

UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Computer Science
Computer Science Curriculum

Gustav Nikopensius

Knowledge Graph Reasoning with Reinforcement Learning for Explainable Fact-checking

Bachelor's Thesis (9 ECTS)

Supervisor(s): Mayank Mohit
Sharma Rajesh

Tartu 2022

Knowledge Graph Reasoning with Reinforcement Learning for Explainable Fact-checking

Abstract:

Manual fact checking can not keep up with the pace at which false claims are produced and spread across the web. Computers are much faster at checking facts than humans. Automated fact checking usually involves comparing a fact claim to some set of knowledge. This comparison is oftentimes carried out by a machine learning algorithm. An effective way of representing knowledge that is also highly machine-readable is Knowledge Graphs. This study frames the problem of computational fact-checking as a reinforcement learning based knowledge graph reasoning problem. The experimental results reveal that reasoning over a knowledge graph is an effective way of producing human readable explanations in the form of paths and classifications for fact claims. The paths may aid fact-checking professionals with highly readable clues, improving trust and transparency in AI systems. The artificial intelligence aims to compute a path that either proves or disproves a factual claim, but does not provide a verdict itself. A verdict is reached by a voting mechanism which utilizes paths produced by the artificial intelligence. These paths can be presented to a human reader so that they themselves can decide whether or not the provided evidence is convincing or not. Understanding between AI and humans makes for trust and cooperation.

Keywords:

Explainable Machine Learning, Fact-checking, Knowledge graph, Reinforcement learning

CERCS:

P176 Artificial intelligence

Seletatav faktikontroll tugevdusõppepõhise teabegraafi-arutluskäiguga

Lühikokkuvõte:

Faktide käsitsi kontrollimine jääb valeväidete tootmise ja levitamise väledusele alla. Arvutid suudavad fakte inimestest palju kiiremini kontrollida. Automatiseeritud faktikontroll tähendab faktiväite kõrvutamist asjakohaste teadmistega. Kõrvutamisega tegeleb tihti masinõppe algoritm. Teabegraafid esitavad teadmisi efektiivselt ja need on ka masinloetavad. Käesolev uurimus sõnastab arvutusliku faktikontrolli probleemi tugevdusõppe põhise teabegraafil arutlemise probleemina. Katsetulemused näitavad, et teabegraafil arutlemist saab kasutada, et kontrollida fakte ja seletada kontrolli tulemust inimesele mõistetaval viisil arutluse teekonnana. Teekonnad võivad aidata faktikontrolli spetsialiste lihtsasti loetavate juhtnööridega, mis parandab tehisintellektisüsteemide läbipaistvust ja seega ka usaldust saavutatud otsuste vastu. Tehisintellekti ülesanne on leida teekond, mis lükkab ümber või kinnitab faktiväidet, ent tehisintellekt ei anna ise tõesusotsust.

Tõesusotsuseni jõutakse hääletusmehhanismiga, mis kasutab tehisintellekti toodetud teekondasid. Saadud teekondi võib esitleda inimesele, et ta saaks nende põhjal ise otsustada, kas esitatud tõendid on veenvad või mitte. Tehisintellekti ja inimeste vaheline mõistmine aitab kaasa usaldusele ja koostööle.

Võtmesõnad:

seletatav masinõpe, faktikontroll, teabegraaf, tugevdusõpe

CERCS:

P176 Tehisintellekt

Contents

1	Introduction	6
2	Preliminaries	8
2.1	Explainable Fact-checking	8
2.1.1	Fact-checking	8
2.1.2	Explainability	8
2.2	Knowledge Graph	9
2.2.1	Triple	9
2.2.2	Embeddings	9
2.2.3	KG example	10
2.3	Path	10
2.4	Reinforcement Learning	11
2.4.1	Environment	11
2.4.2	Policy Gradient	12
3	Related Work	13
3.1	Multi-hop reasoning	13
3.2	Fact-checking	14
3.2.1	Fact-checking with knowledge graphs	14
3.2.2	Explainable fact-checking	15
3.2.3	Explainable fact-checking with knowledge graphs	15
4	Methodology	16
4.1	Reinforcement Learning Agent	16
4.1.1	Goal and means of the Agent	16
4.1.2	The Agent’s strategy for reaching its goal	16
4.2	Policy	17
4.3	Environment	17
4.3.1	State	17
4.3.2	Action	18
4.3.3	Transition	18
4.3.4	Reward	19
4.4	Beam search and voting	19
5	Dataset	20
5.1	FB15K-237	20
5.2	NELL-995:	20

6	Evaluation	21
6.1	Experimental setting	21
6.1.1	Implementation Details	21
6.1.2	Hyperparameters	21
6.1.3	Evaluation protocol	22
6.2	Results	22
6.2.1	Basis for interpreting the results	22
6.2.2	Formatting the results	23
6.2.3	Fact checking	23
6.2.4	Explainability	25
6.2.5	Results discussion	27
7	Conclusion	28
	References	35
	Appendix	36
	I. Figures	36
	II. Licence	37

1 Introduction

Social media has been used by billions to communicate and millions of people get their news from social media as well [1]. However, this high level of communication has also brought forth a high number of misinformation spread on social media [2].

In the past, a lot of work has been done to combat misinformation using machine learning techniques. The majority of such techniques handles the context of the misinformation, e.g what kind of users propagate the misinformation, or processes the content with NLP techniques to derive clues from the way the misinformation is written. Only a small proportion of misinformation detection is concerned with checking the facts. Some of this work is done with knowledge graphs (KG), structured representations of knowledge, to also explain the results of the fact checking, such as [3–5].

In this thesis, we have employed a reinforcement learning (RL) agent to produce explanations in the form of relevant knowledge from the KG in the form of a path. The paths are arrived at by a technique from the field of KG reasoning known as multi-hop reasoning where an RL agent hops over facts in an attempt to connect two nodes that correspond to real world entities. The path describes the relation between those two entities and this path is used to classify a factual statement as true or false and also provides as a human readable explanation to the classification result. We use two KGs that are commonly used in multi-hop reasoning tasks to evaluate the performance of our proposed model. The KGs represent real world relationships, sets of true facts are gathered from these KGs and corresponding subsets of false facts are generated to simulate false facts. The results show that a multi-hop reasoning model can successfully check facts and provide highly human readable explanations to the classifications.

The major highlights of this work are:

1. We are the first to consider a multi-hop reasoning agent for explainable fact checking.
2. The reasoning Agent finds explanations and basis for classification from a structured knowledge representation which is not directly tied to any specific fact checking task.

The rest of the thesis is organized as follows:

1. Chapter 2 Preliminaries discusses the major concepts and keywords used within the thesis.
2. Chapter 3 Related Works gives a comprehensive overview of KG reasoning and fact checking, KG-related and explainable.
3. Chapter 4 Methodology formulates the implementation of the reinforcement learning technique and KG based environment and the Beam search based voting mechanism.

4. Chapter 5 Dataset describes the KGs used in the work.
5. Chapter 6 Evaluation describes the experimental setup, parameters and evaluation protocol, and discusses the results of the experimentation.
6. Chapter 7 Conclusion summarizes the method and results of the work and introduces directions for future work.

2 Preliminaries

This section gives an overview of the main concepts. It covers relevant aspects of explainable fact-checking, knowledge graphs (KG) and reinforcement learning (RL). Explainable fact-checking is the field of study that this thesis is situated in, a combination of KG and RL is proposed as a contribution to the field of explainable fact-checking.

The task of fact-checking aims to provide a veracity verdict to a fact claim by comparing the claim to relevant existing knowledge. In this work, KGs are used to represent existing knowledge and RL is the means of extracting the relevant knowledge subset from the KG. The subset (a path) is specified as an ordered set of KG nodes and edges or entities and relations. The path can in turn be interpreted as an explanation for the veracity verdict as it is highly human readable and the basis of the verdict.

2.1 Explainable Fact-checking

Explainable fact-checking is the process of providing a verity verdict to a given claim with an explanation to the verdict. This paper deals with the notion of automated fact-checking which means the fact-checking and explaining is done computationally and not by a human.

2.1.1 Fact-checking

The task of fact-checking is to arrive at a veracity verdict for a given claim by referring to relevant existing knowledge. For example *Putin is the president of the USA* is a false claim. To disprove this claim, one would go on Wikipedia and find the relevant article about Putin, where it states that Putin is the president of Russia.

2.1.2 Explainability

Explainability aims to overcome the machine learning (ML) black box problem. Different methods have been explored to explain the workings of fact-checking ML, the main two being 1) explaining the working principles of the ML model in human readable terms, 2) using the ML model to generate an explanation (e.g with summarization algorithms). The work at hand provides an explanation in the form of the side product of the ML model's work procedure - the paths used to arrive at the veracity verdict. For the example of *Putin's claimed presidency of the USA* an explanation for the veracity verdict of one such claim would be *Putin is the president of Russia*.

2.2 Knowledge Graph

Knowledge graphs represent relations between real world entities like people, institutions, dates and places in the form of a directed graph. The graph nodes represent real world entities and edges connecting the nodes represent the relations between the entities.

The main concepts of a KG from the respect of the work at hand are entities, relations, triples and embeddings.

2.2.1 Triple

A triple represents a real world fact. It consists of two entities and a connecting relation.

$$\langle JoeBiden, presidentOf, USA \rangle$$

$$\langle source, relation, target \rangle$$

The two entities are referred to as the *source entity* and *target entity*, *subject* and *object*, *head* and *tail*. In graph terms the entities are referred to as *nodes* and the relation as the *edge*. In the context of KG specific RL tasks, such as the work at hand, the term *action* will oftentimes refer to relation.

2.2.2 Embeddings

An embedding represents the semantic features of a KG entity or relation in the form of a vector. To produce an embedding for each entity and relation in a KG a **scoring function** is used. Embeddings are generated randomly and negative examples - triples that contain KG elements, but do not belong to KG - are produced. Positive and negative examples are scored by the scoring function to predict whether a given example triple belongs to the KG. Embeddings are then fitted according to the accuracy of the scoring function.

Complex embedding ComplEx embedding introduced in [6] maps the entities and relations into complex vector space. The scoring function:

$$Re(\langle wr, es, \bar{eo} \rangle)$$

Real part of the hermitian dot product of the three vectors of a triple: the relation, the subject and objects entities. For the object entity, the conjugate vector is used. This is done to capture the anti-symmetric relationship of knowledge graph triples. For example the true triple

$$\langle Tallinn, capital, Estonia \rangle$$

is antisymmetric in the sense that

$$\langle Estonia, capital, Tallinn \rangle$$

is false. Estonia is not the capital of Tallinn.

During training of the KG embeddings, the ComplEx score for a triple is mapped to a sigmoid function and the result is used to predict whether the input triple belongs to the KG or not.

2.2.3 KG example

An example KG used for movie recommendations. Relationships between movies, actors, directors and genres are used to arrive at movie recommendations for people.

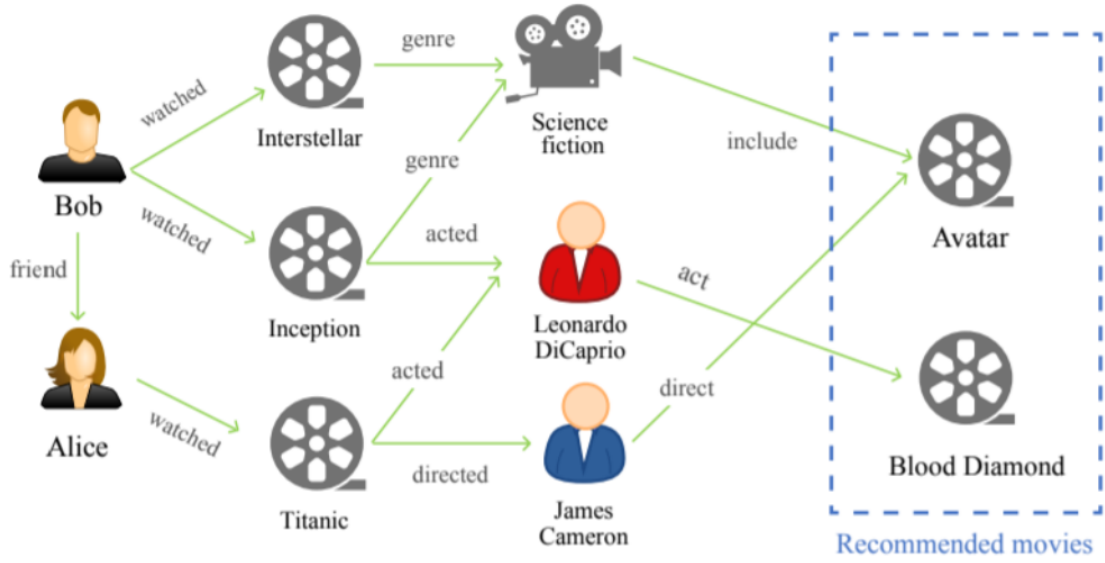


Figure 1. Example KG for movie recommendation [7]

2.3 Path

A path is a collection of entities and the interconnecting relations. This work assumes the notion that a simple claim can be expressed as an and-compound statement or in other words: a path can express the same knowledge that a triple can.

$$R(e_{source}, e_{target}) \iff R_1(e_{source}, e_{inter}) \wedge R_2(e_{inter}, e_{target})$$

For example the claim "Oona Chaplin is the granddaughter of Charlie Chaplin" or

$\langle OonaC., grandDaughter, CharlieC. \rangle$

Can also be expressed as the compound

$\langle OonaC., daughter, GeraldineC. \rangle \wedge \langle GeraldineC., Daughter, CharlieC. \rangle$

A path then is a chain of such triples connected triples.

The reason for using paths for verdicts and explainability instead of triples is that KGs

are incomplete which means many real world relations will be missing from a KG, but these same relations can be proven to be true or false by a path within the KG.

2.4 Reinforcement Learning

Reinforcement learning (RL) provides a means with which to extract relevant knowledge from the knowledge graph. In this work a reinforcement learning agent (the Agent) operates within the knowledge graph hopping from one node to another via connecting edges and records the traversed nodes and edges in the form of a path.

[8] provides a formalization of the RL problem: "The general RL problem is formalized as a discrete time stochastic control process where an agent interacts with its environment in the following way: the agent starts, in a given state within its environment $s_0 \in S$, by gathering an initial observation $\omega_0 \in \Omega$. At each time step t , the agent has to take an action $a_t \in A$. [...] It follows three consequences: (i) the agent obtains a reward $r_t \in R$, (ii) the state transitions to $s_{t+1} \in S$, and (iii) the agent obtains an observation $\omega_{t+1} \in \Omega$." The action taken by the Agent is chosen by a policy function. This work utilizes the policy gradient method which means that the policy function is parametrized by a neural network known in this context as a policy network. Environment and policy network are elaborated on below.

2.4.1 Environment

In RL it is common to model the environment as a Markov decision process (MDP) which controls the interaction between the Agent and the knowledge graph. In accordance with [8] The environment of an RL model is represented by a 4-Tuple $\langle S, \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ which corresponds to the state space, the action space, the transition function and the reward function. One iteration of the Agent's work process is called an episode.

State space The state space is a collection of all the possible states in which a RL model can be at in any given time step. State space can be fairly simple like all of the possible combinations in a game of tic-tac-toe or complex such as all possible game states on a chess board.

Action space The action space is a collection of all possible actions that the Agent can take. It is not the case that any action can be taken at any state. For example, in a game of tic-tac-toe one can not draw their symbol onto an already occupied square. There are two main ways of dealing with this: one is to give a large negative reward if an "illegal" move is attempted to discourage such moves in the future, the other way is to mask all "illegal" actions at a given time step making the Agent unable to choose those actions.

Transition function The transition function is what takes the state and action as input and outputs a new state. The transition function updates a state according to the action. In a game of tic-tac-toe this will be drawing a symbol into a chosen empty square and, if it was a winning move, drawing a line through three consecutive identical symbols. In a game of chess it might be moving the knight piece and checking the queen.

Reward function The reward function is what takes the state as input and outputs a number. This number is used to inform the Agent's decisions for future iterations. A reward is formally given at each time step, but in many problems, the reward is given only at the end and is then distributed over all preceding steps. For example in a game of chess the reward is given in the end (e.g 1 for a win, 0 for a tie and -1 for a loss) and once the game is through and it is time to analyze the importance of individual moves towards the result, each action, with respect to the state it was taken in, is either encouraged if it lead to a win or discouraged if it lead to a loss.

Episode In the beginning of an episode the Agent receives a state and reward as input, outputs a chosen action and the environment updates the state and reward.

2.4.2 Policy Gradient

Policy gradient methods optimize the parameters of a policy function to maximize the expected reward. The reward is produced by the reward function given the environment state. The gradient of the reward function is calculated with regard to each parameter and the parameter is adjusted by a learning rate to maximize the reward functions output. Incrementally adjusting the parameters of the policy function increases the likelihood of receiving a higher reward. The policy's function is to generate a probability distribution over the action space for a given state. The goal of the parameter update is to reduce the resulting probabilities of actions that lead to a negative result and increase the probabilities of actions that contributed to a positive result.

3 Related Work

This section describes related work the fields of multi-hop reasoning (Section 3.1) and fact-checking (Section 3.2). The multi-hop reasoning section gives an overview of the emergence of reinforcement learning techniques for relation and entity prediction in KGs. The fact-checking section introduces fact-checking works that employ knowledge graphs, explain the fact-checking model or do both. A figure 2 is provided in the Appendix that depicts the dynamics between the RL Agent and the environment.

3.1 Multi-hop reasoning

Knowledge graph reasoning (reasoning) infers new knowledge from existing knowledge in the knowledge graph, and detects errors in the knowledge graph. Large scale knowledge graphs are inherently incomplete. The main effort in reasoning research is to "fill in" the missing knowledge. This is done by either knowledge graph traversal known as multi-hop reasoning [9–25], or by mining logic rules based on KG semantics and using those rules for reasoning. [26–36] In this section we will have a closer look at major research in multi-hop reasoning as it is closely related to this thesis' method due to its traversing the knowledge graph. There are two main postulations in multi-hop reasoning: entity prediction and relation prediction. The former looks for a tail entity for a head entity and relation pair

$$\langle e_h, r, ? \rangle \rightarrow \langle e_h, r, e_t \rangle$$

and the latter looks for a missing relation between two entities.

$$\langle e_h, ?, e_t \rangle \rightarrow \langle e_h, r, e_t \rangle$$

The task of looking for a missing tail entity is more similar to this thesis' task of finding a path that leads from a given head entity to the desired (true) tail entity. The major difference is that this thesis' model receives the claimed tail entity as an input as well. The field of multi-hop reasoning began by building upon the Path Ranking Algorithm (PRA) introduced in [37] which ranks paths based on prior probabilities for a relation R connecting two entities x and y : $p(R(x, y))$. PRA gathers relational data from a graph using random walk and inputs the data as features to regularize a probability function using Logistic Regression. [20] Utilized this algorithm in a KG-specific entity prediction task. [21] argues that this approach is not suited for large scale knowledge bases due to the vast number of distinct paths and proposes operating on embedded vector representations of relations: training a recurrent neural network (RNN) to approximate a vector from an input of a relation path's vector representation. This proposal rests on research in embedding models like [38] where the vector embedding of a relation between two entities approximates the difference between the embedding of the two entities. In addition to reducing the scale of preprocessing the proposed method allows to predict

relations unseen during training time. [22] incorporates entities to the reasoning process and utilizes attention mechanisms to reason with multiple paths and for the first time proposes a single model that can reason for multiple relations unlike previous relation-per-models. [23] proposes reasoning in the continuous vector space of KG embeddings. Compared to [20–22] which rely on PRA to produce the initial paths, [23] utilizes a policy based reinforcement learning algorithm REINFORCE to learn paths in the KG. It takes in states as vector representations and outputs an action, a relation, to further the agents path towards the target entity. This approach opened up the floodgates in RL based multi-hop reasoning. The first of the following papers being [9] which builds on [23] by directing the task from relation prediction to link prediction, a more complicated task, and also building a model that can handle multiple reasoning entities, [23] being a model-per-relation task. Further work involves the problem of sparse rewards in complex KG environment with [10] introducing M-Walk, a method that combines Monte Carlo Tree Search with Q-learning enhanced RNN-s, [11] training a onehop embedding model to estimate the reward of unobserved relations, [17] providing reward supervision with rules generated by symbolic-based methods. Another issue that has been researched is KG incompleteness: Meta-KGR proposed in [12] generalizes meta-parameters of high-frequency relations to be adapted to less frequent relations. Another approach is to "fill in" the KG on the go. [14] generates missing relations on the fly from a background text corpus, [18] accomplishes this by leveraging embedding models' functionality.

3.2 Fact-checking

The need for computational fact-checking arises from the fact that false claims are generated and spread at an unprecedented level and human fact-checkers cannot keep up with this onflow of false claims. Computational fact-checking generally utilizes machine learning techniques and recently the field has diverted attention towards the interpretability of such machine learning solutions' predictions asking the question "Why?" to combat the machine learning black-box problem.

For the sake of brevity the overview of fact-checking related work is limited to fact-checking that utilizes knowledge graphs or attempts to explain the fact-checking process or result.

3.2.1 Fact-checking with knowledge graphs

[39] frames the problem of fact-checking as finding a connection between the subject and object of a fact statement: if the statement exists as an edge or a short path between the entities in a KG then the statement is true and false otherwise. The method utilizes a truth valuation function which takes into account the length of the path between the two entities and the generality of intermittent entities, generality meaning *how many statements it participates in*. [40] builds upon the notion of generality of intermittent entities but

considers multiple paths between the subject and object entities as a knowledge stream. Some of the work is concerned with mining the KG for patterns to be used in classifying a fact statement. [41] mines the KG for patterns associated with specific relations. For example if a city is the capital of a country then very often the parliament will be situated in that city. These patterns are used to classify the fact statement. [42] uses a supervised pattern discovery algorithm to find patterns relevant to training data. [43] builds upon this work by also taking into account the ontological closeness of entities, for example: "logician" and "theologian" are ontologically close to "philosopher" and this should be taken into account in pattern mining.

3.2.2 Explainable fact-checking

[44, 45] both formulate the task of explanation generation as a extractive-abstractive summarization task using transformer models introduced in [46]. [44] deals with a political news dataset from [47] and [45] explores facts that require specific expertise with a public health related dataset. Both datasets are annotated with gold standard justifications by journalist. [48] evaluates health related claims against scientific articles using transformer models.

3.2.3 Explainable fact-checking with knowledge graphs

[3] refers to knowledge graphs to assess claims and accompany these with human readable explanations. This is achieved through a combination of logical rule discovery and probabilistic answer set programming (ASP). The KG is mined for rules in the form of triples associated with specific relations and these relations' negations. For fact assessment the heads of these rules are replaced by claim values and the KG is scavenged for corresponding triples. Explanations and an assessment is derived from these triples through ASP. [4] uses rules to extract explanations from KGs and text resources. [5] mines for patterns for fact-discovery in the biomedical domain to produce human readable explanations to professionals for drug or treatment discovery.

4 Methodology

This section describes the Reinforcement Learning framework. The framework is composed of an environment in the form of a Markov decision process and a policy network. A figure (figure 2) is presented in the appendix, which depicts the dynamics between the Agent and the environment.

4.1 Reinforcement Learning Agent

4.1.1 Goal and means of the Agent

The Agent has one goal: to reach a specified target entity. This target entity is unknown to the Agent. What is known to the Agent is a fact claim in the form of a triple: $\langle \text{subject}, \text{relation}, \text{object} \rangle$ or $\langle \text{source}, \text{claimedRelation}, \text{claimedTarget} \rangle$.

The ground truth $\langle \text{source}, \text{claimedRelation}, \text{trueTarget} \rangle$ is stored within the environment as a triple. The *true target* being the aforementioned *target entity* for the Agent. The Agent also has intelligence in the form of a policy network that guides its actions towards reaching its goal. The policy network is regularly updated based on the feedback that the Agent receives from the environment based on its performance.

4.1.2 The Agent's strategy for reaching its goal

Knowledge graph traversal The Agent traverses the knowledge graph moving from one entity node to another with the goal of reaching the aforementioned *true target*. Since the true target is unknown to the Agent, then the task is not to predict which neighboring entity to move to, but rather which edge to follow, edge meaning action, meaning relation.

The Agent starts from the *source entity* presented within the fact claim triple. At each step, it considers the fact claim triple and the path traversed thus far and predicts which action will lead it towards the *true target*. More specifically, the policy function maps the environment state to a probability distribution over the action space, the distribution is masked according to which actions are available at the current state (meaning the Agent only considers edges that connect the current entity node) and a top probability action is chosen to update current state.

Since the policy function is initialized with random parameters, the Agent will start out making a lot of mistakes and mostly failing to achieve its goal. Luckily, the Agent has the ability to learn from feedback.

Updating the policy network Each time the Agent ends an episode of KG traversal, the final entity it arrives at is compared to the *true target* and the Agent receives a reward based on its performance. This reward is used to update the policy function. The idea

behind updating the policy function is to encourage the actions that lead to a high reward and discourage actions that lead to a low reward taking into account the states in which these actions were performed.

Since the action decisions are based on the output probability distribution of the policy function, then the policy function is the object of the updating process. More specifically, the parameters of the the policy function are updated. The mathematical decision made in the updating process is: with all other parameters and the input state being fixed values, how should each parameter be changed to increase the value of the reward function? This is calculating the gradient of the reward function with respect to a given parameter. Updating all parameters, according to the reward function's gradient and a specified *learning rate* which controls how big the parameter change should be, is known as **gradient ascent**.

4.2 Policy

The policy function is parametrized by a fully-connected neural network (NN). The NN is composed of one hidden layer with Rectified Linear Unit (ReLU) serving as the activation function and an output layer normalized by a softmax function. A stochastic gradient-based method is used for optimization (Adam) introduced in [49]. The policy function takes an environment state as input and maps it to a distribution over the entire action space. The goal of training is to optimize the parameters of the policy function in order to maximize the expected reward of the Agent.

4.3 Environment

The environment is modeled as a Markov decision process (MDP) which controls the interaction between the reinforcement learning agent (the Agent) and the knowledge graph. A 4-tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ represents the MDP. Each element is elaborated below.

4.3.1 State

The state consists of entities and relations. Two representations of the state are operated with in parallel: 1) a human readable collection of real world elements: entities and relations, and 2) a corresponding collection of embeddings - vector representations which capture the semantic information of the state elements.

The state contains the **fact claim**, **traversed path** and **current location of the Agent** which is itself an entity. The initial state consists of an input triple which is the fact claim. This triple is reversed so that

$$\begin{aligned} &\langle JoeBiden, presidentOf, USA \rangle \\ &\langle USA, presidentOf, JoeBiden \rangle . \end{aligned}$$

Thus the initial state can be thought of as constituting

$$\langle claimedTarget, claimedRelation, beginningOfPath, \dots \rangle .$$

The embedded state is padded with zero vectors since the state is used as input to the policy network which requires a fixed size input. These zero vectors are incrementally replaced by relation-entity embedding pairs during the Agent's path traversal process. The final entity in the set of these pairs is of particular importance: this is the entity that is used to give a veracity verdict for the fact claim. So in essence, once the path traversal is completed then the veracity verdict stands as:

$$claimedTarget == finalEntity$$

4.3.2 Action

Not only is the input size of the policy network fixed but so is the output size - it is always the same size as the **action space**. The policy network takes the state as input and computes probabilities for each action in the action space. This can be thought of as the Agent making an estimate: which actions are most likely to take me from the current entity to the desired final entity?

During training time, one of the top 3 most probable actions are chosen for each transition. The choice is random with a distribution that is calculated from the output of the policy network to fit the new sample size (3). The random choice is introduced in order to reduce the likelihood of the Agent getting stuck looking for sub-optimal paths brought forth by the local minima problem. In other words, the random choice is introduced to encourage the Agent to explore a bit more.

4.3.3 Transition

Choosing a new action entails a transition of state. The transition itself is a function $T : S \times A \rightarrow S$ which means that performing an action at a given state produces a new state. This new state is the input triple, a traversed path plus the choice action and a new current entity. The new current entity is a node in the KG that is connected to the previous current entity by the choice action.

It is not uncommon for an entity to have multiple neighbors that are all connected to it via the same relation. For example, the entity *The Beatles* will have 4 neighbors connected to it via the relation *memberOf*. Most previous literature concerning multi-hop reasoning picks a neighboring entity at random, the focus being on the performance of the RL agent, unsuited for the task of choosing a candidate from a variable size set of entities. In this work a heuristic is proposed for choosing a most likely entity to extend the path in the following subsection "Beam search and voting".

4.3.4 Reward

The agent is given a reward once the path traversal is complete. The reward is 1, if the final entity in the found path is equal to the true target. Otherwise, the reward is 0.

$$reward = \begin{cases} 1, & \text{if final entity} = \text{true target} \\ 0, & \text{otherwise} \end{cases}$$

For example, if the fact claim is

$$< JoeBiden, presidentOf, Russia >$$

then the Agent is given a reward of 1 if it arrives at *USA* and a reward 0 for anything else.

4.4 Beam search and voting

The accuracy of the RL model are heightened by combining it with the beam search algorithm (Beam search). This allows the Agent to traverse multiple paths stemming from one source entity. These paths are then compared in regard to which final entity they arrive at. A voting mechanism counts the final entities over all produced paths and uses the counts to arrive at a veracity verdict. The intuition behind this is that there are many ways to arrive at a true conclusion but incorrect (misleading or uninformative) paths tend to arrive at a different target each.

Beam numbers The beam numbers used in experimentation are 3, 5 and 10 with 10 yielding the highest accuracy scores for all fact checking task sets.

Heuristic The beam search algorithm makes use of the heuristic

$$\pi(a|s, \theta) + Re(< r, e_s, \bar{e}_c >)^{step}$$

where the first part of the addition is the policy network's probability for the chosen action and the second part is the ComplEx embedding's score function for input relation, source entity and the candidate entity. [6]

5 Dataset

In this section, we describe two widely used datasets, which we have used for our analysis.

5.1 FB15K-237

This dataset, a knowledge graph, introduced in [50] is derived from the FB15K dataset introduced in [38] which in turn is derived from the Freebase dataset introduced in [51] which combines structured databases with the diversity of collaborative wikis like Wikipedia.

The dataset was chosen because it appears in many multi-hop reasoning papers such as [9, 11–13, 15–19].

FB15K limits the number of entities just under 15000 most frequent entities. FB15K-237 further limits the number of relations to the 401 most frequent relations and decreases that number to a further 237 by removing duplicate and inverse relations.

For training three reasoning relations were used: *origin*, *nationality* and *language*. The train and test sets were prepared in [23]. All of the KG triples containing the specified triple were extracted from the knowledge graph. Based on these triples, a set of negative examples were formed by removing the initial tail entity from the triple and replacing it with a random entity of the same type. For example:

$$\langle \textit{Putin}, \textit{president}, \textit{Russia} \rangle \rightarrow \langle \textit{Putin}, \textit{president}, \textit{USA} \rangle .$$

This yields in 9 negative examples per 1 positive example.

5.2 NELL-995:

The dataset was introduced in [23]. A subset of the 995th iteration of the Never-ending Language Learning dataset introduced in [52], [23] removes relations with no reasoning value and keeps the triples with the top 200 most frequent relations to form the final dataset. The NELL-995 is often experimented on alongside FB15K-237. It has been used in the works This dataset has been used in [9–13, 16, 17, 19, 23] work.

The preprocessing for the NELL-995 dataset follows the same pipeline as for FB15K-237. The three chosen relations are: *athleteplaysinleague*, *athleteplaysforteam*, *worksfor*.

Table 1. Datasets

Dataset	#Ent	#Rel	#Fact
FB15k-237	14,505	237	272,115
NELL-995	75,492	200	154,213

The statistics for both the KGs are displayed in Table 1.

6 Evaluation

In this section, we first describe experimental settings (Section 6.1), followed by results of our evaluations in Section 6.2

6.1 Experimental setting

The model is evaluated on two equal magnitude knowledge graphs: FB15K-237 and NELL-995. NELL-995 has 5 times as many entities as FB15K-237 (75K to 15K) and FB15K-237 contains twice as many triples as NELL-995 (300K to 150K). This means FB15K-237 is a much denser graph than NELL-995 in the sense that it contains many more triples per entity. 1-20 vs NELL-995's 1-2. The number of relations is roughly the same. For each KG, 3 reasoning relations were chosen. These reasoning relations were *origin*, *tvLanguage*, *nationality* for FB15K-237, and *athletePlaysInLeague*, *athletePlaysForTeam*, *worksFor* for NELL-995.

"We use the TopK negative mode to generate negative samples for both train and test samples. For each positive samples, there are approximately 10 corresponding negative samples. Each negative sample is generated by replacing the true target entity t with a faked one t_0 in each triple (h, r, t) ."

Embeddings were trained for both datasets using ComplEx embedding from [6]. Training the embeddings was in 1000 epochs and produced vectors with length $k = 40$. [23] performed experiments on a fact prediction task. It is not specified what subsets resulted in the presented numbers. Still this could be considered as one Baseline to compare our results to.

6.1.1 Implementation Details

The knowledge graphs, train and test sets are from [23]. Relations that are present in the train or test sets are removed from the action space for the Agent. This is to keep the Agent from picking the exact same relation as in the input triple each time.

6.1.2 Hyperparameters

The embedding size for the KG models is set to 20. Xavier initialization from [53] is used for the KG embeddings and NN layers. The number of steps is set to 3. This number is arrived at after extensive manual and Agent explorations on the dataset - informative reasoning paths do not contain more than 3 relations in the given datasets. Adam optimization [49] is used for both the policy network and the KG embeddings. Learning rate for embeddings: $1e-4$ Each KG was trained for 1000 epochs Learning rate for policy network: $1e-5$ Each dataset was trained for 100 000 input triples. Three different Beam numbers were considered [3, 5, 10]

6.1.3 Evaluation protocol

The model receives an input fact claim in the form of a KG triple, based on the input triple, the Agent builds a path stemming from the source entity of the input triple, and the performance is evaluated by whether the path reached a desired target entity, the target entity is and Environment variable and the performance is evaluated once the Agent is finished with traversing the path.

6.2 Results

This section discusses three main points: 1) what are the parameters that are considered in interpreting the results, 2) how the results are formatted and 3) interpreting results for the two main tasks of the model: fact checking and explainability.

6.2.1 Basis for interpreting the results

In interpreting the results presented in tables 2-5, the relationships between different properties of the datasets and their respective results will be taken into account. The two KGs used in experimentation are different. The most prominent difference being how densely the KG entities are connected to each other. This is apparent from the ratio of triples per entity in the dataset statistics table 1.

1. FB15K-237 contains 18.8 triples per entity,
2. NELL-995 contains 2 triples per entity.

The density feature might have an influence on have raising the beam number for the search algorithm will result in accuracy.

The subsets that have been formed from these datasets are also different. The size of the input datasets range from 614 to 12660 triples. This might influence the Agent's ability to generalize knowledge from the presented knowledge.

The tasks contained within the subsets. For both KGs there are 3 single relation tasks which means the Agent only has to work with figuring out paths for one single relation. For each KG there is also a combined subset which combines all three relations. For example, the FB15K-237 Combined subset contains the relations "origin", "tvLanguage" and "nationality". In a sense, this diversity makes the Agent's task assignment more complicated, but it seems from the result that complexity actually assists the Agent. It might be so, because the input triple's relation acquires a meaning for the agent. In single relation tasks, the input triple's relation does not ever change thus does not inform the Agent's actions.

An important parameter to be considered is the beam number for the Beam search algorithm. An intuitive argument for raising the beam number might be that the number

of paths increases and thus the results will be more varied. This is not always the case. It is apparent in voting mechanism that more votes is not always better. And even in some cases for just finding one informative path that arrives at the correct target (search accuracy) more paths does not lead to an increase in accuracy but even a decrease. This might be attributed to the fact that the more paths the Agent considers, the more paths it also has to discard. For each step of the search mechanism, all paths that have been collected thus far are compared to each other.

6.2.2 Formatting the results

The results are expressed as four tables that are divided by the KG used and two accuracy metrics: search accuracy and voting accuracy.

Search accuracy Search accuracy considers all paths found by the beam search algorithm for a given beam number (b). This metric expresses the proportion of input triples for which at least one path reached the desired target entity.

Naturally, the higher the beam number, the more unique paths are considered for an input triple and the higher the probability that a true target will be reached. This is clearly expressed in table 2: considering a higher number of paths for an input triple increases the result of atleast on arriving at the target entity.

Voting accuracy Voting accuracy expresses the proportion of input triples for which the true target was predicted correctly. This is the proposed veracity verdict method's accuracy.

Beam number Three different beam numbers are considered: 3, 5 and 10. The beam number states how many paths are considered for one input triple.

6.2.3 Fact checking

The FB15K-237 dataset's results for search accuracy in table 2 seem to confirm intuitive conclusions: a larger input dataset and higher beam number result in higher accuracy. For each subset, the highest beam number (10) achieves the highest accuracy and as a rule the larger datasets also have higher accuracy. The exception being "origin" and "tvLanguage". "origin" being the slightly larger dataset that achieves a lower accuracy. The anomalous nature of the "origin" dataset is further exemplified by the fact that in the "voting" task, it is the only FB15K-237 subset that receives a lower accuracy for beam number 10 than for the other beam numbers.

Concerning the difference in results between search accuracy and voting accuracy, it is apparent, that voting for a a chosen target entity has a payoff in accuracy. A notable

subset is the combined dataset for FB15K-237 in table 3 where increasing the beam number has a significant positive impact for voting accuracy. This is a positive indicator for the models scalability.

The results of the overall path accuracy show that given a high enough beam number, the method will find a path to the true target with high probability. After voting, these results drop by a small margin. This is because a verdict must be reached from the found paths and sometimes, even though the correct target has been found, it might be undermined by a higher majority of an entity in the vicinity of the source entity.

The most apparent result is that voting on an answer reduces the accuracy: the correct target entity and path may have been arrived at, but it must also be found out, which one the correct path is. This has a toll on accuracy. This can be seen most obviously for FB15K-237's "origin" dataset which did not have a very good search accuracy to begin with and lost more than half of the correct results as a consequence of the voting.

Table 2. Search accuracy for FB15K-237

	input dataset size	accuracy, b=3	accuracy, b=5	accuracy, b=10
origin	852	0.464	0.53	0.661
tvLanguage	614	0.52	0.659	0.894
nationality	7134	0.866	0.907	0.921
combined	8600	0.807	0.886	0.943

Table 3. Voting accuracy for FB15K-237

FB15K-237	input dataset size	accuracy, b=3	accuracy, b=5	accuracy, b=10
origin	852	0.357	0.369	0.292
tvLanguage	614	0.463	0.472	0.642
nationality	7134	0.727	0.75	0.765
combined	8600	0.252	0.602	0.727

Table 4. Search accuracy for NELL-995

NELL-995	input dataset size	accuracy, b=3	accuracy, b=5	accuracy, b=10
athleteplaysinleague	12660	0.919	0.971	0.994
athleteplayssport	6781	0.903	0.853	0.868
worksfor	9669	0.748	0.78	0.825
combined	29110	0.485	0.575	0.611

Table 5. Voting accuracy for NELL-995

NELL-995	input dataset size	accuracy, b=3	accuracy, b=5	accuracy, b=10
athleteplaysinleague	12660	0.858	0.938	0.875
athleteplayssport	6781	0.871	0.828	0.83
worksfor	9669	0.604	0.59	0.509
combined	29110	0.45	0.467	0.464

6.2.4 Explainability

For explainability, for each KG, examples for a single relation task are discussed with examples of good and bad performance. In addition the results of one combined subset are discussed.

The following paths are the result of the NELL-995 Combined dataset consisting of relations “worksfor”, “athleteplayssport” and “athleteplaysinleague”. These are followed by two examples from the FB15K-237 "origin" subset. Six examples are presented in total:

1. the Agent arrives at the desired target entity with an informative path. This conclusion and the presented ground truth are actually false. The KG contains outdated information,
2. the Agent uses good reasoning, but arrives at a false conclusion, giving a true verdict for a false claim. The path itself is inconclusive and should be followed by further investigation,
3. the Agent finds a complex path that is both informative and arrives at the true target,
4. the Agent stays on the fence, not moving from the source entity for the entirety of the path.
5. the Agent finds two paths that could be considered true, but only one is considered correct by the ground truth interpretation.
6. the Agent finds two paths, one being the correct interpretation, the other supporting the claim indirectly, but being considered incorrect according to the ground truth.

KG contains outdated information

Claim: Dick Cheney works for retailstore Halliburton

Path: Dick Cheney –leads organization→ retailstore Halliburton

The Agent has found a correlation between a person leading an organization and working for the organization. This is an interesting successful case, since Dick Cheney

stopped leading Halliburton the day George W. Bush took him on as vice president candidate for the 2000 US presidential elections. This means that the ground truth presented by the dataset is actually false.

Inconclusive path

Claim: Coach John Gruden works for sportsteam Buccaneers.

Path: Coach John Gruden <–organization hired person– sportsteam Buccaneers

The Agent attempts to find a solution by considering the relation “organization hired person”. In fact Tampa Bay Buccaneers had hired John Gruden in 2002 and Gruden worked for the team until 2008. Thus, the claim that John Gruden works for the Buccaneers was true from 2002-2008, but is not true any more. The Agent has learned a correlation between the relations “worksfor” and “organizationhiredperson inv” meaning if an organization has hired a person then the person works for that organization. For this path, it might be reasonable for the Agent to also look for a relation between Gruden and the Buccaneers such as “organization fired person” or “contract ended” to further check the claim.

Informative path with complex reasoning

Claim: Coach Brendan Shanahan plays Hockey.

Path: Brendan Shanahan –plays for team→ the Devils –team plays sport→ Hockey

This is a good example of the Agent finding a path which does not just contain a (near) synonym for the claimed relation (like *works for* and *leads organization*), but finding a more complicated path. The Agent finds that Brendan Shanahan plays for the Devils hockey team and then the Agent finds out that the Devils play hockey.

Agent stays on the fence

Claim: Journalist Amir Taheri works for Los Angeles county.

Path: Amir Taheri → Amir Taheri A path like this is possible because of three reasons:

1. The length of the path is fixed to three relations, no more, no less.
2. The Agent must have the possibility to decide upon a target entity and “stick with it”
3. The Agent can decide to stay put at an entity for all three transitions.

Since most resultative paths only take one or two hops from entity to entity, then the action of staying put is very common, thus it’s always one of the top actions to choose for any state. With beam search also in place, it is very common that one path will result in the Agent staying put for the entirety of the path.

Two true paths with only one interpreted as correct

Claim: Ben Folds originates from Nashville.

Path 1: Ben Folds –place of birth→ Winston-Salem

Path 2: Ben Fold –location→ Nashville

These paths state that Ben Folds was born in Winston-Salem and that he is located in Nashville. For this specific case, the ground truth is that Ben Folds originates from Winston-Salem, so the first path leads to the true target and the second path reinforces the "false claim". The majority of explainable paths for the "origin" subset consist of one of two (near) synonymous relations: "place of birth" and "location". It is also common for one of the path relations to lead to a correct target entity and the other to lead to an incorrect one, the meaning of "origin" staying quite ambiguous. The subset is concerned with musical artists, so sometimes "origin" will mean the artists place of birth, other times it will mean the city in which they became established as a musician which will often coincide with the "location" relation.

Two true explainable paths with only one being considered as correct

Claim: Kylie Minogue originates from Australia.

Path 1: Kylie Minogue –place of birth→ Melbourne

Path 2: Kylie Minogue –place of birth→ Melbourne ←contains– Australia

The claim that Kylie Minogue originates from Australia is true in real life, but in regard to the ground truth, it is considered false. The example claims that Kylie Minogue originates from Australia which is true, but the ground truth is that Kylie Minogue originates from Melbourne, which is a city in Australia. Due to the setup of the veracity verdict mechanism, a path that concludes that Kylie Minogue originates from Australia is be considered false even though the path contains Melbourne and is very explainable.

6.2.5 Results discussion

The results show that deciding upon an entity to be considered for the veracity verdict has a trade-off in accuracy, since the correct entity may have been reached by the search algorithm, but not chosen by the voting mechanism. The trade-off is not a linear function, meaning tasks that receive a lower accuracy in finding a correct path suffer more severely from the results of voting.

Analysing common patterns in paths found by the Agent some issues arise to be considered in further research into the problem: 1) It is important to use an up to date KG to arrive at correct fact classifications, since some of the knowledge contained in a KG might be out of date and thus produce false results, 2) the verdict mechanism would improve by identifying and complimenting inconclusive paths, 3) the Agent should be higher motivated to move away from the source entity, 4) paths that do not end with the desired entity, but do support the ground truth should also be exploited to arrive at a correct conclusion, 5) classification could improve from learning to deal with ambiguous relationships like "origin".

7 Conclusion

This work proposed a combination of reinforcement learning and knowledge graphs to provide explainable fact classifications. The problem was formatted as a multi-hop reasoning problem enhanced by a beam search based voting mechanism, which uses the paths produced by a graph hopping reinforcement learning agent to vote on the veracity of a claimed fact statement. The results show that this sort of approach can be used for fact classification with varying accuracy dependent on the nature of the input claims, and it produces highly human readable explanations for classifications. Future directions should make use of more massive KGs and extract facts-to-check from Fake News datasets to provide a highly usable framework for real world misinformation detection tasks.

References

- [1] Pablo J. Boczkowski, Eugenia Mitchelstein, and Mora Matassi. Incidental news: How young people consume news on social media. In *HICSS*, 2017.
- [2] Martin Flintham, Christian Karner, Khaled Bachour, Helen Creswick, Neha Gupta, and Stuart Moran. Falling for fake news: Investigating the consumption of news via social media. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, page 1–10, New York, NY, USA, 2018. Association for Computing Machinery.
- [3] Naser Ahmadi, Joohyung Lee, Paolo Papotti, and Mohammed Saeed. Explainable fact checking with probabilistic answer set programming. In Maria Liakata and Andreas Vlachos, editors, *Proceedings of the 2019 Truth and Trust Online Conference (TTO 2019)*, London, UK, October 4-5, 2019, 2019.
- [4] Mohamed H. Gad-Elrab, Daria Stepanova, Jacopo Urbani, and Gerhard Weikum. Exfakt: A framework for explaining facts over knowledge graphs and text. In J. Shane Culpepper, Alistair Moffat, Paul N. Bennett, and Kristina Lerman, editors, *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, February 11-15, 2019*, pages 87–95. ACM, 2019.
- [5] Kara Schatz, Cleber C. Melo-Filho, Alexander Tropsha, and Rada Chirkova. Explaining drug-discovery hypotheses using knowledge-graph patterns. In Yixin Chen, Heiko Ludwig, Yicheng Tu, Usama M. Fayyad, Xingquan Zhu, Xiaohua Hu, Suren Byna, Xiong Liu, Jianping Zhang, Shirui Pan, Vagelis Papalexakis, Jianwu Wang, Alfredo Cuzzocrea, and Carlos Ordonez, editors, *2021 IEEE International Conference on Big Data (Big Data)*, Orlando, FL, USA, December 15-18, 2021, pages 3709–3716. IEEE, 2021.
- [6] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. June 2016.
- [7] Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He. A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, 34:3549–3568, 2022.
- [8] Vincent François-Lavet, Peter Henderson, Riashat Islam, Marc G. Bellemare, and Joelle Pineau. An introduction to deep reinforcement learning. *Foundations and Trends® in Machine Learning*, 11(3-4):219–354, 2018.
- [9] Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. Go for a walk and

- arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. November 2017.
- [10] Caiming Xiong Xi Victoria Lin, Richard Socher. M-walk: Learning to walk over graphs using monte carlo tree search. February 2018.
 - [11] Xi Victoria Lin, Richard Socher, and Caiming Xiong. Multi-hop knowledge graph reasoning with reward shaping. August 2018.
 - [12] Xin Lv, Yuxian Gu, Xu Han, Lei Hou, Juanzi Li, and Zhiyuan Liu. Adapting meta knowledge graph information for multi-hop reasoning over few-shot relations. August 2019.
 - [13] Ruiping Li and Xiang Cheng. DIVINE: A generative adversarial imitation learning framework for knowledge graph reasoning. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2642–2651. Association for Computational Linguistics, 2019.
 - [14] Cong Fu, Tong Chen, Meng Qu, Woojeong Jin, and Xiang Ren. Collaborative policy learning for open knowledge graph reasoning. August 2019.
 - [15] Marcel Hildebrandt, Jorge Andres Quintero Serna, Yunpu Ma, Martin Ringsquandl, Mitchell Joblin, and Volker Tresp. Reasoning on knowledge graphs with debate dynamics. January 2020.
 - [16] Guojia Wan, Shirui Pan, Chen Gong, Chuan Zhou, and Gholamreza Haffari. Reasoning like human: Hierarchical reinforcement learning for knowledge graph reasoning. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 1926–1932. ijcai.org, 2020.
 - [17] Deren Lei, Gangrong Jiang, Xiaotao Gu, Kexuan Sun, Yuning Mao, and Xiang Ren. Learning collaborative agents with rule guidance for knowledge graph reasoning. *CoRR*, abs/2005.00571, 2020.
 - [18] Xin Lv, Xu Han, Lei Hou, Juanzi Li, Zhiyuan Liu, Wei Zhang, Yichi Zhang, Hao Kong, and Suhui Wu. Dynamic anticipation and completion for multi-hop reasoning over sparse knowledge graph. *CoRR*, abs/2010.01899, 2020.
 - [19] Guojia Wan and Bo Du. Gaussianpath: A bayesian multi-hop reasoning framework for knowledge graph reasoning. In *Thirty-Fifth AAAI Conference on Artificial*

- Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 4393–4401. AAAI Press, 2021.
- [20] Ni Lao, Tom M. Mitchell, and William W. Cohen. Random walk inference and learning in A large scale knowledge base. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 529–539. ACL, 2011.
 - [21] Arvind Neelakantan, Benjamin Roth, and Andrew McCallum. Compositional vector space models for knowledge base inference. In *2015 AAAI Spring Symposia, Stanford University, Palo Alto, California, USA, March 22-25, 2015*. AAAI Press, 2015.
 - [22] Rajarshi Das, Arvind Neelakantan, David Belanger, and Andrew McCallum. Chains of reasoning over entities, relations, and text using recurrent neural networks. In Mirella Lapata, Phil Blunsom, and Alexander Koller, editors, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*, pages 132–141. Association for Computational Linguistics, 2017.
 - [23] Wenhan Xiong, Thien Hoang, and William Yang Wang. Deeppath: A reinforcement learning method for knowledge graph reasoning. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 564–573. Association for Computational Linguistics, 2017.
 - [24] Wenhua Chen, Wenhan Xiong, Xifeng Yan, and William Yang Wang. Variational knowledge graph reasoning. In Marilyn A. Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1823–1832. Association for Computational Linguistics, 2018.
 - [25] Heng Wang, Shuangyin Li, Rong Pan, and Mingzhi Mao. Incorporating graph attention mechanism into knowledge graph reasoning based on deep reinforcement learning. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language*

- Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2623–2631. Association for Computational Linguistics, 2019.
- [26] Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. Fast rule mining in ontological knowledge bases with AMIE+. *VLDB J.*, 24(6):707–730, 2015.
 - [27] Pouya Ghiasnezhad Omran, Kewen Wang, and Zhe Wang. Scalable rule learning via learning representation. In Jérôme Lang, editor, *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 2149–2155. ijcai.org, 2018.
 - [28] Vinh Thinh Ho, Daria Stepanova, Mohamed H. Gad-Elrab, Evgeny Kharlamov, and Gerhard Weikum. Rule learning from knowledge graphs guided by embedding models. In Denny Vrandečić, Kalina Bontcheva, Mari Carmen Suárez-Figueroa, Valentina Presutti, Irene Celino, Marta Sabou, Lucie-Aimée Kaffee, and Elena Simperl, editors, *The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part I*, volume 11136 of *Lecture Notes in Computer Science*, pages 72–90. Springer, 2018.
 - [29] Christian Meilicke, Melisachew Wudage Chekol, Daniel Ruffinelli, and Heiner Stuckenschmidt. Anytime bottom-up rule learning for knowledge graph completion. In Sarit Kraus, editor, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 3137–3143. ijcai.org, 2019.
 - [30] Fan Yang, Zhilin Yang, and William W. Cohen. Differentiable learning of logical rules for knowledge base reasoning. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 2319–2328, 2017.
 - [31] Tim Rocktäschel and Sebastian Riedel. End-to-end differentiable proving. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 3788–3800, 2017.
 - [32] Ali Sadeghian, Mohammadreza Armandpour, Patrick Ding, and Daisy Zhe Wang. DRUM: end-to-end differentiable rule mining on knowledge graphs. In Hanna M.

- Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d’Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 15321–15331, 2019.
- [33] Meng Qu, Junkun Chen, Louis-Pascal A. C. Xhonneux, Yoshua Bengio, and Jian Tang. Rnnlogic: Learning logic rules for reasoning on knowledge graphs. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.
- [34] Pasquale Minervini, Sebastian Riedel, Pontus Stenetorp, Edward Grefenstette, and Tim Rocktäschel. Learning reasoning strategies in end-to-end differentiable proving. In Pascal Hitzler and Md. Kamruzzaman Sarker, editors, *Neuro-Symbolic Artificial Intelligence: The State of the Art*, volume 342 of *Frontiers in Artificial Intelligence and Applications*, pages 280–293. IOS Press, 2021.
- [35] Wen Zhang, Bibek Paudel, Liang Wang, Jiaoyan Chen, Hai Zhu, Wei Zhang, Abraham Bernstein, and Huajun Chen. Iteratively learning embeddings and rules for knowledge graph reasoning. In Ling Liu, Ryen W. White, Amin Mantrach, Fabrizio Silvestri, Julian J. McAuley, Ricardo Baeza-Yates, and Leila Zia, editors, *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 2366–2377. ACM, 2019.
- [36] Meng Qu and Jian Tang. Probabilistic logic neural networks for reasoning. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d’Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7710–7720, 2019.
- [37] Ni Lao and William W. Cohen. Relational retrieval using a combination of path-constrained random walks. *Mach. Learn.*, 81(1):53–67, 2010.
- [38] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795, 2013.

- [39] Giovanni Luca Ciampaglia, Prashant Shiralkar, Luis M. Rocha, Johan Bollen, Filippo Menczer, and Alessandro Flammini. Computational fact checking from knowledge networks. *CoRR*, abs/1501.03471, 2015.
- [40] Prashant Shiralkar, Alessandro Flammini, Filippo Menczer, and Giovanni Luca Ciampaglia. Finding streams in knowledge graphs to support fact checking. In Vijay Raghavan, Srinivas Aluru, George Karypis, Lucio Miele, and Xindong Wu, editors, *2017 IEEE International Conference on Data Mining, ICDM 2017, New Orleans, LA, USA, November 18-21, 2017*, pages 859–864. IEEE Computer Society, 2017.
- [41] Baoxu Shi and Tim Weninger. Discriminative predicate path mining for fact checking in knowledge graphs. *Knowl. Based Syst.*, 104:123–133, 2016.
- [42] Peng Lin, Qi Song, Yinghui Wu, and Jiaxing Pi. Discovering patterns for fact checking in knowledge graphs. *ACM J. Data Inf. Qual.*, 11(3):13:1–13:27, 2019.
- [43] Peng Lin, Qi Song, and Yinghui Wu. Fact checking in knowledge graphs with ontological subgraph patterns. *Data Sci. Eng.*, 3(4):341–358, 2018.
- [44] Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. Generating fact checking explanations. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7352–7364. Association for Computational Linguistics, 2020.
- [45] Neema Kotonya and Francesca Toni. Explainable automated fact-checking for public health claims. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 7740–7754. Association for Computational Linguistics, 2020.
- [46] Yang Liu and Mirella Lapata. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China, November 2019. Association for Computational Linguistics.
- [47] Tariq Alhindi, Savvas Petridis, and Smaranda Muresan. Where is your evidence: Improving fact-checking by justification modeling. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 85–90, Brussels, Belgium, November 2018. Association for Computational Linguistics.

- [48] Mourad Sarrouiti, Asma Ben Abacha, Yassine Mrabet, and Dina Demner-Fushman. Evidence-based fact-checking of health-related claims. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 3499–3512. Association for Computational Linguistics, 2021.
- [49] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.
- [50] Kristina Toutanova and Danqi Chen. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66, Beijing, China, July 2015. Association for Computational Linguistics.
- [51] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Free-base: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, SIGMOD '08*, page 1247–1250, New York, NY, USA, 2008. Association for Computing Machinery.
- [52] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka, and Tom M. Mitchell. Toward an architecture for never-ending language learning. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI'10*, page 1306–1313. AAAI Press, 2010.
- [53] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, 2010.

Appendix

I. Figures

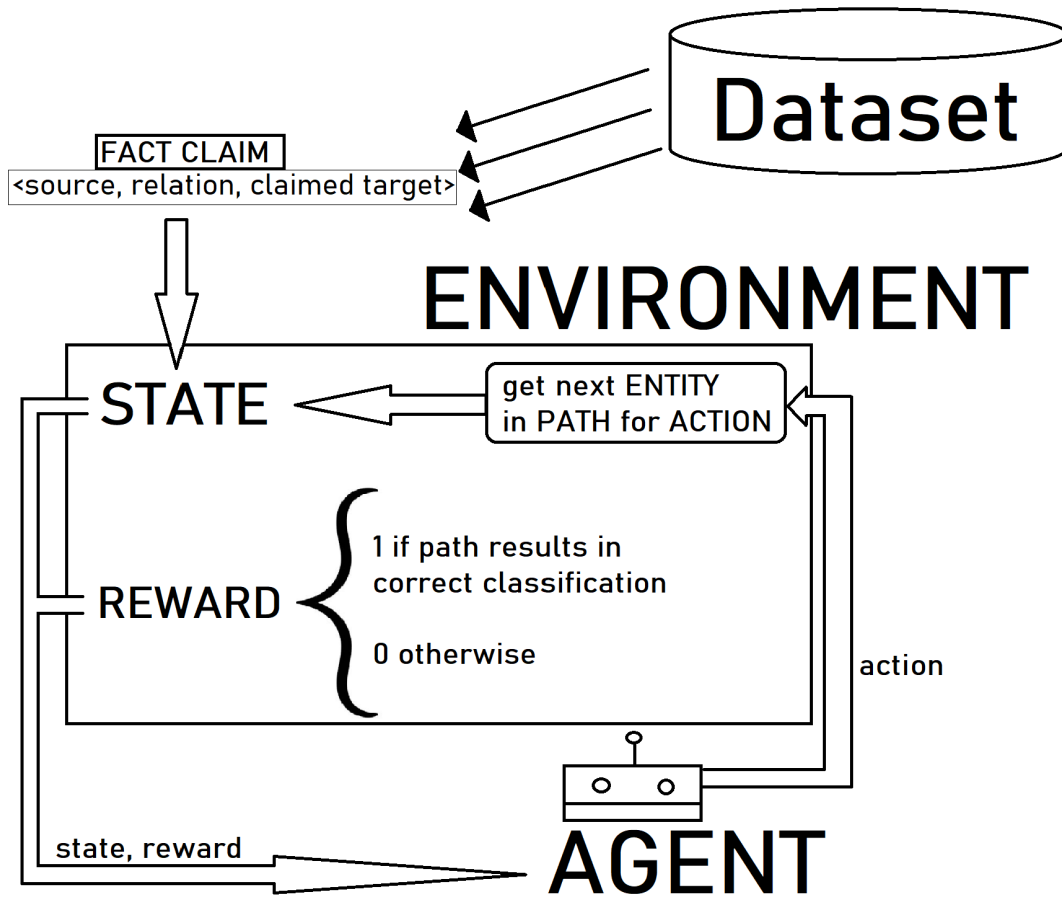


Figure 2. Agent-environment dynamics

II. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Gustav Nikopensius**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

Knowledge Graph Reasoning with Reinforcement Learning for Explainable Fact-checking,

supervised by Mohit Mayank and Rajesh Sharma.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.
3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.
4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Gustav Nikopensius

10/11/2022