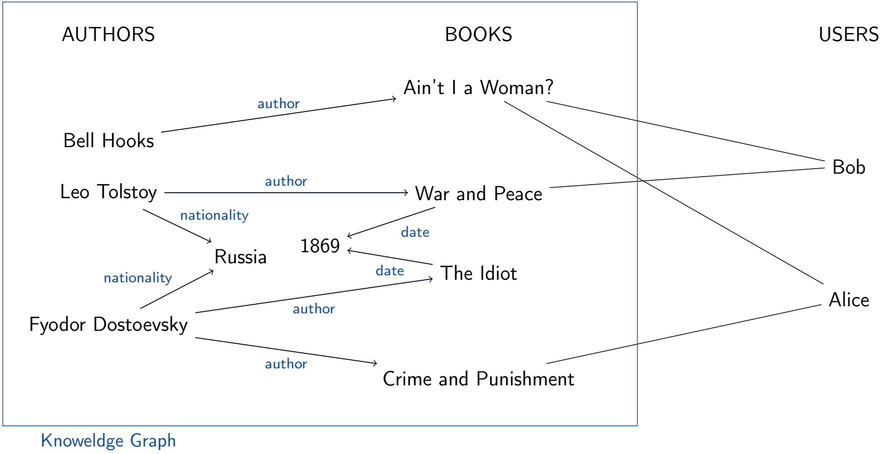
Explainable GNN-Based Models over Knowledge Graphs

David J. Tena Cucala[1]
Bernardo Cuenca Grau[1]
Egor V. Kostylev[2]
Boris Motik[1]

1.University of Oxford, 2.University of Oslo

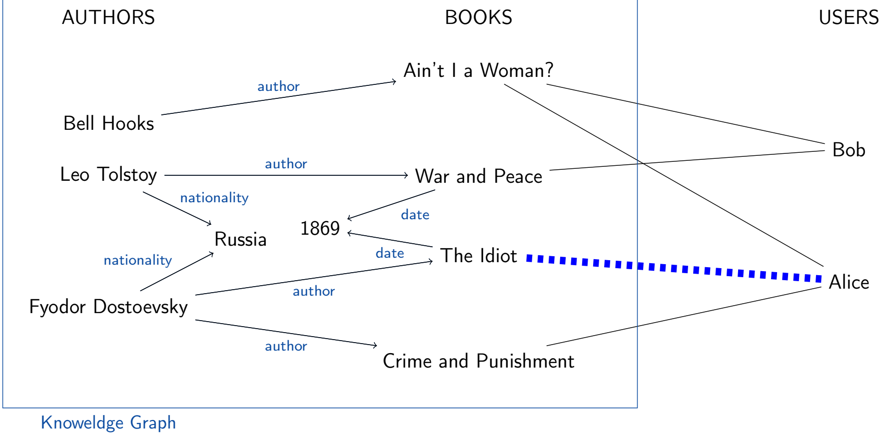
Motivation

Many applications of KGs involve ***learning a function*** from KGs to KGs.



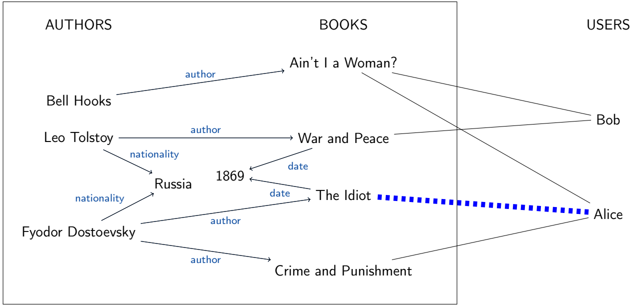
Motivation

Many applications in KGs involve ***learning a function*** from KGs to KGs.



Motivation

- The function is unknown, but examples are available.
- Learning must be ***noise-tolerant***.
- Functions can be realised using a ML model.
- *However*, such models are often difficult to interpret.
- *Symbolic rules* can provide explanations for model predictions.



For example, the suggestion *Recommend*(Alice, The_Idiot) can be explained by rule:

$$Author(x, y_1) \wedge Author(x, y_2) \wedge Likes(z, y_1) \rightarrow Recommend(z, y_2),$$

and KG facts

$$\begin{aligned} Author(\text{Dostoevsky}, \text{Crime\_And\_Punishment}) \\ Author(\text{Dostoevsky}, \text{The\_Idiot}) \\ Likes(\text{Alice}, \text{Crime\_And\_Punishment}). \end{aligned}$$

Our Contribution

A family of GNN-based transformations of KGs which:

- can be effectively trained in practice as usual, but
- all its predictions can be explained symbolically in terms of Datalog rules.

A Monotonic GNN-Based Transformation of KGs

Our approach is implemented in three steps:

1. encode the KG to a coloured graph G
2. apply the GNN model to G
3. decode the GNN model output to a new KG

The ***encoder*** introduces a vertex for each entity or pair of linked entities in the KG, an encodes KG triples encoded as feature vector components.

The ***GNN model*** is based on ***Monotonic Graph Neural Networks*** (MGNNs):

- Aggregate information from neighbour nodes via **maximum**.
- Model weights (but not the bias) must be **non-negative**.
- Activation and classification functions must be **monotonically increasing**.

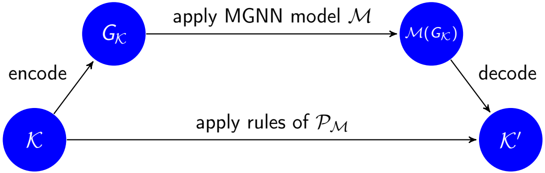
The ***decoder*** reverses the transformation by the encoder.

Key Results

Our transformation of KGs is ***monotonic under homomorphisms***.

No predictions (up to constant renaming) are lost by extending the input or renaming constants in it. This is a key property of Datalog reasoning.

For each MGNNs $\mathcal{M}$, there exists a Datalog program $\mathcal{P}_{\mathcal{M}}$ such that ***on any knowledge graph***, the predictions of $\mathcal{M}$ on the graph ***coincide*** with the inferences of $\mathcal{P}_{\mathcal{M}}$ on it.



We provide a correct and terminating ***algorithm*** for extracting $\mathcal{P}_{\mathcal{M}}$ from a given $\mathcal{M}$.

Evaluation

Can MGNN-based transformations be effectively trained and used in practice?

- Evaluation on Knowledge Graph completion (inductive setting)
- MGNNs were on par with state of the art approaches like DRUM and AnyBURL
- We extracted explicitly non-redundant rules of $\mathcal{P}_{\mathcal{M}}$ with at most *two* body atoms.
- Such rules accounted for almost all model predictions on the benchmarks.

Conclusion

MGNNs can be effectively trained in practice and perform as state-of-the-art models, while providing the added benefit of *full translatability* to Datalog rules.

Contact: david.tena.cucala@cs.ox.ac.uk