Composition-based Multi-Relational Graph Convolutional Networks

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, Partha Talukdar

Presented by - Sripath Mishra, Te-Lin Wu

01/19, CS249, Winter 2021

Outline

- Introduction
- Related Works
- Background
- Main Method: CompGCN
- Experiments
- Conclusions & Summaries

Outline

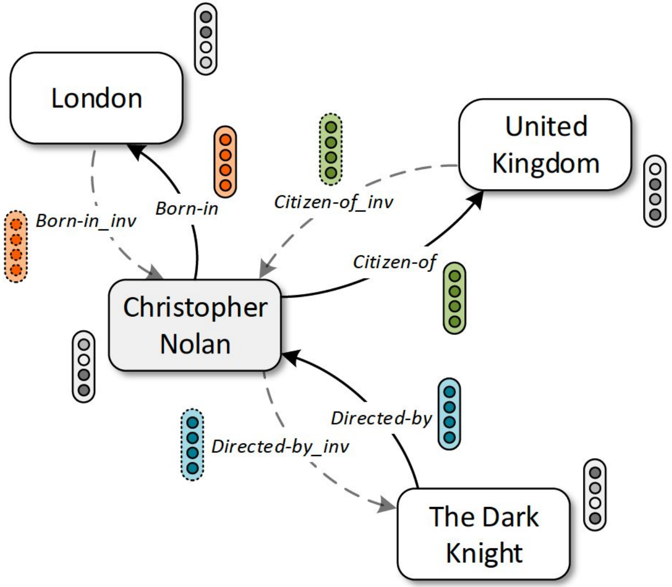
- **Introduction**
- Related Works
- Background
- Main Method: CompGCN
- Experiments
- Conclusions & Summaries

Motivation

- Graphs are very expressive data structures
- Graph Convolutional Neural Network (GCN) is proven powerful
- Most GCN handles undirected graphs

Motivation

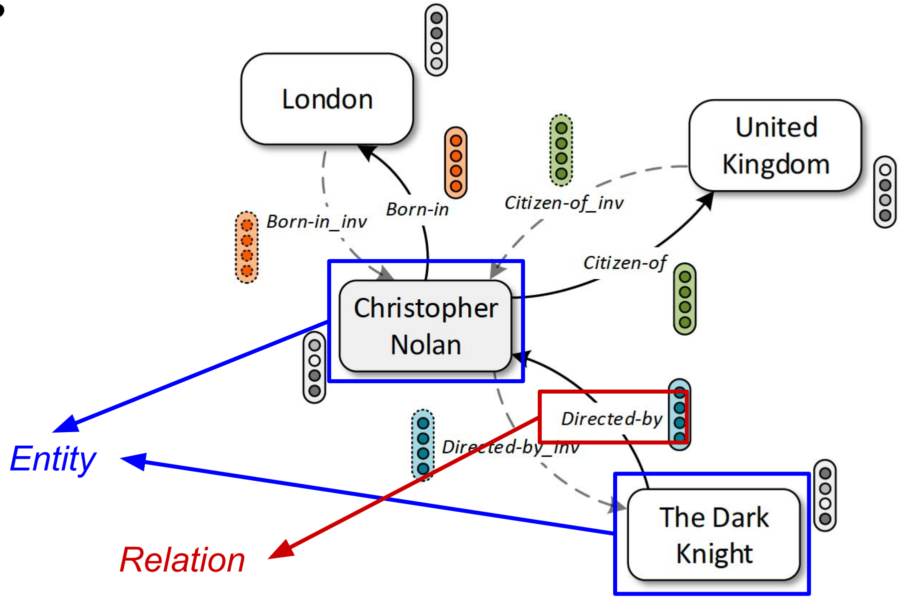
- Graphs are very expressive data structures
- Graph Convolutional Neural Network (GCN) is proven powerful
- Most GCN handles undirected graphs
- Multi-Relational Graphs
  - Graphs with ***relations***
  - Eg. Knowledge Graphs



Relational Graph with Embeddings

Motivation

- Graphs are very expressive data structures
- Graph Convolutional Neural Network (GCN) is proven powerful
- Most GCN handles undirected graphs
- Multi-Relational Graphs
  - Graphs with ***relations***
  - Eg. Knowledge Graphs



Relational Graph with Embeddings

Motivation

- Cons of prior works on multi-relational graphs
  - Over-parameterization
  - Only handles node representations
    - Can not perform link predictions
- Existing techniques on knowledge graph embeddings (KE)
  - Only does link predictions

Motivation

- Cons of prior works on multi-relational graphs
  - Over-parameterization
  - Only handles node representations
    - Can not perform link predictions
- Existing techniques on knowledge graph embeddings (KE)
  - Only does link predictions
- **Solution:**
  - **Combine both: GCN + KE**

CompGCN

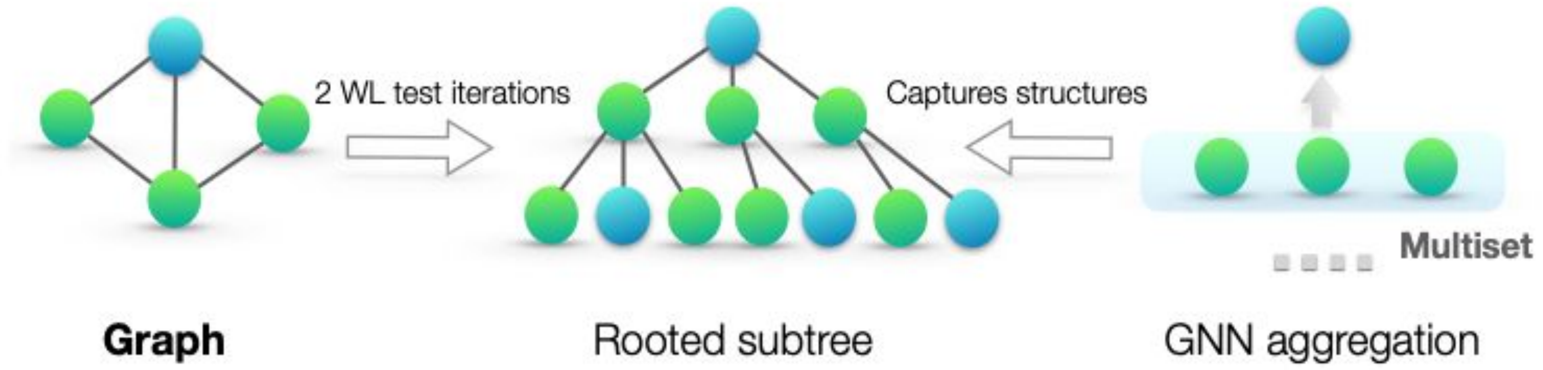
- **Comp**ositional-based **G**raph **C**onvolutional **N**etwork

Outline

- Introduction
- **Related Works**
- Background
- Main Method: CompGCN
- Experiments
- Conclusions & Summaries

Related Works

- Graph Convolutional Networks

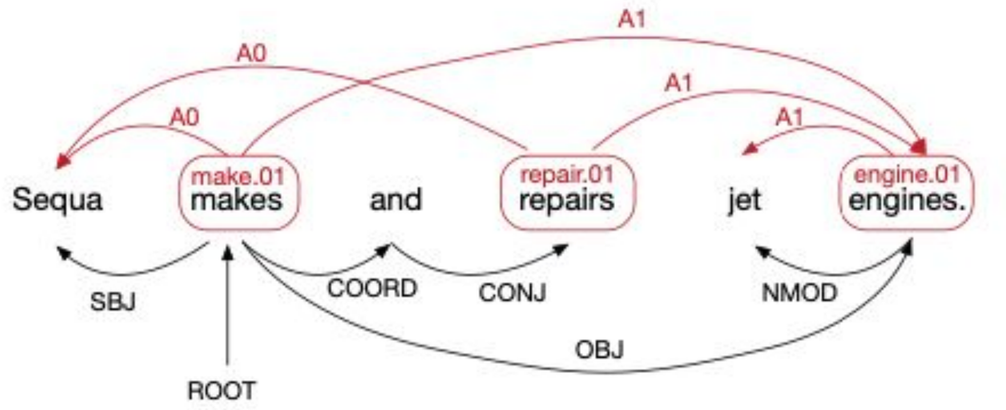


Middle panel: **rooted subtree structures** (at the blue node) that the *WL test* uses to distinguish different graphs. Right panel: if a GNN's aggregation function captures the *full multiset* of node neighbors, the GNN can capture the rooted subtrees in a recursive manner and be as powerful as the WL test.

[Xu et al., 2019]

Related Works

- GCNs for Multi-Relational Graph

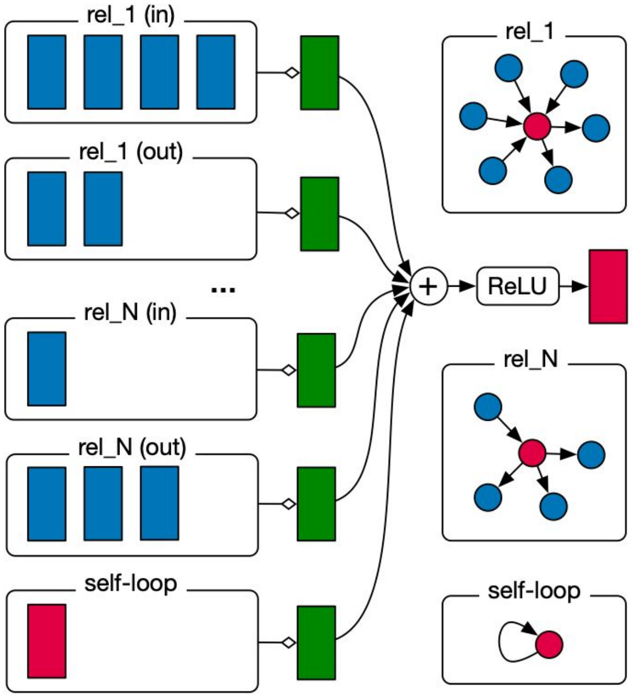


An example sentence annotated with semantic (top) and syntactic dependencies (bottom).

[Marcheggiani & Titov (2017)]

Related Works

- GCNs for Multi-Relational Graphs



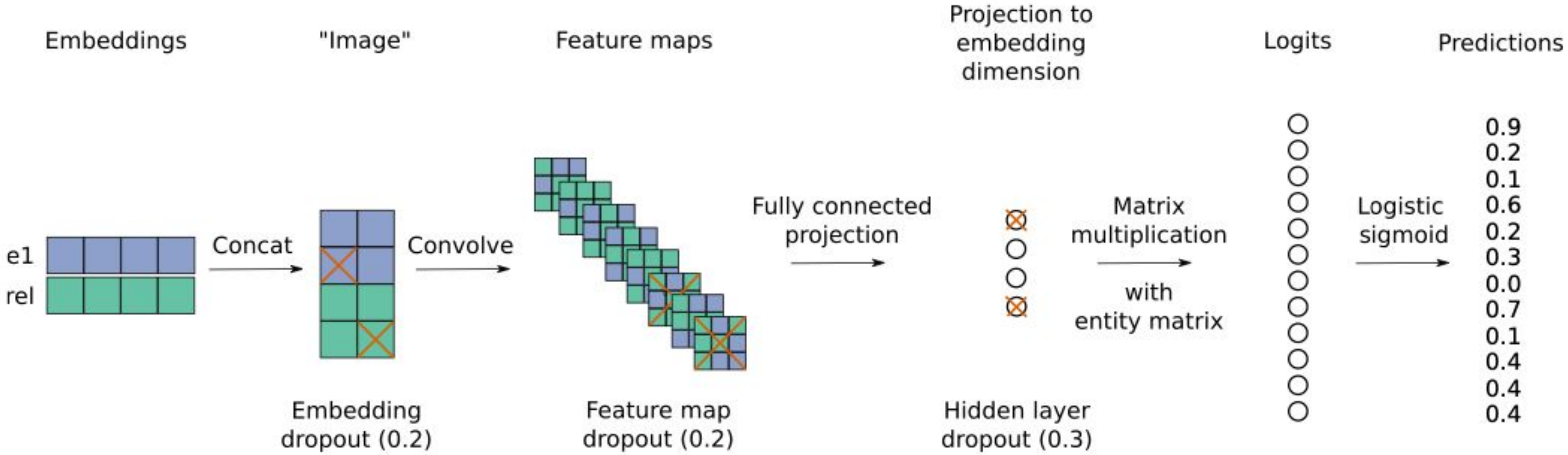
Update in the **R-GCN** model.

Activations (d-dimensional vectors) from neighboring nodes (dark blue) are gathered and then transformed ***for each relation type individually*** (for both in- and outgoing edges). The resulting representation (green) is accumulated in a (normalized) sum and passed through an activation function (such as the ReLU).

[Schlichtkrull et al., 2017]

Related Works

- Knowledge Graph Embedding



In the **ConvE** model, the entity and relation embeddings are first reshaped and concatenated (steps 1, 2); the resulting matrix is then used as input to a convolutional layer (step 3); the resulting feature map tensor is vectorised and projected into a k-dimensional space (step 4) and matched with all candidate object embeddings (step 5).

[Dettmers et al., 2018]

Outline

- Introduction
- Related Works
- **Background**
- Main Method: CompGCN
- Experiments
- Conclusions & Summaries

GCN on Undirected Graphs

- Undirected Graph: $G = (V, E, X)$
  - V: set of vertices
  - E: set of edges
  - $X \in R^{|v|*d0}$: d_0-dimensional input features of each node

- The node representation for a single GCN layer: $H = f(\hat{A}XW)$
  - $\hat{A} = D^{-1/2}(A+I)D^{+1/2}$
  - $D_{ii} = \sum_j (A+I)_{ij}$,
  - $W \in R^{d0xd1}$
  - f: some activation function

GCN on Multi-Relational Graphs

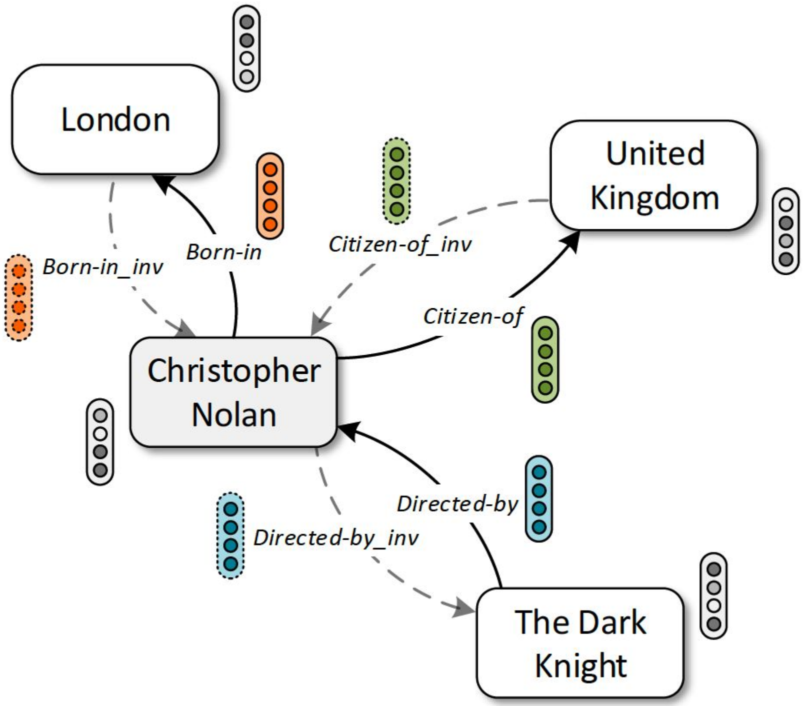
- Multi-Relational Graph: G = (V, R, E, X)
  - R: set of relations
  - (u, v, r): relation $r \in R$ from node u to v.
  - Inverse edge: (v, u, r^{-1})

- k-th layer representations of a relational graph: $H^{k+1}=f(\hat{A}H^kW^k_r)$
  - W^k_r : relation-specific parameters

Outline

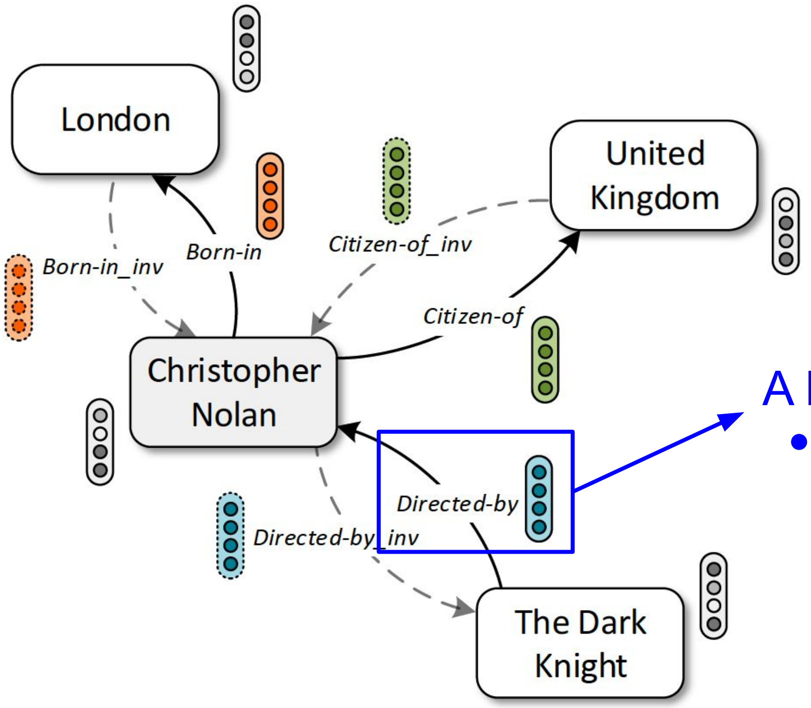
- Introduction
- Related Works
- Background
- **Main Method: CompGCN**
- Experiments
- Conclusions & Summaries

Overview



Relational Graph with Embeddings

Overview

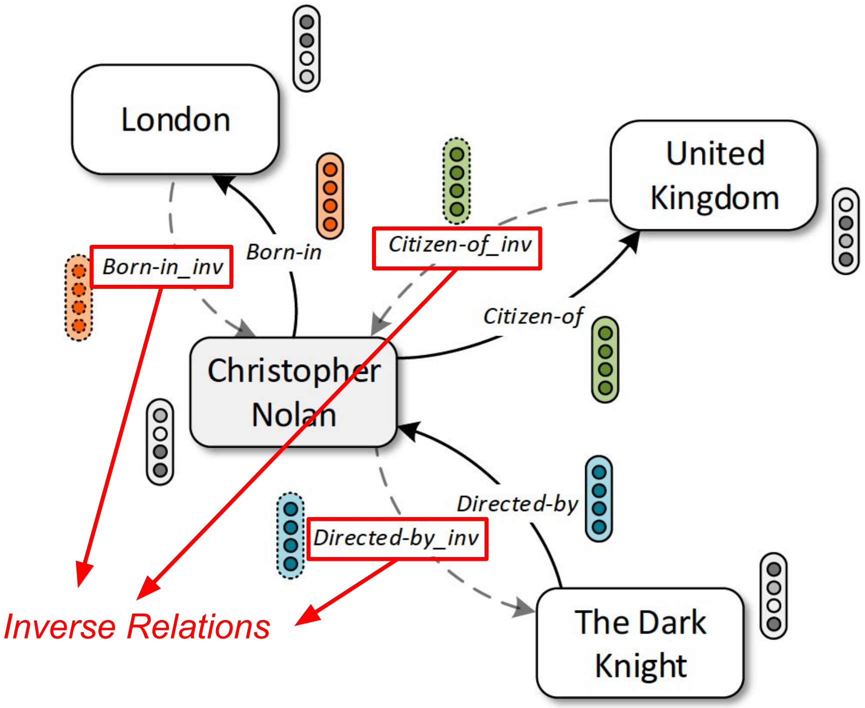


Relational Graph with Embeddings

A KG Triplet (E_1, R, E_2):

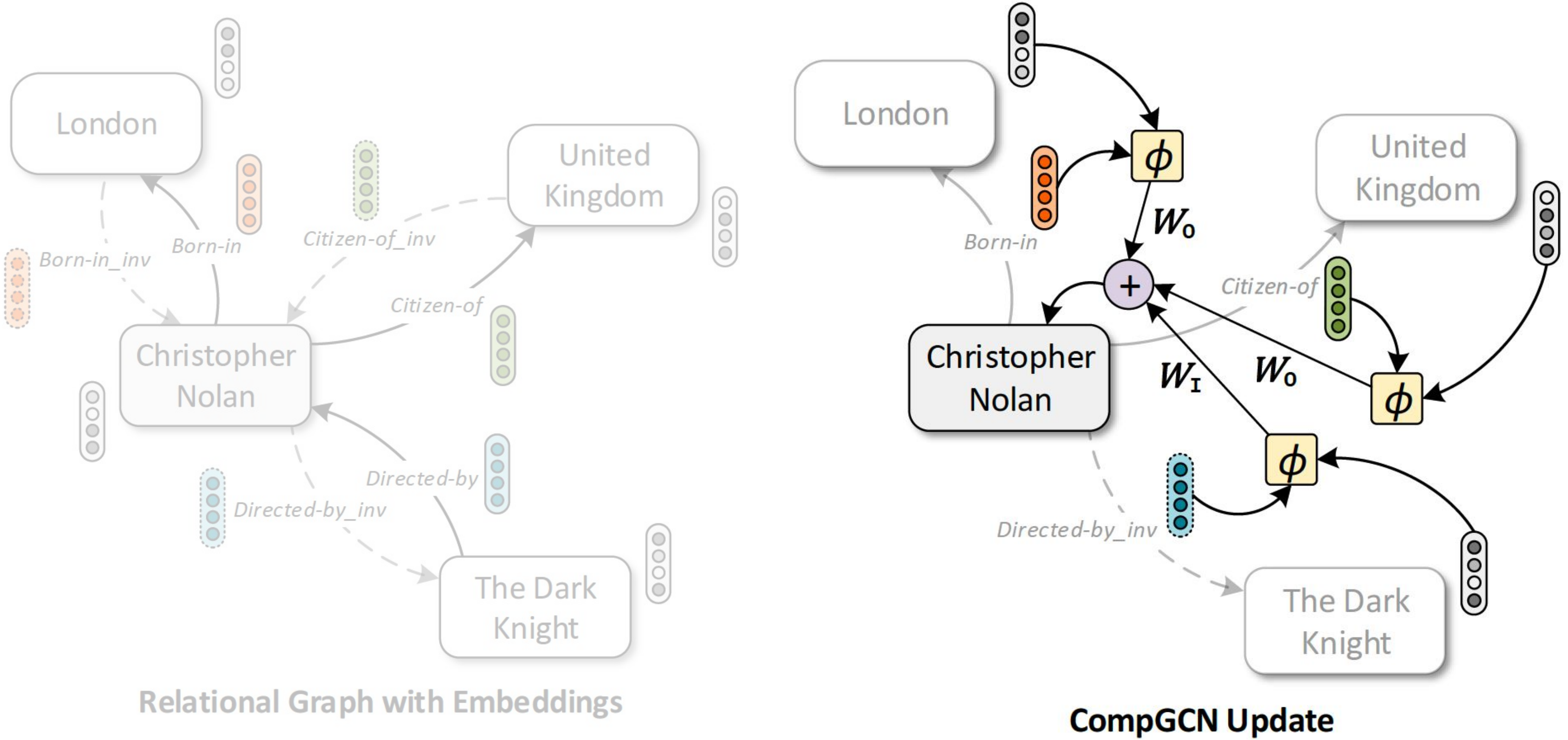
- Eg. (The Dark Knight, Directed by, Christopher Nolan)

Overview

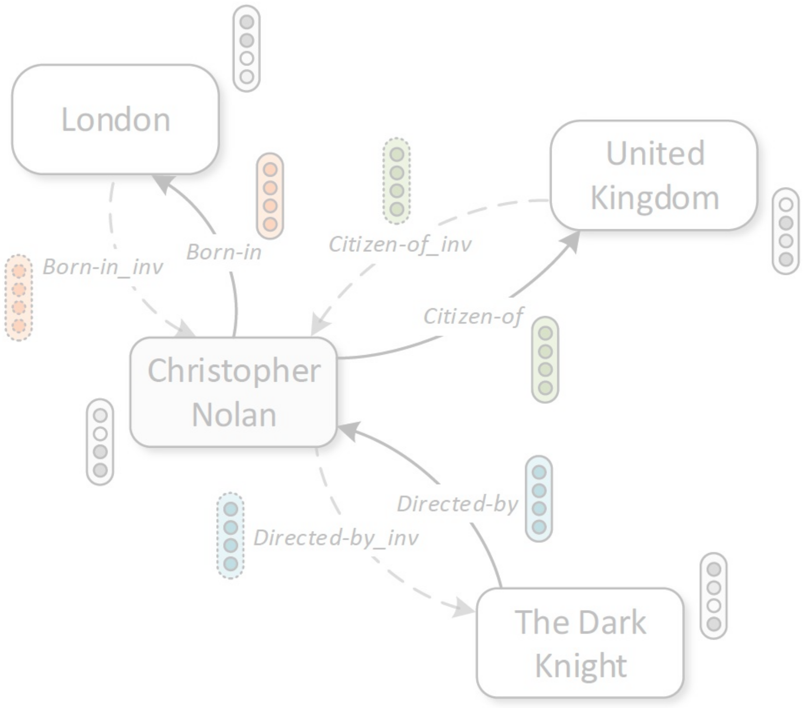


Relational Graph with Embeddings

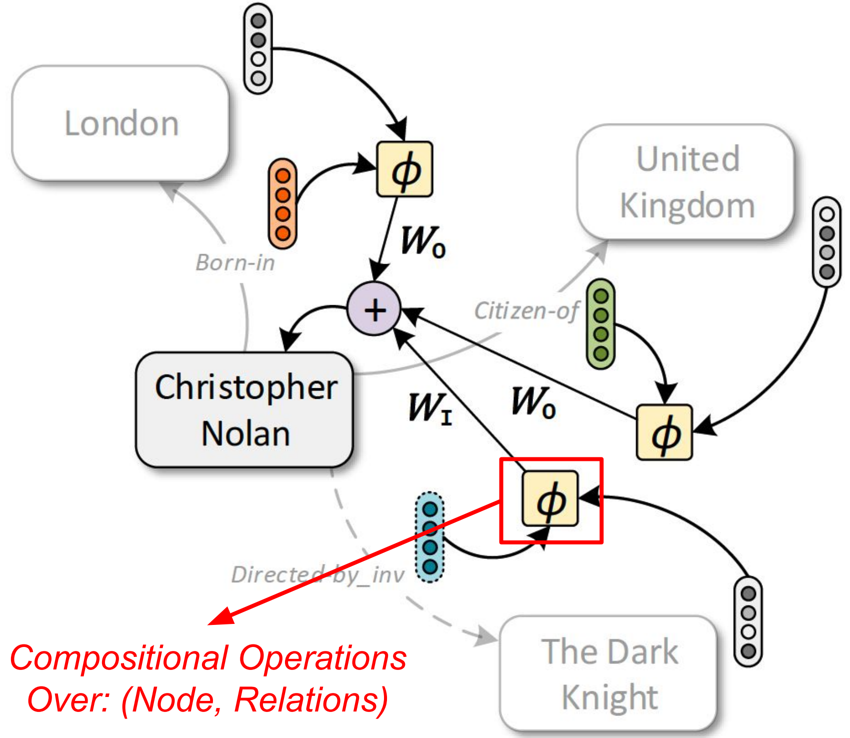
Overview


London
United Kingdom
Born-in
Born-in_inv
Citizen-of_inv
Citizen-of
Christopher Nolan
Directed-by
Directed-by_inv
The Dark Knight
Relational Graph with Embeddings
London
United Kingdom
ϕ
W_O
Born-in
+
Citizen-of
Christopher Nolan
W_I
W_O
ϕ
ϕ
Directed-by_inv
The Dark Knight
CompGCN Update

Overview



Relational Graph with Embeddings



CompGCN Update

Basic Notations

- Multi-Relational Graph:

$$\mathcal{G} = (\mathcal{V}, \mathcal{R}, \mathcal{E}, \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Z}})$$

Basic Notations

- Multi-Relational Graph:

$$\mathcal{G} = (\boxed{\mathcal{V}}, \boxed{\mathcal{R}}, \boxed{\mathcal{E}}, \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Z}})$$

Entity Node

Relations

Edges (Directed)

Basic Notations

- Multi-Relational Graph:

$$\mathcal{G} = (\boxed{\mathcal{V}}, \boxed{\mathcal{R}}, \boxed{\mathcal{E}}, \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Z}})$$

Entity Node

Relations

$$\mathcal{R}' = \mathcal{R} \cup \mathcal{R}_{inv} \cup \{\top\}$$
$$\mathcal{R}_{inv} = \{r^{-1} \mid r \in \mathcal{R}\}$$

Edges (Inverse & Self-Loops)

$$\mathcal{E}' = \mathcal{E} \cup \{(v, u, r^{-1}) \mid (u, v, r) \in \mathcal{E}\} \cup \{(u, u, \top) \mid u \in \mathcal{V})\}$$

Basic Notations

- Multi-Relational Graph:

$$\mathcal{G} = (\mathcal{V}, \mathcal{R}, \mathcal{E}, \boxed{\boldsymbol{\mathcal{X}}}, \boxed{\boldsymbol{\mathcal{Z}}})$$

Node Input Features

Initial Relation Features

$$\boldsymbol{\mathcal{Z}} \in \mathbb{R}^{|\mathcal{R}| \times d_0}$$

Representing Relations as Vectors

- Alleviates over-parameterization on relational graphs
- Allows utilizing any available relational features as initializations

Representing Relations as Vectors

- Alleviates over-parameterization on relational graphs
- Allows utilizing any available relational features as initializations
- **Compositional Operations:**

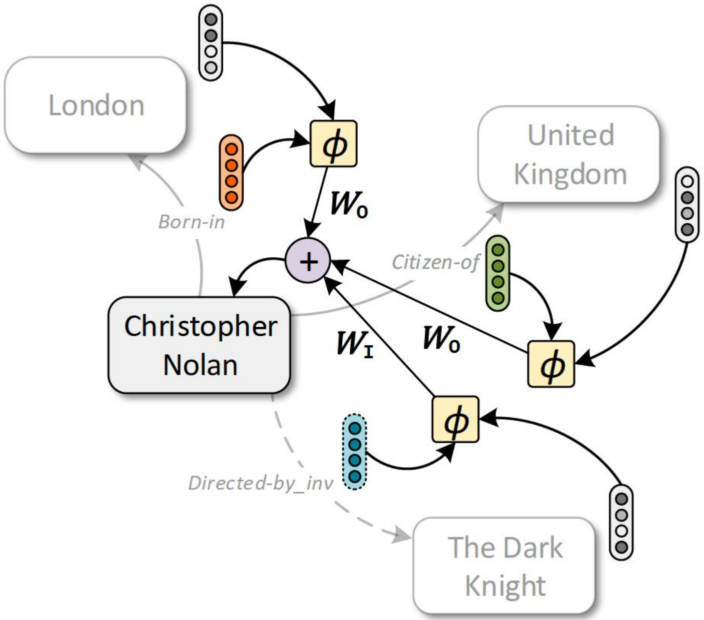
$$\boldsymbol{e}_o = \phi(\boldsymbol{e}_s, \boldsymbol{e}_r)$$

$$\phi : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d$$

Non-Parameterized Operations

- Subtraction
- Multiplication
- Circular-Correlation $[\boldsymbol{a} \star \boldsymbol{b}]_k = \sum_{i=0}^{d-1} a_i b_{(k+i) \bmod d}$

Update Rule

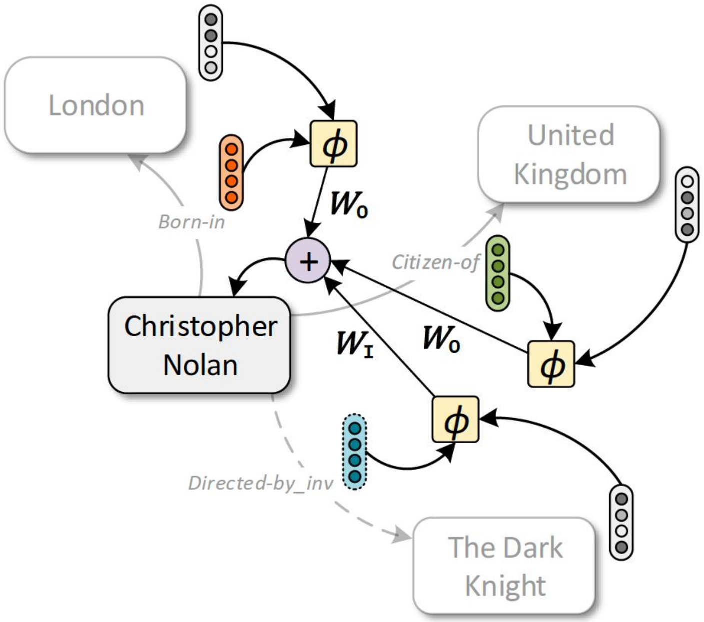


CompGCN Update

Standard Update Rule:

$$\boldsymbol{h}_v = f\left(\sum_{(u,r)\in\mathcal{N}(v)} \boldsymbol{W}_r \boldsymbol{h}_u \right)$$

Update Rule



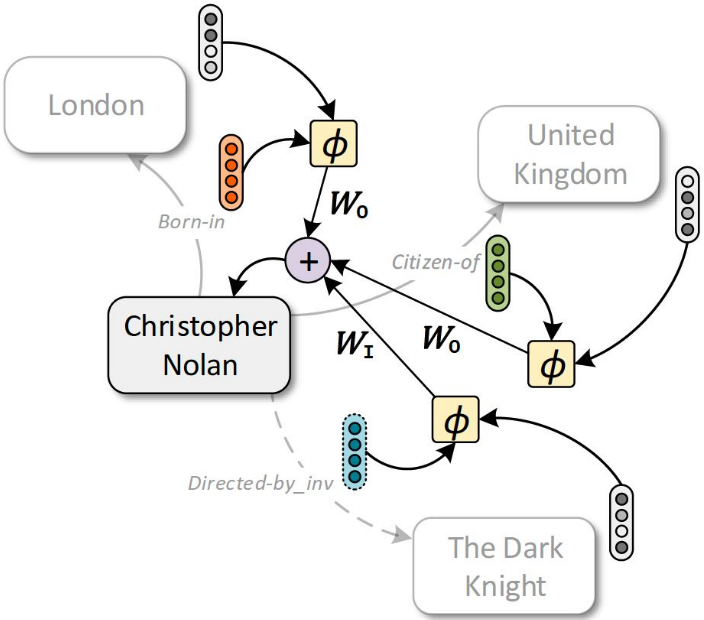
CompGCN Update

Standard Update Rule:

$$\boldsymbol{h}_v = f\left(\sum_{(u,r)\in\mathcal{N}(v)} \boldsymbol{W}_r \boldsymbol{h}_u\right)$$

Suffers from over-parameterization as each r needs its W!

Update Rule



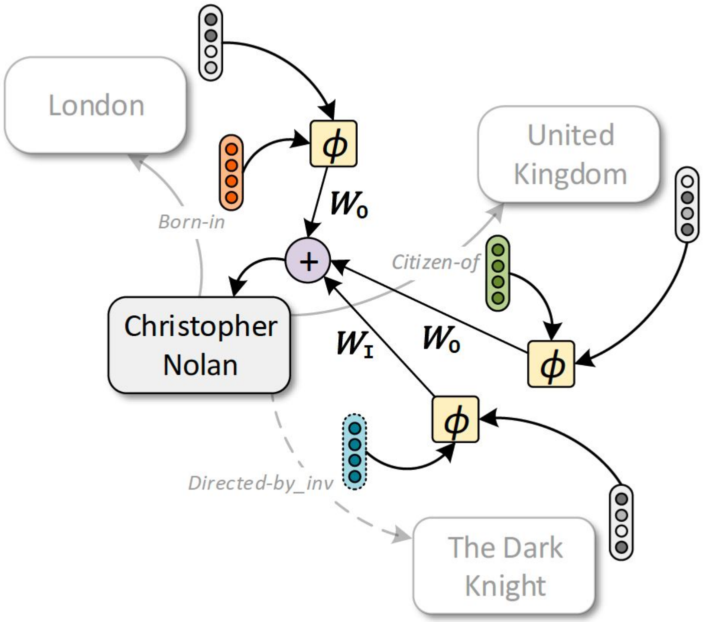
CompGCN Update

Standard Update Rule:

$$h_v = f\left(\sum_{(u,r)\in\mathcal{N}(v)} W_r h_u\right)$$

$$h_v = f\left(\sum_{(u,r)\in\mathcal{N}(v)} W_{\lambda(r)}\phi(x_u, z_r)\right)$$

Update Rule



CompGCN Update

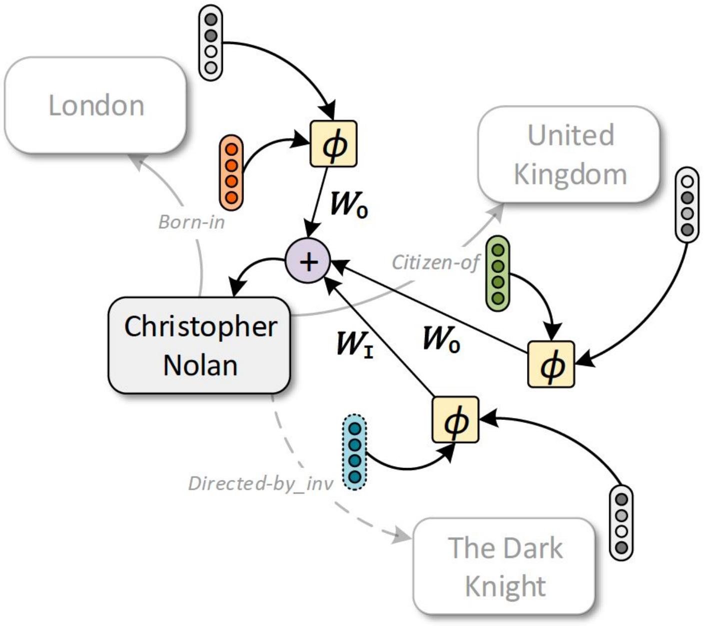
Standard Update Rule:

$$\boldsymbol{h}_v = f\left(\sum_{(u,r)\in\mathcal{N}(v)} \boldsymbol{W}_r \boldsymbol{h}_u\right)$$

$$\boldsymbol{h}_v = f\left(\sum_{(u,r)\in\mathcal{N}(v)} \boldsymbol{W}_{\lambda(r)} \phi(\boldsymbol{x}_u, \boldsymbol{z}_r)\right)$$

Number of feature dimensions: $(\mathcal{O}(|\mathcal{R}|d))$

Update Rule



CompGCN Update

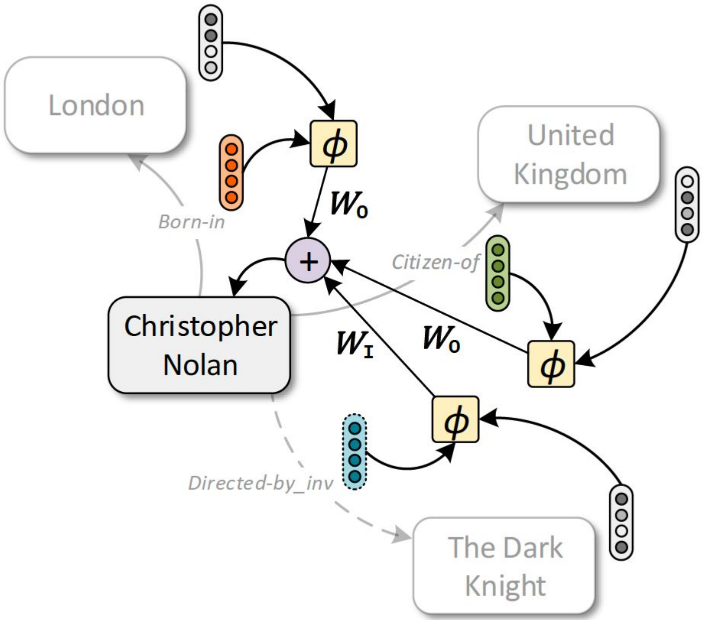
Relation-Specific Weights:

$$\boldsymbol{W}_{\mathrm{dir}(r)} = \begin{cases} \boldsymbol{W}_O, & r \in \mathcal{R} \\ \boldsymbol{W}_I, & r \in \mathcal{R}_{inv} \\ \boldsymbol{W}_S, & r = \top \; (\textit{self-loop}) \end{cases}$$

$$\boldsymbol{h}_v = f\left(\sum_{(u,r)\in\mathcal{N}(v)} \boldsymbol{W}_{\lambda(r)} \phi(\boldsymbol{x}_u, \boldsymbol{z}_r) \right) \quad \boxed{\lambda(r) = \mathrm{dir}(r)}$$

Number of feature dimensions: $(\mathcal{O}(|\mathcal{R}|d))$

Update Rule



CompGCN Update

Relation-Specific Weights:

$$\boldsymbol{W}_{\mathrm{dir}(r)} = \begin{cases} \boldsymbol{W}_O, & r \in \mathcal{R} \\ \boldsymbol{W}_I, & r \in \mathcal{R}_{inv} \\ \boldsymbol{W}_S, & r = \top \ (\textit{self-loop}) \end{cases}$$

$$\boldsymbol{h}_v = f\left(\sum_{(u,r)\in\mathcal{N}(v)} \boldsymbol{W}_{\lambda(r)} \phi(\boldsymbol{x}_u, \boldsymbol{z}_r) \right)$$

$$\boxed{\lambda(r) = \mathrm{dir}(r)}$$

Relational Embedding Update:

$$\boldsymbol{h}_r = \boldsymbol{W}_{\mathrm{rel}} \boldsymbol{z}_r \quad \boldsymbol{W}_{\mathrm{rel}} \in \mathbb{R}^{d_1 \times d_0}$$

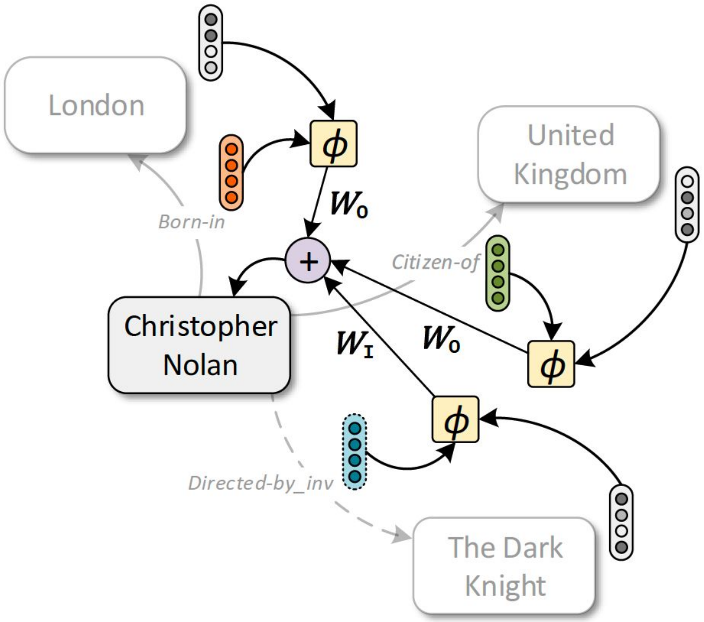
Scaling the Number of Relations

- Relational Vectors are:
  - **Linear Combinations** of a set of *learnable basis vectors*

$$\boldsymbol{z}_r = \sum_{b=1}^{\mathcal{B}} \alpha_{br} \boldsymbol{v}_b \qquad \{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_{\mathcal{B}}\}$$

Learnable Scalar Weights: $\alpha_{br} \in \mathbb{R}$

Stacking CompGCN (k-Layers)



CompGCN Update

Node Embedding Update:

$$\boldsymbol{h}_v^{k+1} = f\left(\sum_{(u,r)\in\mathcal{N}(v)} \boldsymbol{W}_{\lambda(r)}^{k} \phi(\boldsymbol{h}_u^k, \boldsymbol{h}_r^k) \right)$$

Relational Embedding Update:

$$\boldsymbol{h}_r^{k+1} = \boldsymbol{W}_{\text{rel}}^{k}\, \boldsymbol{h}_r^{k}$$

GCN Comparisons

Methods	Node Embeddings	Directions	Relations	Relation Embeddings	Number of Parameters
GCN Kipf & Welling (2016)	✓				$\mathcal{O}(Kd^2)$
Directed-GCN Marcheggiani & Titov (2017)	✓	✓			$\mathcal{O}(Kd^2)$
Weighted-GCN Shang et al. (2019)	✓		✓		$\mathcal{O}(Kd^2 + K\|\mathcal{R}\|)$
Relational-GCN Schlichtkrull et al. (2017)	✓	✓	✓		$\mathcal{O}(\mathcal{B}Kd^2 + \mathcal{B}K\|\mathcal{R}\|)$
CompGCN (Proposed Method)	✓	✓	✓	✓	$\mathcal{O}(Kd^2 + \mathcal{B}d + \mathcal{B}\|\mathcal{R}\|)$

- **K:** Number of GCN Layers
- **d:** Embedding dimension
- **B:** Number of bases
- **|R|:** Total number of relations

Reduction of CompGCN

Proposition 4.1. COMPGCN *generalizes the following Graph Convolutional based methods:* ***Kipf-GCN*** *(Kipf & Welling, 2016),* ***Relational GCN*** *(Schlichtkrull et al., 2017),* ***Directed GCN*** *(Marcheggiani & Titov, 2017), and* ***Weighted GCN*** *(Shang et al., 2019).*

Proof. For Kipf-GCN, this can be trivially obtained by making weights $(\boldsymbol{W}_{\lambda(r)})$ and composition function (ϕ) relation agnostic in Equation 5, i.e., $\boldsymbol{W}_{\lambda(r)} = \boldsymbol{W}$ and $\phi(\boldsymbol{h}_u, \boldsymbol{h}_r) = \boldsymbol{h}_u$. Similar reductions can be obtained for other methods as shown in Table 2. □

Reduction of CompGCN

Methods	$\boldsymbol{W}^k_{\lambda(r)}$	$\phi(\boldsymbol{h}^k_u, \boldsymbol{h}^k_r)$
Kipf-GCN (Kipf & Welling, 2016)	$\boldsymbol{W}^k$	$\boldsymbol{h}^k_u$
Relational-GCN (Schlichtkrull et al., 2017)	$\boldsymbol{W}^k_r$	$\boldsymbol{h}^k_u$
Directed-GCN (Marcheggiani & Titov, 2017)	$\boldsymbol{W}^k_{\mathrm{dir}(r)}$	$\boldsymbol{h}^k_u$
Weighted-GCN (Shang et al., 2019)	$\boldsymbol{W}^k$	$\alpha^k_r \boldsymbol{h}^k_u$

Outline

- Introduction
- Related Works
- Background
- Main Method: CompGCN
- **Experiments**
- Conclusions & Summaries

Benchmarking Tasks

- Link Predictions
- Node Classifications
- Graph Classifications

Benchmarking Tasks

- **Link Predictions**
  - Inferring missing facts based on known facts in KGs
  - **Datasets:** FB15k-237 (FreeBase), WN18-RR (WordNet)
  - **Metrics:** MRR & MR & Hits@N
- **Node Classifications**
  - Predicting labels of nodes
  - Based on node features and connections
  - **Datasets:** MUTAG & AM
- **Graph Classifications**
  - Learns representations of graphs, and predicts the types of graphs
  - **Datasets:** MUTAG & PTC

Dataset Statistics

	Link Prediction		Node Classification		Graph Classification	
	FB15k-237	WN18RR	MUTAG (Node)	AM	MUTAG (Graph)	PTC
Graphs	1	1	1	1	188	344
Entities	14,541	40,943	23,644	1,666,764	17.9 (Avg)	25.5 (Avg)
Edges	310,116	93,003	74,227	5,988,321	39.6 (Avg)	29.5 (Avg)
Relations	237	11	23	133	4	4
Classes	-	-	2	11	2	2

GCN Baselines

- Relational-GCN (R-GCN) (Schlichtkrull et al., 2017)
- Directed-GCN (D-GCN) (Marcheggiani & Titov, 2017)
- Weighted-GCN (W-GCN) (Shang et al., 2019)

GCN Baselines

- **Relational-GCN (R-GCN) (Schlichtkrull et al., 2017)**
  - Relation-specific weight matrices
  - Linear combinations of a set of basis matrices
- **Directed-GCN (D-GCN) (Marcheggiani & Titov, 2017)**
  - Separate weight matrices for incoming, outgoing edges and self-loops
  - Relation-specific biases
- **Weighted-GCN (W-GCN) (Shang et al., 2019)**
  - Learnable scalar weights assigned to each relation
  - Multiplies the weights with an “incoming message”

Task-Specific Baselines

- **Link Predictions**
  - TransE, DistMult, KBGAN, ComplEx, … etc.
- **Node Classifications**
  - Feat, WL, RDF2Vec, … etc.
- **Graph Classifications**
  - Deep Graph CNN, Graph Isomorphism Network, … etc.

TransE Objective: $$\mathcal{L} = \sum_{(h,\ell,t)\in S} \sum_{(h',\ell,t')\in S'_{(h,\ell,t)}} \left[\gamma + d(\boldsymbol{h}+\boldsymbol{\ell},\boldsymbol{t}) - d(\boldsymbol{h'}+\boldsymbol{\ell},\boldsymbol{t'})\right]_+$$

Goals of Experiments

- Q1. CompGCN compared to existing methods on Link Predictions
- Q2. Compare different GCN encoders and compositional operators
- Q3. CompGCN scalability
- Q4. CompGCN performance on node & graph classification

Results - Link Predictions

	FB15k-237					WN18RR				
	MRR	MR	H@10	H@3	H@1	MRR	MR	H@10	H@3	H@1
TransE (Bordes et al., 2013)	.294	357	.465	-	-	.226	3384	.501	-	-
DistMult (Yang et al., 2014)	.241	254	.419	.263	.155	.43	5110	.49	.44	.39
ComplEx (Trouillon et al., 2016)	.247	339	.428	.275	.158	.44	5261	.51	.46	.41
R-GCN (Schlichtkrull et al., 2017)	.248	-	.417		.151	-	-	-		-
KBGAN (Cai & Wang, 2018)	.278	-	.458		-	.214	-	.472	-	-
ConvE (Dettmers et al., 2018)	.325	244	.501	.356	.237	.43	4187	.52	.44	.40
ConvKB (Nguyen et al., 2018)	.243	311	.421	.371	.155	.249	**3324**	.524	.417	.057
SACN (Shang et al., 2019)	.35	-	**.54**	**.39**	.26	.47	-	.54	.48	.43
HypER (Balažević et al., 2019)	.341	250	.520	.376	.252	.465	5798	.522	.477	.436
RotatE (Sun et al., 2019)	.338	**177**	.533	.375	.241	.476	3340	**.571**	.492	.428
ConvR (Jiang et al., 2019)	.350	-	.528	.385	.261	.475	-	.537	.489	**.443**
VR-GCN (Ye et al., 2019)	.248	-	.432	.272	.159	-	-	-	-	-
CompGCN (Proposed Method)	**.355**	197	**.535**	**.390**	**.264**	**.479**	3533	.546	**.494**	**.443**

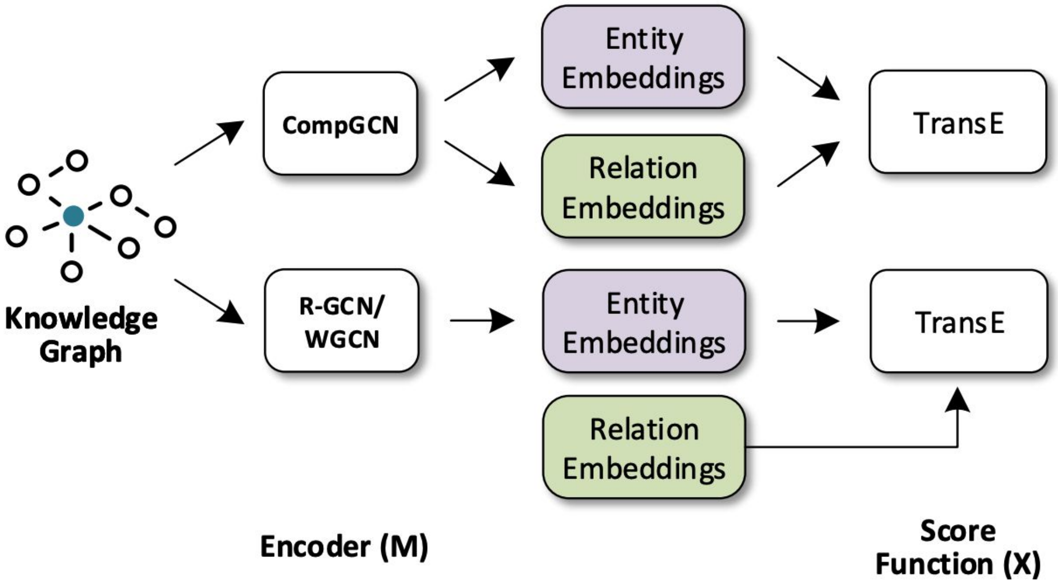
Results - Link Predictions

	FB15k-237					WN18RR				
	MRR	MR	H@10	H@3	H@1	MRR	MR	H@10	H@3	H@1
TransE (Bordes et al., 2013)	.294	357	.465	-	-	.226	3384	.501	-	-
DistMult (Yang et al., 2014)	.241	254	.419	.263	.155	.43	5110	.49	.44	.39
ComplEx (Trouillon et al., 2016)	.247	339	.428	.275	.158	.44	5261	.51	.46	.41
R-GCN (Schlichtkrull et al., 2017)	.248	-	.417		.151	-	-	-		-
KBGAN (Cai & Wang, 2018)	.278	-	.458		-	.214	-	.472	-	-
ConvE (Dettmers et al., 2018)	.325	244	.501	.356	.237	.43	4187	.52	.44	.40
ConvKB (Nguyen et al., 2018)	.243	311	.421	.371	.155	.249	**3324**	.524	.417	.057
SACN (Shang et al., 2019)	.35	-	**.54**	**.39**	.26	.47	-	.54	.48	.43
HypER (Balažević et al., 2019)	.341	250	.520	.376	.252	.465	5798	.522	.477	.436
RotatE (Sun et al., 2019)	.338	**177**	.533	.375	.241	.476	3340	**.571**	.492	.428
ConvR (Jiang et al., 2019)	.350	-	.528	.385	.261	.475	-	.537	.489	**.443**
VR-GCN (Ye et al., 2019)	.248	-	.432	.272	.159	-	-	-	-	-
CompGCN (Proposed Method)	**.355**	197	**.535**	**.390**	**.264**	**.479**	3533	.546	**.494**	**.443**

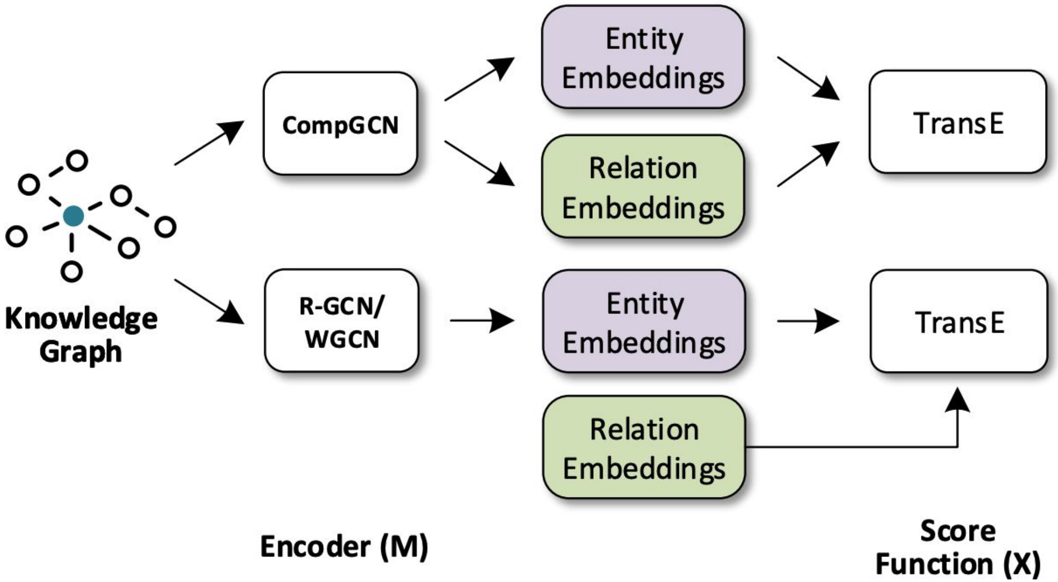
Results - Link Predictions

	FB15k-237					WN18RR				
	MRR	MR	H@10	H@3	H@1	MRR	MR	H@10	H@3	H@1
TransE (Bordes et al., 2013)	.294	357	.465	-	-	.226	3384	.501	-	-
DistMult (Yang et al., 2014)	.241	254	.419	.263	.155	.43	5110	.49	.44	.39
ComplEx (Trouillon et al., 2016)	.247	339	.428	.275	.158	.44	5261	.51	.46	.41
R-GCN (Schlichtkrull et al., 2017)	.248	-	.417		.151	-	-	-		-
KBGAN (Cai & Wang, 2018)	.278	-	.458		-	.214	-	.472	-	-
ConvE (Dettmers et al., 2018)	.325	244	.501	.356	.237	.43	4187	.52	.44	.40
ConvKB (Nguyen et al., 2018)	.243	311	.421	.371	.155	.249	**3324**	.524	.417	.057
SACN (Shang et al., 2019)	.35	-	**.54**	**.39**	.26	.47	-	.54	.48	.43
HypER (Balažević et al., 2019)	.341	250	.520	.376	.252	.465	5798	.522	.477	.436
RotatE (Sun et al., 2019)	.338	**177**	.533	.375	.241	.476	3340	**.571**	.492	.428
ConvR (Jiang et al., 2019)	.350	-	.528	.385	.261	.475	-	.537	.489	**.443**
VR-GCN (Ye et al., 2019)	.248	-	.432	.272	.159	-	-	-	-	-
COMPGCN (Proposed Method)	**.355**	197	**.535**	**.390**	**.264**	**.479**	3533	.546	**.494**	**.443**

GCN Comparisons



GCN Comparisons



- **Subtraction (Sub):** $\phi(e_s, e_r) = e_s - e_r$.
- **Multiplication (Mult):** $\phi(e_s, e_r) = e_s * e_r$.
- **Circular-correlation (Corr):** $\phi(e_s, e_r) = e_s \star e_r$

GCN Comparison Results - Link Predictions

- X + M (Y): method M is used for obtaining entity/relation embeddings with X as scoring function. Y is the composition operator.

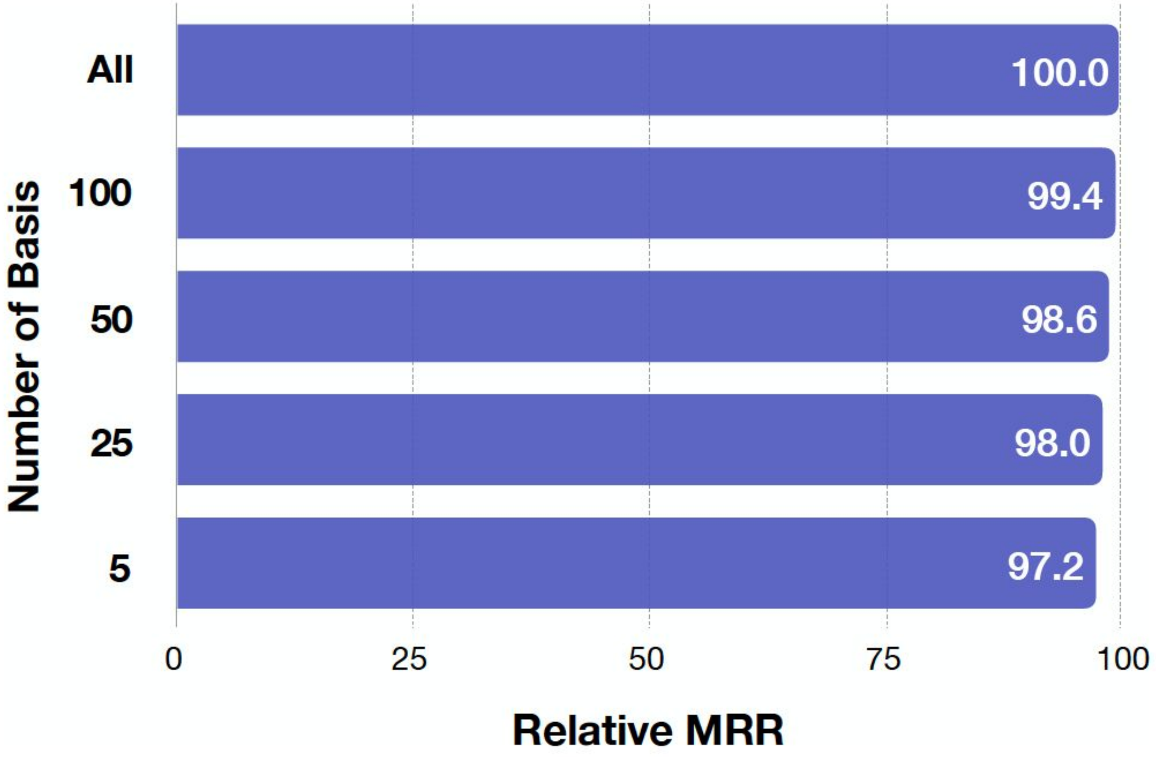
Scoring Function (=X) →	TransE			DistMult			ConvE		
Methods ↓	**MRR**	**MR**	**H@10**	**MRR**	**MR**	**H@10**	**MRR**	**MR**	**H@10**
X	0.294	357	0.465	0.241	354	0.419	0.325	244	0.501
X + D-GCN	0.299	351	0.469	0.321	225	0.497	0.344	200	0.524
X + R-GCN	0.281	325	0.443	0.324	230	0.499	0.342	197	0.524
X + W-GCN	0.267	1520	0.444	0.324	229	0.504	0.344	201	0.525
X + CompGCN (Sub)	0.335	**194**	0.514	0.336	231	0.513	0.352	199	0.530
X + CompGCN (Mult)	**0.337**	233	0.515	**0.338**	**200**	**0.518**	0.353	216	0.532
X + CompGCN (Corr)	0.336	214	**0.518**	0.335	227	0.514	**0.355**	197	**0.535**
X + CompGCN ($\mathcal{B} = 50$)	0.330	203	0.502	0.333	210	0.512	0.350	**193**	0.530

CompGCN Code Demo

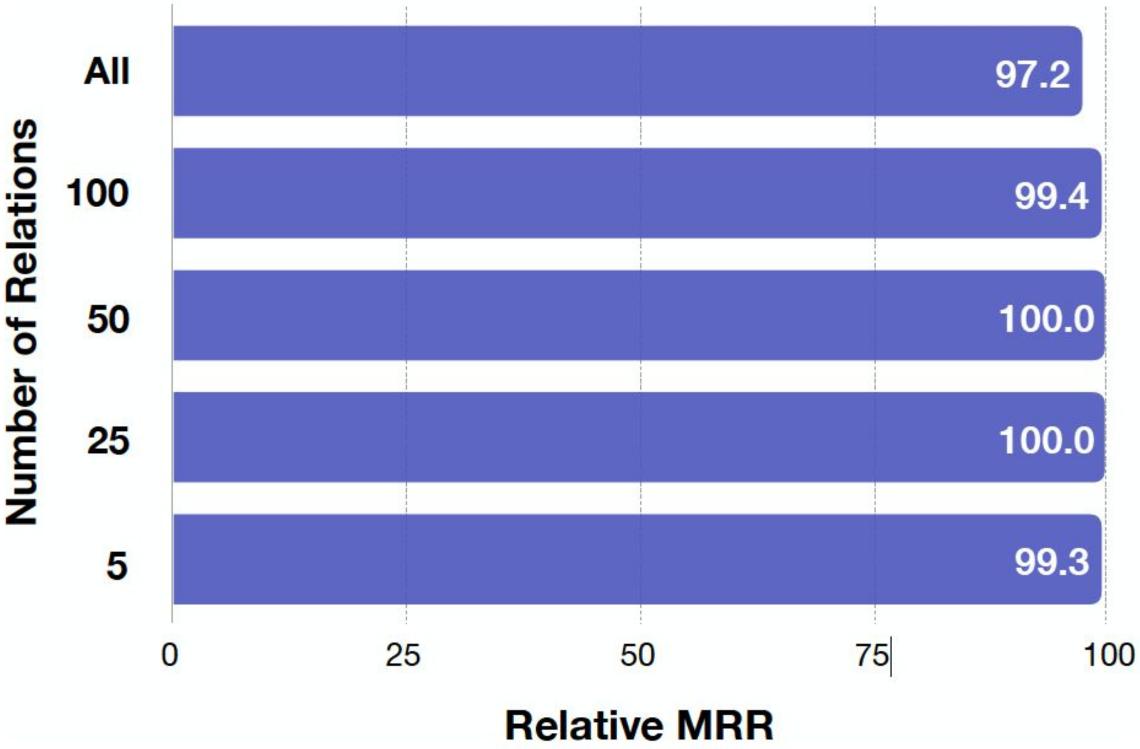
Evaluating the Scalability

- Effect of different number of basis vectors
- Scaling up different number of relations
- Compare to prior work R-GCN
  - Basis matrices instead of basis vectors

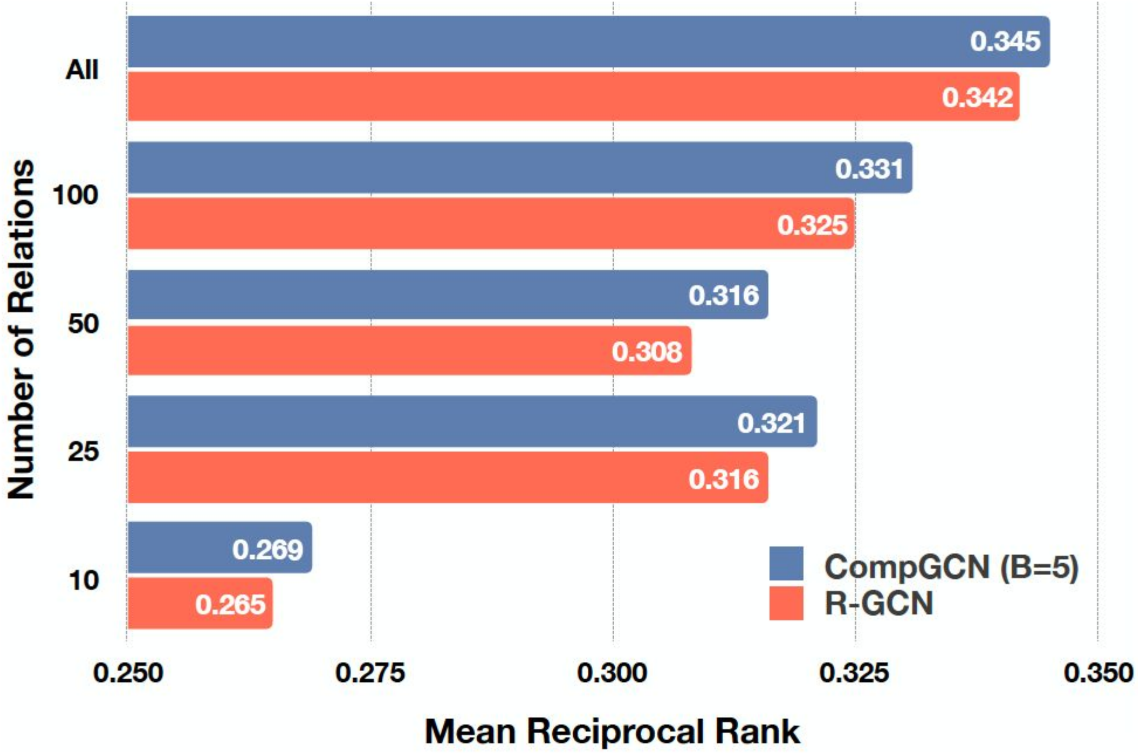
Number of Base Vectors



Number of Relations



Compared to R-GCN



Node & Graph Classifications

Node Classification

	MUTAG (Node)	AM
Feat*	77.9	66.7
WL*	80.9	87.4
RDF2Vec*	67.2	88.3
R-GCN*	73.2	89.3
SynGCN	74.8 ± 5.5	86.2 ± 1.9
WGCN	77.9 ± 3.2	90.2 ± 0.9
CompGCN	**85.3 ± 1.2**	**90.6 ± 0.2**

Graph Classification

	MUTAG (Graph)	PTC
PachySAN†	**92.6 ± 4.2**	60.0 ± 4.8
DGCNN†	85.8	58.6
GIN†	89.4 ± 4.7	64.6 ± 7.0
R-GCN	82.3 ± 9.2	67.8 ± 13.2
SynGCN	79.3 ± 10.3	69.4 ± 11.5
WGCN	78.9 ± 12.0	67.3 ± 12.0
CompGCN	89.0 ± 11.1	**71.6 ± 12.0**

Read Out: $$\boldsymbol{h}_{\mathcal{G}} = \frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{V}} \boldsymbol{h}_v$$

Outline

- Introduction
- Related Works
- Background
- Main Method: CompGCN
- Experiments
- **Conclusions & Summaries**

Summaries of CompGCN

- Multi-Relational Knowledge Graphs
- Representing Relations as Vectors
- Compositional Operators
- CompGCN Model
  - General
  - Alleviates over-parameterization
  - Scalable with basis vectors
- Extensive Experiments

CompGCN Future Works

- More complex compositional operators
  - Learnable eg. ConvE
  - Superior compositional operations
- Interpretable relational vectors

Paper: CompGCN at ICLR 2020

Published as a conference paper at ICLR 2020

COMPOSITION-BASED MULTI-RELATIONAL GRAPH CONVOLUTIONAL NETWORKS

Shikhar Vashishth[*†1,2] **Soumya Sanyal**[*1] **Vikram Nitin**[†3] **Partha Talukdar**[1]
[1]Indian Institute of Science, [2]Carnegie Mellon University, [3]Columbia University
svashish@cs.cmu.edu, {shikhar,soumyasanyal,ppt}@iisc.ac.in,
vikram.nitin@columbia.edu

ABSTRACT

Graph Convolutional Networks (GCNs) have recently been shown to be quite successful in modeling graph-structured data. However, the primary focus has been on handling simple undirected graphs. Multi-relational graphs are a more general and prevalent form of graphs where each edge has a label and direction associated with it. Most of the existing approaches to handle such graphs suffer from over-parameterization and are restricted to learning representations of nodes only. In this paper, we propose COMPGCN, a novel Graph Convolutional framework which jointly embeds both nodes and relations in a relational graph. COMPGCN leverages a variety of entity-relation composition operations from Knowledge Graph Embedding techniques and scales with the number of relations. It also generalizes several of the existing multi-relational GCN methods. We evaluate our proposed method on multiple tasks such as node classification, link prediction, and graph classification, and achieve demonstrably superior results. We make the source code of COMPGCN available to foster reproducible research.

Thank You & QA