boxes, but this results in a quadratic blow up in the representation space, and destroys any parameter sharing, which negatively affects learning.

Box embeddings have also been used for the task of query answering, see, e.g., Query2Box (Ren et al.).

Can box embeddings be used for knowledge graph completion?

BoxE: Model Representation

BoxE: Model Representation

BoxE is a KG embedding model which uses boxes to represent relations. BoxE is a general model that applies to arbitrary knowledge bases, not necessarily those in the form of KGs, as it can handle higher-arity facts beyond binary. We restrict our attention to KGs, for ease of presentation.

BoxE: Model Representation

BoxE is a KG embedding model which uses boxes to represent relations. BoxE is a general model that applies to arbitrary knowledge bases, not necessarily those in the form of KGs, as it can handle higher-arity facts beyond binary. We restrict our attention to KGs, for ease of presentation.

Representation: BoxE encodes each entity $h, t \in E$ in terms of two d-dimensional vectors $\mathbf{h}, \mathbf{b_h} \in \mathbb{R}^d$ and $\mathbf{t}, \mathbf{b_t} \in \mathbb{R}^d$, respectively. The embedding $\mathbf{h}$ (resp., $\mathbf{t}$) defines the base position of an entity h (resp., t), and the embedding $\mathbf{b_h}$ (resp., $\mathbf{b_t}$) defines its translational bump, which translates other entities from their base positions to their final embeddings by "bumping" them.

BoxE: Model Representation

BoxE is a KG embedding model which uses boxes to represent relations. BoxE is a general model that applies to arbitrary knowledge bases, not necessarily those in the form of KGs, as it can handle higher-arity facts beyond binary. We restrict our attention to KGs, for ease of presentation.

Representation: BoxE encodes each entity $h, t \in E$ in terms of two d-dimensional vectors $\mathbf{h}, \mathbf{b_h} \in \mathbb{R}^d$ and $\mathbf{t}, \mathbf{b_t} \in \mathbb{R}^d$, respectively. The embedding $\mathbf{h}$ (resp., $\mathbf{t}$) defines the base position of an entity h (resp., t), and the embedding $\mathbf{b_h}$ (resp., $\mathbf{b_t}$) defines its translational bump, which translates other entities from their base positions to their final embeddings by "bumping" them.

The final embedding of a head entity h relative to a fact $r(h, t)$ is given by: $\mathbf{h}^{\mathbf{r(h,t)}} = \mathbf{h} + \mathbf{b_t}$. Similarly, the final embedding of a tail entity t relative to a fact $r(h, t)$ is given by: $\mathbf{t}^{\mathbf{r(h,t)}} = \mathbf{t} + \mathbf{b_h}$.

BoxE: Model Representation

BoxE is a KG embedding model which uses boxes to represent relations. BoxE is a general model that applies to arbitrary knowledge bases, not necessarily those in the form of KGs, as it can handle higher-arity facts beyond binary. We restrict our attention to KGs, for ease of presentation.

Representation: BoxE encodes each entity $h, t \in E$ in terms of two d-dimensional vectors $\mathbf{h}, \mathbf{b_h} \in \mathbb{R}^d$ and $\mathbf{t}, \mathbf{b_t} \in \mathbb{R}^d$, respectively. The embedding $\mathbf{h}$ (resp., $\mathbf{t}$) defines the base position of an entity h (resp., t), and the embedding $\mathbf{b_h}$ (resp., $\mathbf{b_t}$) defines its translational bump, which translates other entities from their base positions to their final embeddings by "bumping" them.

The final embedding of a head entity h relative to a fact $r(h,t)$ is given by: $\mathbf{h}^{\mathbf{r(h,t)}} = \mathbf{h} + \mathbf{b_t}$. Similarly, the final embedding of a tail entity t relative to a fact $r(h,t)$ is given by: $\mathbf{t}^{\mathbf{r(h,t)}} = \mathbf{t} + \mathbf{b_h}$.

In BoxE, a (binary) relation $r \in R$, is represented in terms of two d-dimensional hyper-rectangles, or boxes, $\mathbf{r^h}, \mathbf{r^t} \in \mathbb{R}^d$, corresponding to a head box and a tail box, respectively.

BoxE: Scoring and Spatial Properties

BoxE: Scoring and Spatial Properties

Scoring: BoxE defines a distance function that determines how close a head entity is to a head box, and similarly, how close a tail entity is to a tail box. BoxE scores a fact $r(h,t)$ as the sum of the L-x norms of such function:

$$\left\| \text{dist}(\mathbf{h}^{\mathbf{r}(\mathbf{h},\mathbf{t})}, \mathbf{r}^{\mathbf{h}}) \right\|_x + \left\| \text{dist}(\mathbf{t}^{\mathbf{r}(\mathbf{h},\mathbf{t})}, \mathbf{r}^{\mathbf{t}}) \right\|_x,$$

where dist is a distance function that grows slowly if a point is in the box (relative to the centre of the box), but grows rapidly if the point is outside of the box, so as to drive points more effectively into their target boxes and ensure they are minimally changed, and can remain there once inside.

BoxE: Scoring and Spatial Properties

Scoring: BoxE defines a distance function that determines how close a head entity is to a head box, and similarly, how close a tail entity is to a tail box. BoxE scores a fact $r(h, t)$ as the sum of the L-x norms of such function:

$$\left\| \text{dist}(\mathbf{h}^{\mathbf{r}(\mathbf{h},\mathbf{t})}, \mathbf{r}^{\mathbf{h}}) \right\|_x + \left\| \text{dist}(\mathbf{t}^{\mathbf{r}(\mathbf{h},\mathbf{t})}, \mathbf{r}^{\mathbf{t}}) \right\|_x,$$

where dist is a distance function that grows slowly if a point is in the box (relative to the centre of the box), but grows rapidly if the point is outside of the box, so as to drive points more effectively into their target boxes and ensure they are minimally changed, and can remain there once inside.

Box sizes are dynamic and their position matters: Every relation may be represented with boxes of different size and their relative position in relation to entities are part of scoring. Hence, BoxE can be seen as a hybrid spatio-translational model.

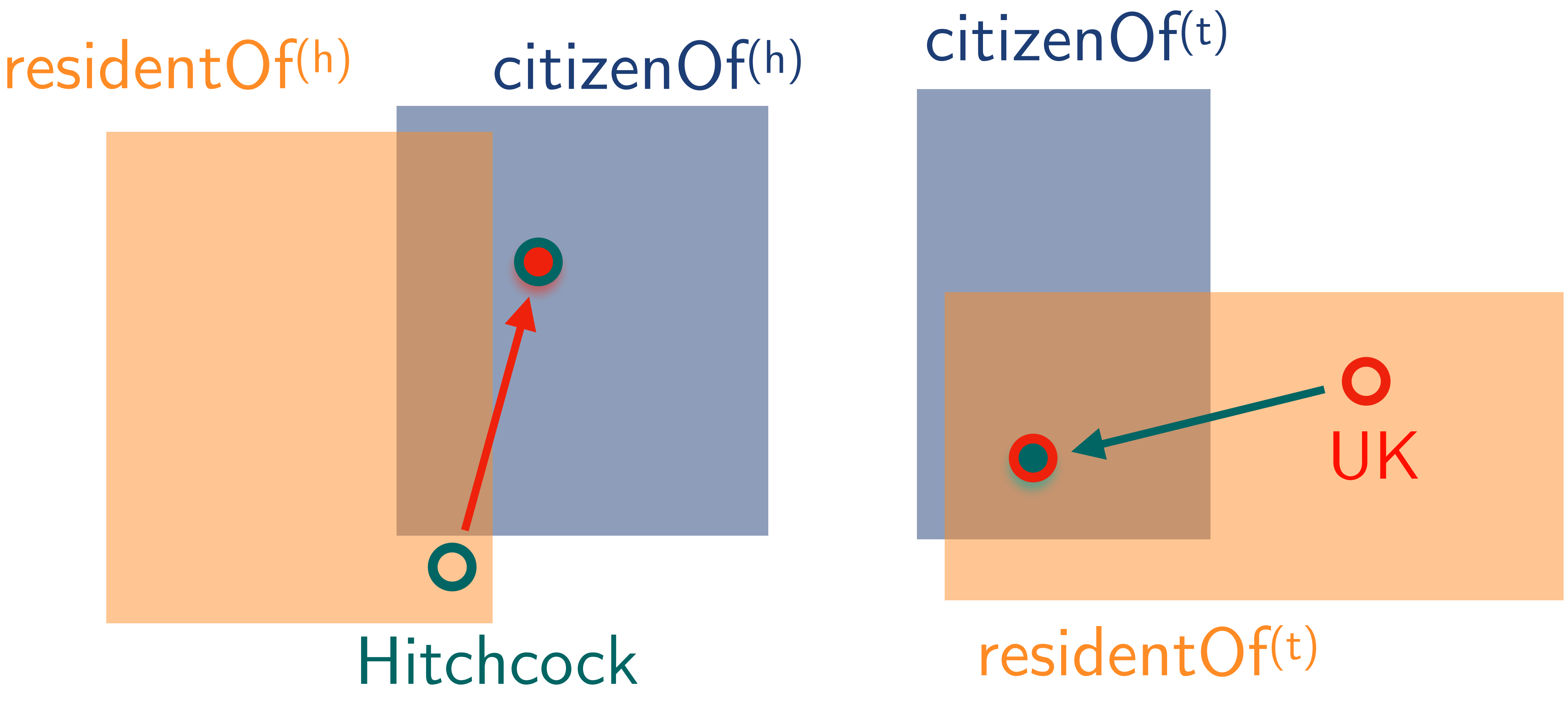
The final entity representation is dynamic: Every entity can have a potentially different final embedding relative to a different fact, since the bump vector depends on the other entity occurring in the fact. Expressive!

How Expressive is BoxE?

How Expressive is BoxE?

Intuitively, head and tail boxes define regions, such that a fact citizenOf(Hitchcock, UK) holds when the final embedding of the entity Hitchcock appears in the box $\text{citizenOf}^{(h)}$ and the the final embedding of the entity UK appears in the box $\text{citizenOf}^{(t)}$.

How Expressive is BoxE?

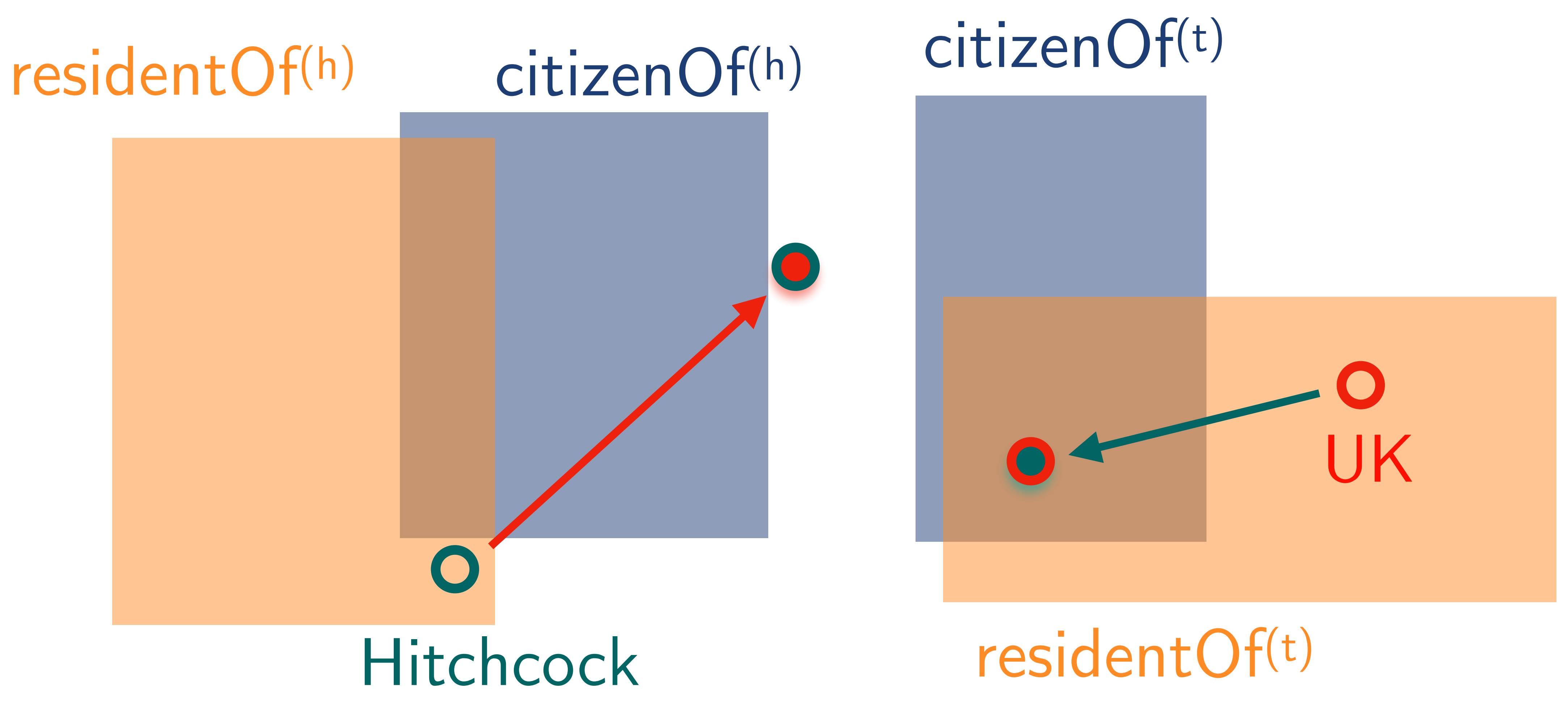


citizenOf(Hitchcock, UK) ✓

residentOf(Hitchcock, UK) ✗

Intuitively, head and tail boxes define regions, such that a fact citizenOf(Hitchcock, UK) holds when the final embedding of the entity Hitchcock appears in the box $\text{citizenOf}^{(h)}$ and the the final embedding of the entity UK appears in the box $\text{citizenOf}^{(t)}$.

How Expressive is BoxE?

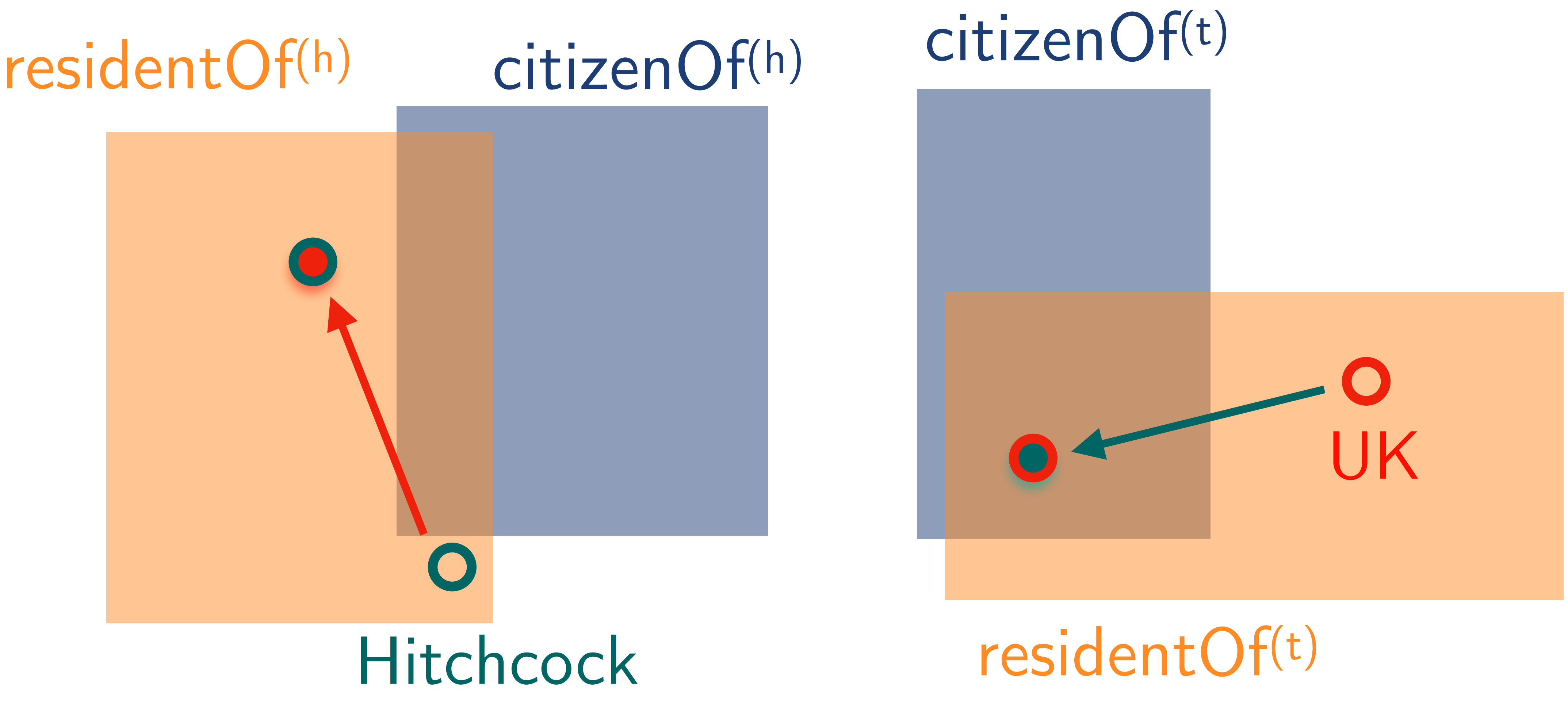


citizenOf(Hitchcock, UK) ✗

residentOf(Hitchcock, UK) ✗

Intuitively, head and tail boxes define regions, such that a fact citizenOf(Hitchcock, UK) holds when the final embedding of the entity Hitchcock appears in the box citizenOf$^{(h)}$ and the the final embedding of the entity UK appears in the box citizenOf$^{(t)}$.

How Expressive is BoxE?

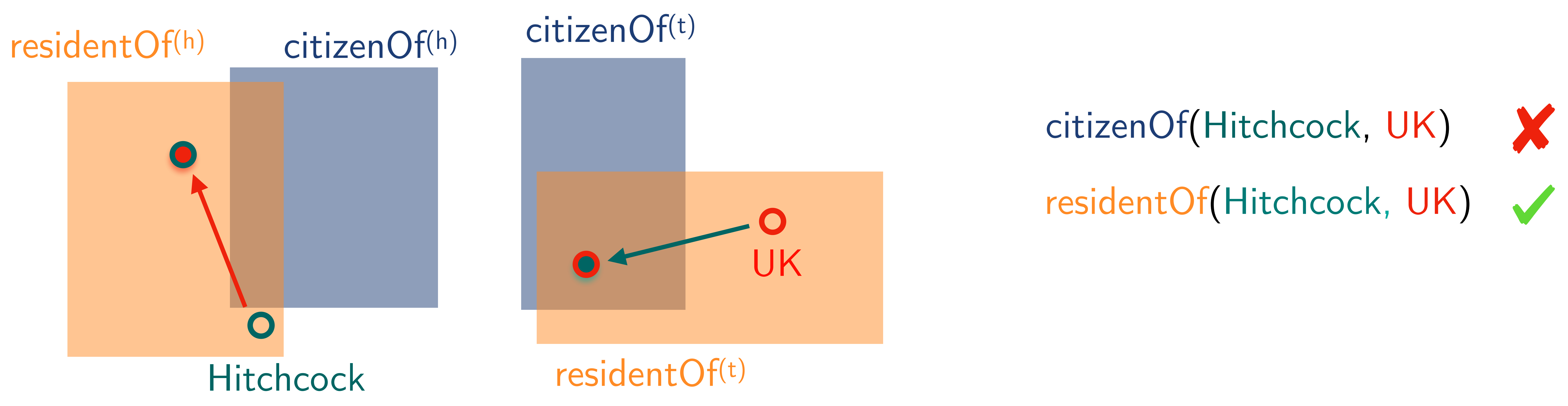


citizenOf(Hitchcock, UK) ✗

residentOf(Hitchcock, UK) ✓

Intuitively, head and tail boxes define regions, such that a fact citizenOf(Hitchcock, UK) holds when the final embedding of the entity Hitchcock appears in the box citizenOf$^{(h)}$ and the the final embedding of the entity UK appears in the box citizenOf$^{(t)}$.

How Expressive is BoxE?

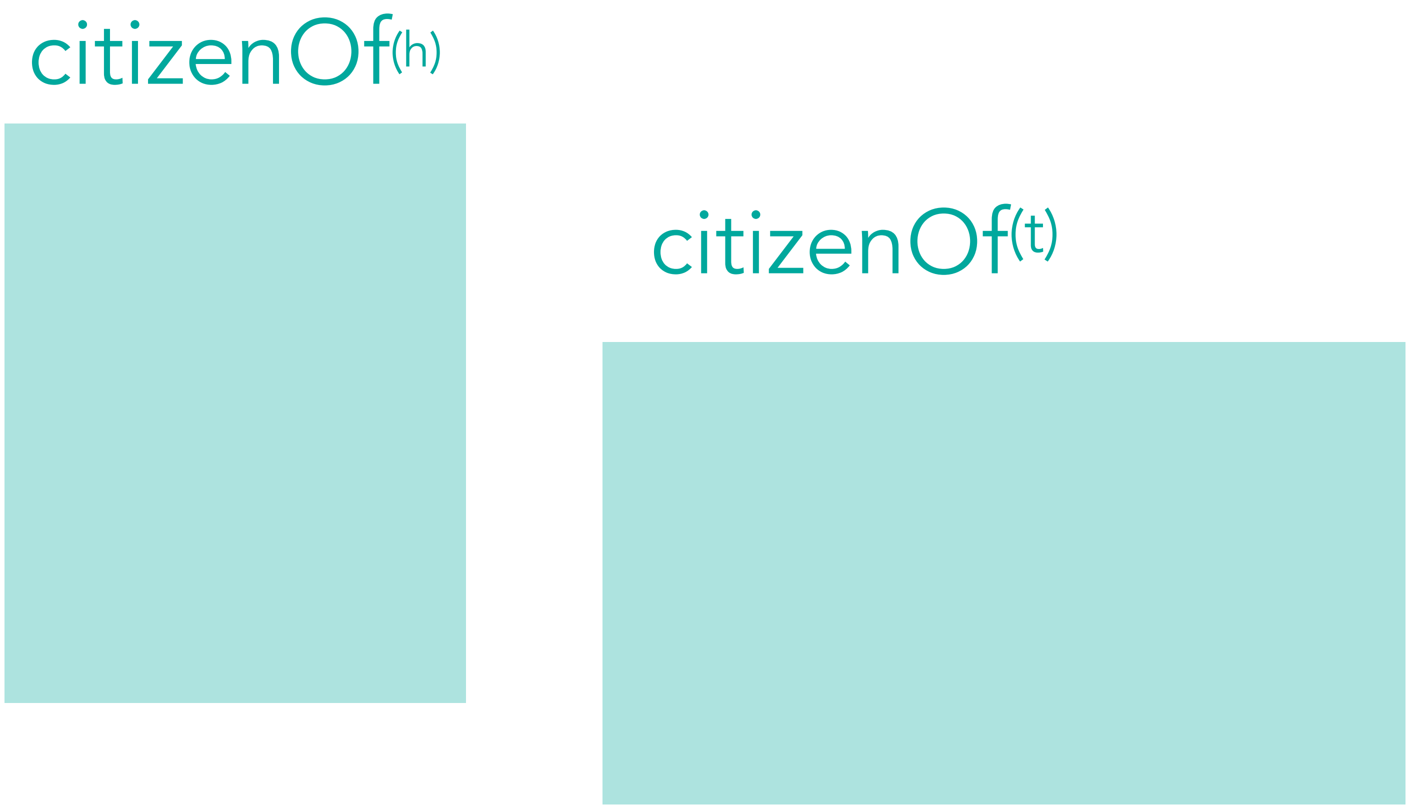


Intuitively, head and tail boxes define regions, such that a fact citizenOf(Hitchcock, UK) holds when the final embedding of the entity Hitchcock appears in the box $\text{citizenOf}^{(h)}$ and the the final embedding of the entity UK appears in the box $\text{citizenOf}^{(t)}$.

Expressiveness: BoxE is indeed fully expressive. Any fact $r(h, t)$ can be made false in the model, by defining a bump vector for, e.g., the head entity h such that the tail entity t is pushed outside of the tail box of r in a single dimension. This operation can be done for all false facts without "harming" the set of true facts, using $E \times R$ dimensions.

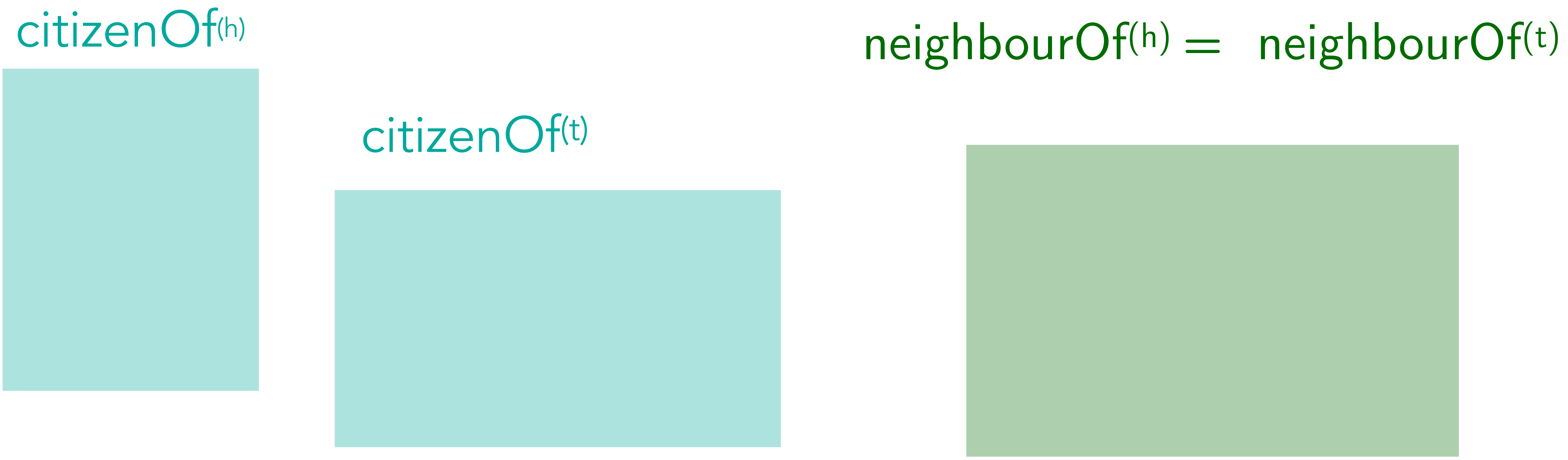
What Inference Patterns can BoxE capture?

What Inference Patterns can BoxE capture?



Anti-symmetry

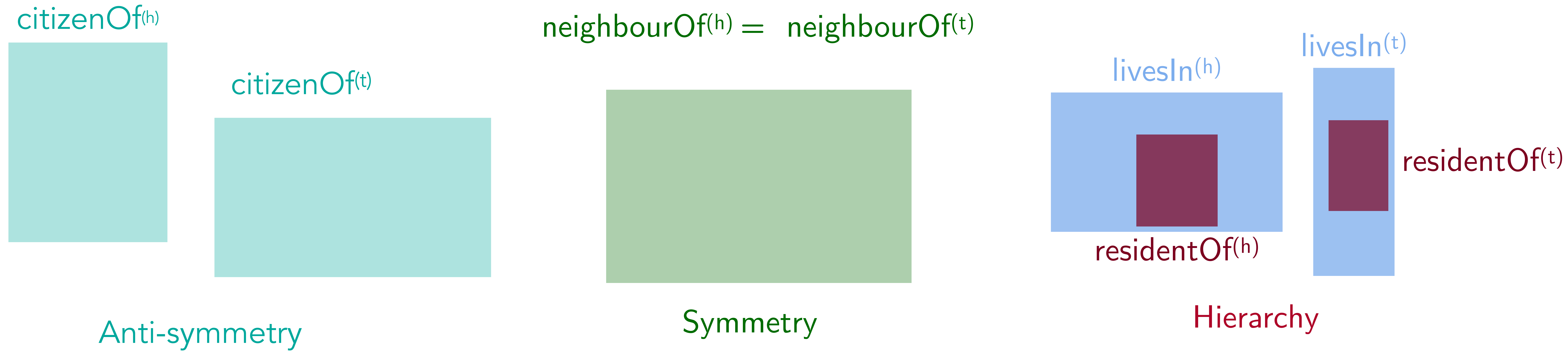
What Inference Patterns can BoxE capture?



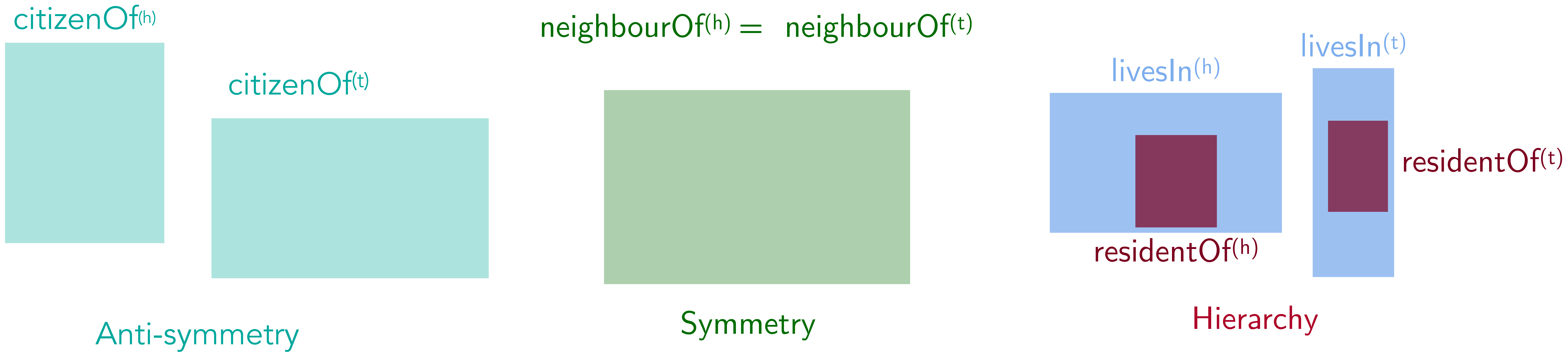
Anti-symmetry

Symmetry

What Inference Patterns can BoxE capture?

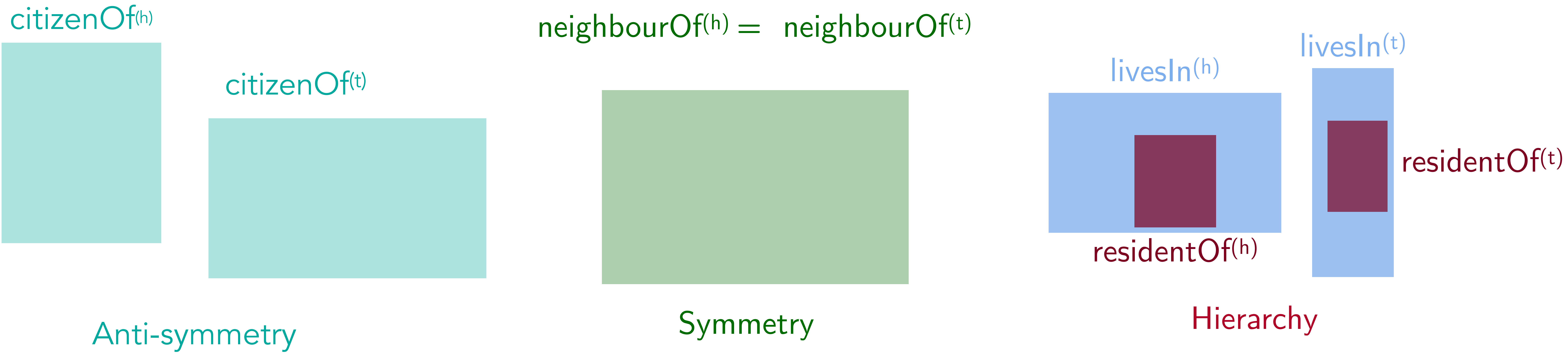


What Inference Patterns can BoxE capture?



Many other inference patterns, e.g., inverse, mutual exclusion, intersection can be captured by configuring boxes in various ways.

What Inference Patterns can BoxE capture?



Many other inference patterns, e.g., inverse, mutual exclusion, intersection can be captured by configuring boxes in various ways.

This approach does not work for the composition pattern: $\forall x, y, z \; r(x, y) \wedge s(y, z) \Rightarrow t(x, z)$! In fact, BoxE cannot capture composition as an inference pattern.

Generalised Inference Patterns

Generalised Inference Patterns

The definition of inference patterns only addresses single application of an inference pattern, and this is generalised to arbitrary applications of the same rule over possibly different relations (Abboud et al.).

Generalised Inference Patterns

The definition of inference patterns only addresses single application of an inference pattern, and this is generalised to arbitrary applications of the same rule over possibly different relations (Abboud et al.).

Question: Can a model capture multiple instances of the same inference pattern jointly?

Generalised Inference Patterns

The definition of inference patterns only addresses single application of an inference pattern, and this is generalised to arbitrary applications of the same rule over possibly different relations (Abboud et al.).

Question: Can a model capture multiple instances of the same inference pattern jointly?

Capturing generalised inference patterns turns out to be significantly more challenging...

Generalised Inference Patterns

The definition of inference patterns only addresses single application of an inference pattern, and this is generalised to arbitrary applications of the same rule over possibly different relations (Abboud et al.).

Question: Can a model capture multiple instances of the same inference pattern jointly?

Capturing generalised inference patterns turns out to be significantly more challenging...

For example, TransE or RotatE can separately capture the composition rules:

$$\forall x, y, z \; r_1(x, y) \wedge r_4(y, z) \Rightarrow r_3(x, z) \text{ and } \forall x, y, z \; r_1(x, y) \wedge r_2(y, z) \Rightarrow r_3(x, z),$$

but jointly capturing these incorrectly forces relation equivalence between r_2 and r_4.

Generalised Inference Patterns

The definition of inference patterns only addresses single application of an inference pattern, and this is generalised to arbitrary applications of the same rule over possibly different relations (Abboud et al.).

Question: Can a model capture multiple instances of the same inference pattern jointly?

Capturing generalised inference patterns turns out to be significantly more challenging...

For example, TransE or RotatE can separately capture the composition rules:

$$\forall x, y, z \; r_1(x, y) \wedge r_4(y, z) \Rightarrow r_3(x, z) \text{ and } \forall x, y, z \; r_1(x, y) \wedge r_2(y, z) \Rightarrow r_3(x, z),$$

but jointly capturing these incorrectly forces relation equivalence between r_2 and r_4.

For another example, consider bilinear models which can separately capture the hierarchy rules:

$$\forall x, y \; r_1(x, y) \Rightarrow r_3(x, y) \text{ and } \forall x, y \; r_2(x, y) \Rightarrow r_3(x, y).$$

Jointly capturing these imposes either $\forall x, y \; r_1(x, y) \Rightarrow r_2(x, y)$ or $\forall x, y \; r_2(x, y) \Rightarrow r_1(x, y)$ (Gutiérrez-Basulto et al.).

Generalised Inference Patterns

The definition of inference patterns only addresses single application of an inference pattern, and this is generalised to arbitrary applications of the same rule over possibly different relations (Abboud et al.).

Question: Can a model capture multiple instances of the same inference pattern jointly?

Capturing generalised inference patterns turns out to be significantly more challenging...

For example, TransE or RotatE can separately capture the composition rules:

$$\forall x, y, z \ r_1(x,y) \wedge r_4(y,z) \Rightarrow r_3(x,z) \text{ and } \forall x, y, z \ r_1(x,y) \wedge r_2(y,z) \Rightarrow r_3(x,z),$$

but jointly capturing these incorrectly forces relation equivalence between r_2 and r_4.

For another example, consider bilinear models which can separately capture the hierarchy rules:

$$\forall x, y \ r_1(x,y) \Rightarrow r_3(x,y) \text{ and } \forall x, y \ r_2(x,y) \Rightarrow r_3(x,y).$$

Jointly capturing these imposes either $\forall x, y \ r_1(x,y) \Rightarrow r_2(x,y)$ or $\forall x, y \ r_2(x,y) \Rightarrow r_1(x,y)$ (Gutiérrez-Basulto et al.).

This means that even a simple relational hierarchy cannot be captured by any of these systems. BoxE can capture these inference patterns also in this general sense, and can capture, e.g., relational hierarchies.

Overview of Embedding Models

Embedding Models: Representation and Scoring

Model	Entity representation	Relation representation	Scoring function
TransE	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$d(\mathbf{h} + \mathbf{r}, \mathbf{t}) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|$
RotatE	$\mathbf{h}, \mathbf{t} \in \mathbb{C}^d$	$\mathbf{r} \in \mathbb{C}^d$	$d(\mathbf{h} \odot \mathbf{r}, \mathbf{t}) = \|\mathbf{h} \odot \mathbf{r} - \mathbf{t}\|$
RESCAL	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{M}_r \in \mathbb{R}^d \times \mathbb{R}^d$	$\mathbf{h}^\top \mathbf{M}_r \mathbf{t}$
DistMult	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{D}_r \in \mathbb{R}^d \times \mathbb{R}^d$	$\mathbf{h}^\top \mathbf{D}_r \mathbf{t}$
ComplEX	$\mathbf{h}, \mathbf{t} \in \mathbb{C}^d$	$\mathbf{D}_r \in \mathbb{C}^d \times \mathbb{C}^d$	$\mathrm{Re}(\mathbf{h}^\top \mathbf{D}_r \bar{\mathbf{t}})$
BoxE	$\mathbf{h}, \mathbf{t}, \mathbf{b_h}, \mathbf{b_t} \in \mathbb{R}^d$	Hyper-rect's $\mathbf{r}^{\mathbf{h}}, \mathbf{r}^{\mathbf{t}} \in \mathbb{R}^d$	$\left\| \mathrm{dist}(\mathbf{h}^{\mathbf{r}(\mathbf{h},\mathbf{t})}, \mathbf{r}^{(\mathbf{h})}) \right\|_x + \left\| \mathrm{dist}(\mathbf{t}^{\mathbf{r}(\mathbf{h},\mathbf{t})}, \mathbf{r}^{(\mathbf{t})}) \right\|_x$

Embedding Models: Representation and Scoring

Model	Entity representation	Relation representation	Scoring function
TransE	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$d(\mathbf{h}+\mathbf{r}, \mathbf{t}) = \Vert\mathbf{h}+\mathbf{r}-\mathbf{t}\Vert$
RotatE	$\mathbf{h}, \mathbf{t} \in \mathbb{C}^d$	$\mathbf{r} \in \mathbb{C}^d$	$d(\mathbf{h}\odot\mathbf{r}, \mathbf{t}) = \Vert\mathbf{h}\odot\mathbf{r}-\mathbf{t}\Vert$
RESCAL	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{M}_r \in \mathbb{R}^d \times \mathbb{R}^d$	$\mathbf{h}^\top \mathbf{M}_r \mathbf{t}$
DistMult	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{D}_r \in \mathbb{R}^d \times \mathbb{R}^d$	$\mathbf{h}^\top \mathbf{D}_r \mathbf{t}$
ComplEX	$\mathbf{h}, \mathbf{t} \in \mathbb{C}^d$	$\mathbf{D}_r \in \mathbb{C}^d \times \mathbb{C}^d$	$\text{Re}(\mathbf{h}^\top \mathbf{D}_r \bar{\mathbf{t}})$
BoxE	$\mathbf{h}, \mathbf{t}, \mathbf{b}_\mathbf{h}, \mathbf{b}_\mathbf{t} \in \mathbb{R}^d$	Hyper-rect's $\mathbf{r}^\mathbf{h}, \mathbf{r}^\mathbf{t} \in \mathbb{R}^d$	$\Big\Vert \text{dist}(\mathbf{h}^{\mathbf{r}(\mathbf{h},\mathbf{t})}, \mathbf{r}^{(\mathbf{h})}) \Big\Vert_x + \Big\Vert \text{dist}(\mathbf{t}^{\mathbf{r}(\mathbf{h},\mathbf{t})}, \mathbf{r}^{(\mathbf{t})}) \Big\Vert_x$

A summary of the models covered in the lecture: Entity representations $h, t \in E$ and relation representations $r \in R$ are given, and the scoring function is given for an arbitrary fact $r(h, t)$.

Embedding Models: Representation and Scoring

Model	Entity representation	Relation representation	Scoring function
TransE	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{r} \in \mathbb{R}^d$	$d(\mathbf{h}+\mathbf{r}, \mathbf{t}) = \Vert\mathbf{h}+\mathbf{r}-\mathbf{t}\Vert$
RotatE	$\mathbf{h}, \mathbf{t} \in \mathbb{C}^d$	$\mathbf{r} \in \mathbb{C}^d$	$d(\mathbf{h}\odot\mathbf{r}, \mathbf{t}) = \Vert\mathbf{h}\odot\mathbf{r}-\mathbf{t}\Vert$
RESCAL	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{M}_r \in \mathbb{R}^d \times \mathbb{R}^d$	$\mathbf{h}^\top \mathbf{M}_r \mathbf{t}$
DistMult	$\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$	$\mathbf{D}_r \in \mathbb{R}^d \times \mathbb{R}^d$	$\mathbf{h}^\top \mathbf{D}_r \mathbf{t}$
ComplEX	$\mathbf{h}, \mathbf{t} \in \mathbb{C}^d$	$\mathbf{D}_r \in \mathbb{C}^d \times \mathbb{C}^d$	$\mathrm{Re}(\mathbf{h}^\top \mathbf{D}_r \bar{\mathbf{t}})$
BoxE	$\mathbf{h}, \mathbf{t}, \mathbf{b}_\mathbf{h}, \mathbf{b}_\mathbf{t} \in \mathbb{R}^d$	Hyper-rect's $\mathbf{r}^\mathbf{h}, \mathbf{r}^\mathbf{t} \in \mathbb{R}^d$	$\Vert \mathrm{dist}(\mathbf{h}^{\mathbf{r}(\mathbf{h},\mathbf{t})}, \mathbf{r}^{(\mathbf{h})}) \Vert_x + \Vert \mathrm{dist}(\mathbf{t}^{\mathbf{r}(\mathbf{h},\mathbf{t})}, \mathbf{r}^{(\mathbf{t})}) \Vert_x$

A summary of the models covered in the lecture: Entity representations $h, t \in E$ and relation representations $r \in R$ are given, and the scoring function is given for an arbitrary fact $r(h, t)$.

Model specific representation constraints are excluded from the Table, and so are regularisation constraints. Please refer to the respective original work for the details.

Embedding Models: Expressiveness and Inferences

Inference pattern	TransE	RotatE	BoxE	DistMult	ComplEX
Symmetry	N/N	Y/Y	Y/Y	Y/Y	Y/Y
Anti-symmetry	Y/Y	Y/Y	Y/Y	N/N	Y/Y
Inversion	Y/N	Y/Y	Y/Y	N/N	Y/Y
Composition	Y/N	Y/N	N/N	N/N	N/N
Hierarchy	N/N	N/N	Y/Y	Y/N	Y/N
Intersection	Y/N	Y/N	Y/Y	N/N	N/N
Mutual exclusion	Y/Y	Y/Y	Y/Y	Y/N	Y/N

Embedding Models: Expressiveness and Inferences

Inference pattern	TransE	RotatE	BoxE	DistMult	ComplEX
Symmetry	N/N	Y/Y	Y/Y	Y/Y	Y/Y
Anti-symmetry	Y/Y	Y/Y	Y/Y	N/N	Y/Y
Inversion	Y/N	Y/Y	Y/Y	N/N	Y/Y
Composition	Y/N	Y/N	N/N	N/N	N/N
Hierarchy	N/N	N/N	Y/Y	Y/N	Y/N
Intersection	Y/N	Y/N	Y/Y	N/N	N/N
Mutual exclusion	Y/Y	Y/Y	Y/Y	Y/N	Y/N

A summary of the inference patterns / generalised inference patterns that can be captured by selected models.

Embedding Models: Expressiveness and Inferences

Inference pattern	TransE	RotatE	BoxE	DistMult	ComplEX
Symmetry	N/N	Y/Y	Y/Y	Y/Y	Y/Y
Anti-symmetry	Y/Y	Y/Y	Y/Y	N/N	Y/Y
Inversion	Y/N	Y/Y	Y/Y	N/N	Y/Y
Composition	Y/N	Y/N	N/N	N/N	N/N
Hierarchy	N/N	N/N	Y/Y	Y/N	Y/N
Intersection	Y/N	Y/N	Y/Y	N/N	N/N
Mutual exclusion	Y/Y	Y/Y	Y/Y	Y/N	Y/N

A summary of the inference patterns / generalised inference patterns that can be captured by selected models.

Another bilinear model TuckER, coincides with ComplEX in terms of the listed inference patterns.

Outlook and Discussions

Neural Models

Neural Models

We have not discussed neural models and focused on so-called shallow embedding models so far.

Neural Models

We have not discussed neural models and focused on so-called shallow embedding models so far.

General approach: Neural models either use a neural network as a scoring function (e.g., ConvE), or use existing embedding models for scoring, but learn the embeddings with a neural network (e.g., r-GCN).

Neural Models

We have not discussed neural models and focused on so-called shallow embedding models so far.

General approach: Neural models either use a neural network as a scoring function (e.g., ConvE), or use existing embedding models for scoring, but learn the embeddings with a neural network (e.g., r-GCN).

Expressiveness vs Interpretability: Neural models are typically expressive, but they are hard to interpret and evaluate, since they are mostly black-box.

Neural Models

We have not discussed neural models and focused on so-called shallow embedding models so far.

General approach: Neural models either use a neural network as a scoring function (e.g., ConvE), or use existing embedding models for scoring, but learn the embeddings with a neural network (e.g., r-GCN).

Expressiveness vs Interpretability: Neural models are typically expressive, but they are hard to interpret and evaluate, since they are mostly black-box.

Practical: Shallow embedding models are state-of-the-art on many benchmark datasets.

Neural Models

We have not discussed neural models and focused on so-called shallow embedding models so far.

General approach: Neural models either use a neural network as a scoring function (e.g., ConvE), or use existing embedding models for scoring, but learn the embeddings with a neural network (e.g., r-GCN).

Expressiveness vs Interpretability: Neural models are typically expressive, but they are hard to interpret and evaluate, since they are mostly black-box.

Practical: Shallow embedding models are state-of-the-art on many benchmark datasets.

Conceptual: Shallow approaches are inherently transductive (i.e., limited to the entities they are trained on; see, e.g., (Hamilton et al., 2017)), while some neural models learn inductive representations (i.e., once learned, they can be applied to unseen entities).

Neural Models

We have not discussed neural models and focused on so-called shallow embedding models so far.

General approach: Neural models either use a neural network as a scoring function (e.g., ConvE), or use existing embedding models for scoring, but learn the embeddings with a neural network (e.g., r-GCN).

Expressiveness vs Interpretability: Neural models are typically expressive, but they are hard to interpret and evaluate, since they are mostly black-box.

Practical: Shallow embedding models are state-of-the-art on many benchmark datasets.

Conceptual: Shallow approaches are inherently transductive (i.e., limited to the entities they are trained on; see, e.g., (Hamilton et al., 2017)), while some neural models learn inductive representations (i.e., once learned, they can be applied to unseen entities).

We will briefly revisit knowledge graph completion in the context of graph neural networks, later in the course.

Beyond This Lecture

Beyond This Lecture

- **Models**: We focused on particular representative models, but there are many more...

Beyond This Lecture

- **Models**: We focused on particular representative models, but there are many more...
  - Beyond Euclidian spaces, e.g., Poincare embeddings.

Beyond This Lecture

- **Models**: We focused on particular representative models, but there are many more...
  - Beyond Euclidian spaces, e.g., Poincare embeddings.
  - Other geometrical abstractions, e.g., TorusE.

Beyond This Lecture

- **Models**: We focused on particular representative models, but there are many more...
  - Beyond Euclidian spaces, e.g., Poincare embeddings.
  - Other geometrical abstractions, e.g., TorusE.
- **Practical considerations**: Many regularisation/optimisation techniques are omitted, see especially for recent evaluations (Rufinelli, 2020).

Beyond This Lecture

- **Models**: We focused on particular representative models, but there are many more...
  - Beyond Euclidian spaces, e.g., Poincare embeddings.
  - Other geometrical abstractions, e.g., TorusE.
- **Practical considerations**: Many regularisation/optimisation techniques are omitted, see especially for recent evaluations (Rufinelli, 2020).
- **Higher-arity knowledge bases**: Real-world data is not necessarily in the form of binary atoms, forming a graph. Facts can be of higher arity, e.g., hasDegreeFrom(Hawking,Cambridge,DPhil), and very few models can handle data with arbitrary arity relations.

Beyond This Lecture

- **Models**: We focused on particular representative models, but there are many more...
  - Beyond Euclidian spaces, e.g., Poincare embeddings.
  - Other geometrical abstractions, e.g., TorusE.
- **Practical considerations**: Many regularisation/optimisation techniques are omitted, see especially for recent evaluations (Rufinelli, 2020).
- **Higher-arity knowledge bases**: Real-world data is not necessarily in the form of binary atoms, forming a graph. Facts can be of higher arity, e.g., hasDegreeFrom(Hawking,Cambridge,DPhil), and very few models can handle data with arbitrary arity relations.
- **Rule injection**: KGs usually have an accompanying schema, or an ontology, encoding the general domain knowledge in the form of first-order rules. Ideally, all predictions in the KG completion task should comply with such knowledge. Is it possible to inject such knowledge into the embedding models and to what extent?

Beyond This Lecture

- **Models**: We focused on particular representative models, but there are many more...
  - Beyond Euclidian spaces, e.g., Poincare embeddings.
  - Other geometrical abstractions, e.g., TorusE.
- **Practical considerations**: Many regularisation/optimisation techniques are omitted, see especially for recent evaluations (Rufinelli, 2020).
- **Higher-arity knowledge bases**: Real-world data is not necessarily in the form of binary atoms, forming a graph. Facts can be of higher arity, e.g., hasDegreeFrom(Hawking,Cambridge,DPhil), and very few models can handle data with arbitrary arity relations.
- **Rule injection**: KGs usually have an accompanying schema, or an ontology, encoding the general domain knowledge in the form of first-order rules. Ideally, all predictions in the KG completion task should comply with such knowledge. Is it possible to inject such knowledge into the embedding models and to what extent?
- **Other tasks**: Taks beyond KG completion, e.g., entity classification, query answering with embedding models.

Summary

Summary

- We have seen many concrete shallow knowledge graph embedding models:

Summary

- We have seen many concrete shallow knowledge graph embedding models:
  - Translational models, e.g., TransE, RotatE.

Summary

- We have seen many concrete shallow knowledge graph embedding models:
  - Translational models, e.g., TransE, RotatE.
  - Bilinear models, e.g., RESCAL, DistMULT, ComplEx.

Summary

- We have seen many concrete shallow knowledge graph embedding models:
  - Translational models, e.g., TransE, RotatE.
  - Bilinear models, e.g., RESCAL, DistMULT, ComplEx.
  - Box embeddings, e.g., BoxE.

Summary

- We have seen many concrete shallow knowledge graph embedding models:
  - Translational models, e.g., TransE, RotatE.
  - Bilinear models, e.g., RESCAL, DistMULT, ComplEx.
  - Box embeddings, e.g., BoxE.
- Many embedding models which are not covered build on similar, or analogous ideas.

Summary

- We have seen many concrete shallow knowledge graph embedding models:
  - Translational models, e.g., TransE, RotatE.
  - Bilinear models, e.g., RESCAL, DistMULT, ComplEx.
  - Box embeddings, e.g., BoxE.
- Many embedding models which are not covered build on similar, or analogous ideas.
- We evaluated the respective models in terms of:

Summary

- We have seen many concrete shallow knowledge graph embedding models:
  - Translational models, e.g., TransE, RotatE.
  - Bilinear models, e.g., RESCAL, DistMULT, ComplEx.
  - Box embeddings, e.g., BoxE.
- Many embedding models which are not covered build on similar, or analogous ideas.
- We evaluated the respective models in terms of:
  - Model expressiveness

Summary

- We have seen many concrete shallow knowledge graph embedding models:
  - Translational models, e.g., TransE, RotatE.
  - Bilinear models, e.g., RESCAL, DistMULT, ComplEx.
  - Box embeddings, e.g., BoxE.
- Many embedding models which are not covered build on similar, or analogous ideas.
- We evaluated the respective models in terms of:
  - Model expressiveness
  - Model inductive capacity and inference patterns

References

- A. Bordes, J. Weston, R. Collobert, and Y. Bengio, Learning structured embeddings of knowledge bases. *AAAI*, 2011.
- A. Bordes, X. Glorot, J. Weston, and Y. Bengio, Joint learning of words and meaning representations for open-text semantic parsing. *AISTATS*, 2012.
- A. Bordes, N. Usunier, A. García-Durán, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. *NIPS*, 2013.
- M. Nickel, V. Tresp, and H.-P. Kriegel. A three-way model for collective learning on multi-relational data. *ICML*, 2011.
- Z. Wang, J. Zhang, J. Feng, and Z. Chen, Knowledge graph embedding by translating on hyperplanes. *AAAI*, 2014.
- S. He, K. Liu, G. Ji, and J. Zhao. Learning to represent knowledge graphs with Gaussian embedding. *CIKM*, 2015.
- T. Ebisu and R. Ichise. TorusE: Knowledge graph embedding on a lie group. *AAAI*, 2018.
- Z. Sun, Z. Deng, J. Nie, and J. Tang. RotatE: Knowledge graph embedding by relational rotation in complex space. *ICLR*, 2019.

References

- R. Abboud, İ.İ. Ceylan, T.Lukasiewicz, T. Salvatori. BoxE: A Box Embedding Model for Knowledge Base Completion. *NeurIPS*, 2020.
- A. Bordes, X. Glorot, J. Weston, and Y. Bengio. A semantic matching energy function for learning with multi-relational data. *Mach. Learn.*, 2014.
- R. Socher, D. Chen, C. D. Manning, and A. Y. Ng. Reasoning with neural tensor networks for knowledge base completion. *NIPS*, 2013.
- B. Yang, W.-T. Yih, X. He, J. Gao, and L. Deng, Embedding entities and relations for learning and inference in knowledge bases. *ICLR*, 2015.
- Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, 2015.
- T. Trouillon, J. Welbl, S. Riedel, E. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction". *ICML*, 2016.
- T. Dettmers, P. Minervini, P. Stenetorp, S. Riedel. Convolutional 2D knowledge graph embeddings. *AAAI*, 2018.
- D. Ruffinelli, S. Broscheit, R. Gemulla. You CAN Teach an Old Dog New Tricks! On Training Knowledge Graph Embeddings, *ICLR*, 2020.

References

- I. Balazevic, C. Allen, and T. Hospedales. TuckER: Tensor factorization for knowledge graph completion. *EMNLP-IJCNLP*, 2019.
- M. Schlichtkrull, T. Kipf, P. Bloem, R. Van Den Berg, I. Titov, M. Welling, Modelling Relational Data with Graph Convolutional Networks. *ESWC*, 2018.
- M. Nickel and D. Kiela. Poincaré embeddings for learning hierarchical representations. *NIPS*, 2017.
- L. Cai and W. Y. Wang. KBGan: Adversarial learning for knowledge graph embeddings. *NAACL-HLT*, 2018.
- V. Gutiérrez-Basulto and S. Schockaert. From knowledge graph embedding to ontology embedding? an analysis of the compatibility between vector space representations and rules. *KR*, 2018
- L. Vilnis, X. Li, X., S. Murty, and A. McCallum. Probabilistic embedding of knowledge graphs with box lattice measures. *ACL, 2018*.
- H. Ren, W. Hu, J. Leskovec. Query2box: Reasoning over Knowledge Graphs in Vector Space Using Box Embeddings, *ICLR*, 2020.
- W.L. Hamilton, R. Ying, and J. Leskovec. Inductive representation learning on large graphs. *NeurIPS*, 2017.