Fact Checking via Path Embedding and Aggregation

Giuseppe Pirrò

Department of Computer Science, Sapienza University of Rome, Italy

SEMIFORM-2020
Virtual Workshop 02 November, 2020

The need for fact checking

"He doesn't have a birth certificate. He may have one, but there's something on that, maybe religion, ***maybe it says he is a Muslim****"*

The need for fact checking

"He doesn't have a birth certificate. He may have one, but there's something on that, maybe religion, ***maybe it says he is a Muslim****"*

"The movie Dune was directed by A. Jodorowki"

"No! Dune was directed by D. Lynch"

Fact
Check

The need for fact checking

"He doesn't have a birth certificate. He may have one, but there's something on that, maybe religion, ***maybe it says he is a Muslim****"*

"The movie Dune was directed by A. Jodorowki"

"No! Dune was directed by D. Lynch"

Shortcomings of existing approaches

- Require humans in the loop
- Mainly synctactic analysis
- Output a score but no evidence

The need for fact checking

"He doesn't have a birth certificate. He may have one, but there's something on that, maybe religion, ***maybe it says he is a Muslim****"*

"The movie Dune was directed by A. Jodorowki"

"No! Dune was directed by D. Lynch"

Fact Check

Shortcomings of existing approaches

- Require humans in the loop
- Mainly synctactic analysis
- Output a score but no evidence

Desiderata

- Automatize Fact Checking
- Consider the semantics of facts
- Information from related facts

Problem Statement and Contributions

Problem Statement

Given a fact of the form (subject, predicate, object):

- Collect semantic evidence (i.e., a graph of supporting facts)
- Provide an assessment of its truthfulness (i.e., an evidence score)

Problem Statement and Contributions

Problem Statement

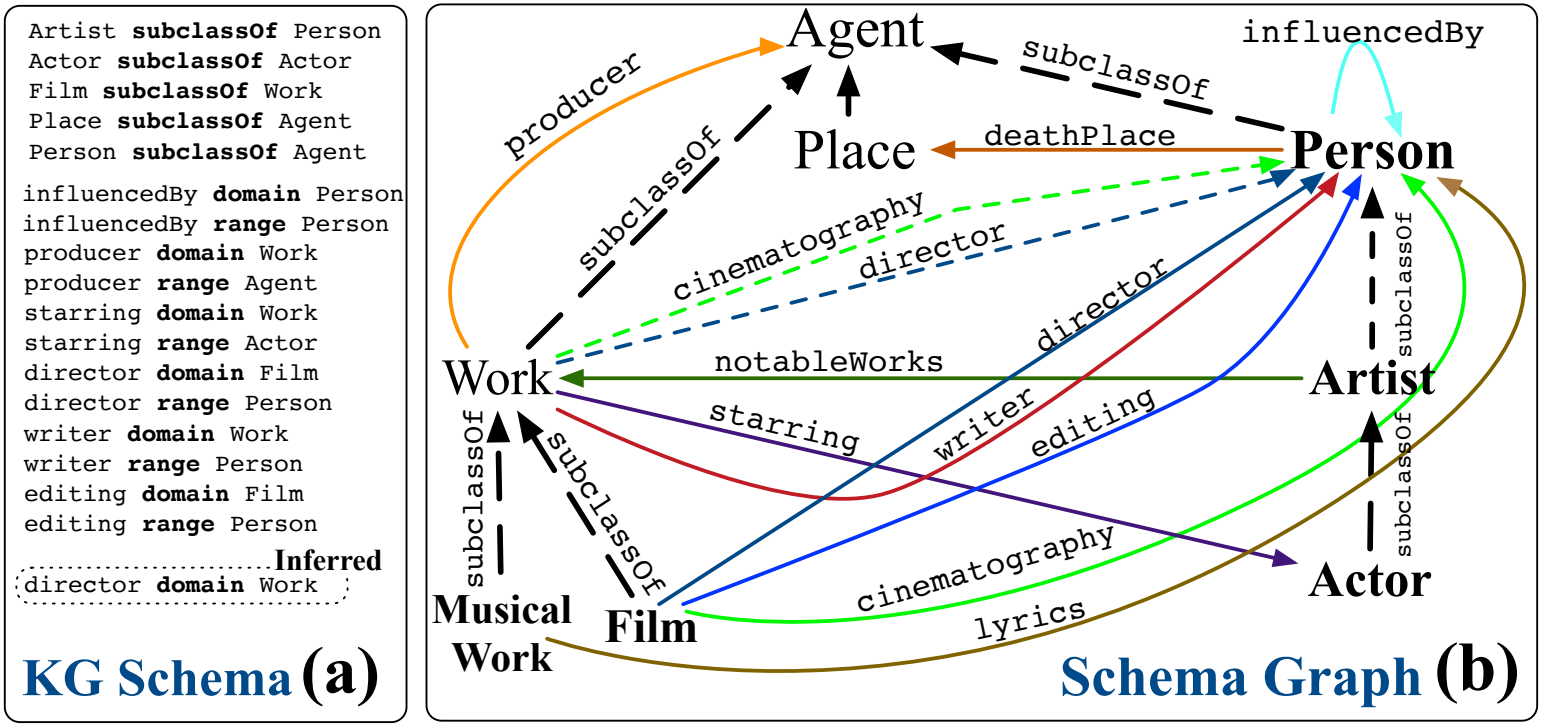
Given a fact of the form (subject, predicate, object):

- Collect semantic evidence (i.e., a graph of supporting facts)
- Provide an assessment of its truthfulness (i.e., an evidence score)

Contributions

- Using Knowledge Graphs (KGs) as a source of knowledge
  - Usage of the KG schema to create hyphoteses
  - Contextualization of facts by considering related facts
  - Verification of the hyphoteses in the KG data

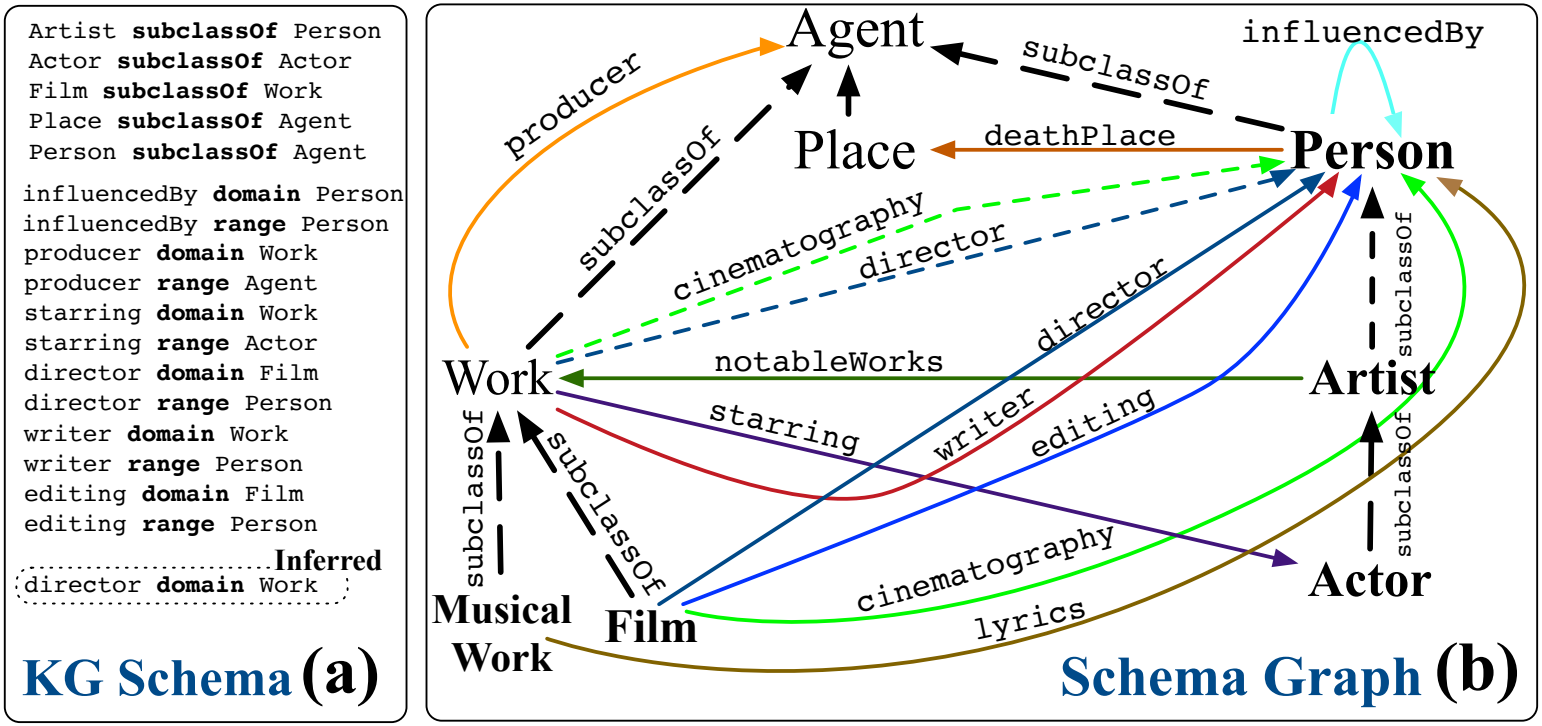
Knowledge Graph Schema



KG Schema

The schema gives a high level overview on the KG data

Knowledge Graph Schema



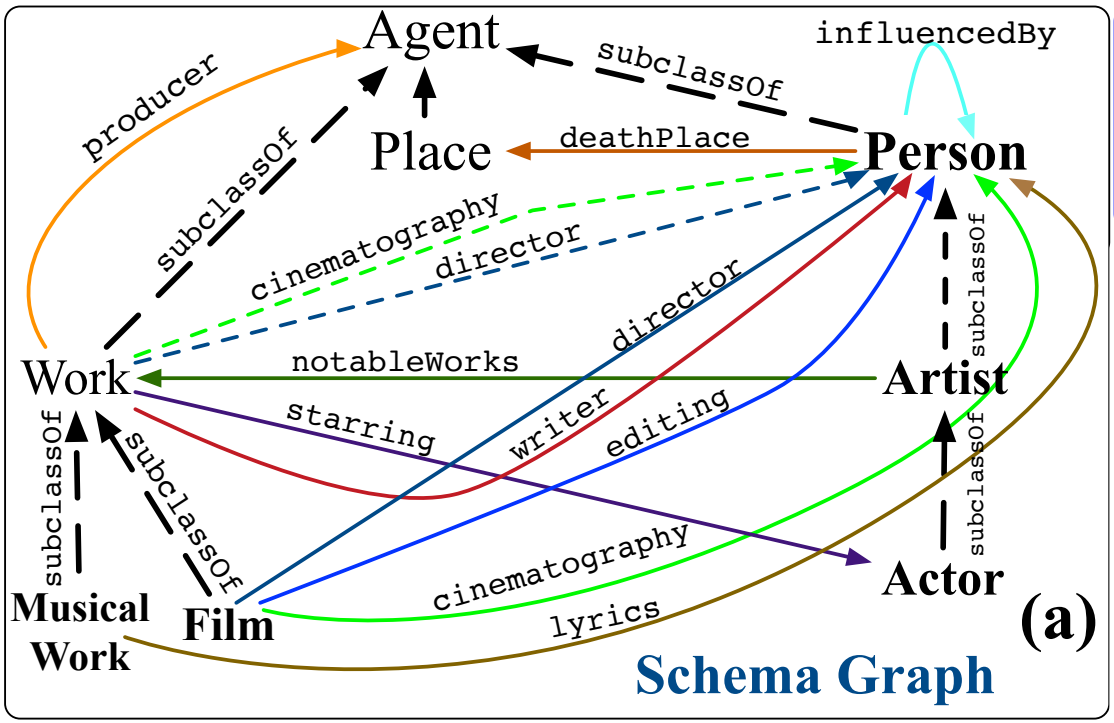
KG Schema

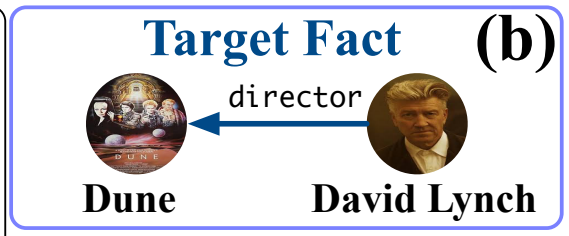
The schema gives a high level overview on the KG data

Schema Graph Construction

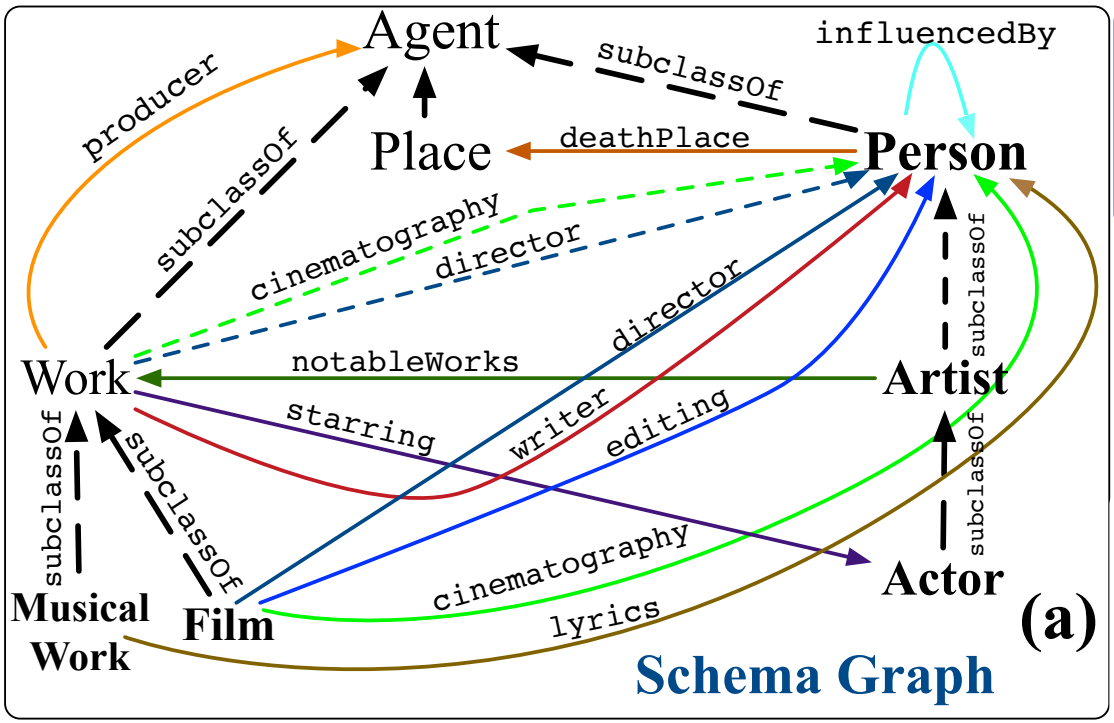
Add a p-edge between C_1 and C_2 if $domain(p)=C_1$ and $range(p)=C_2$

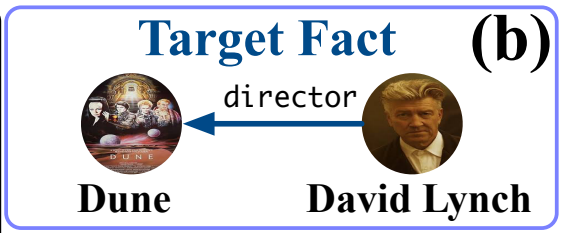
From facts to fact templates

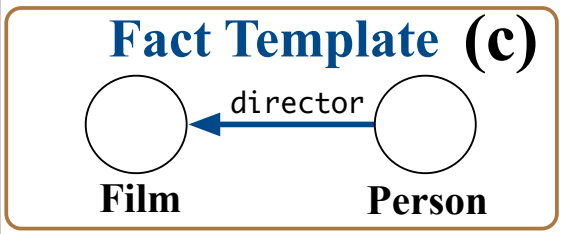




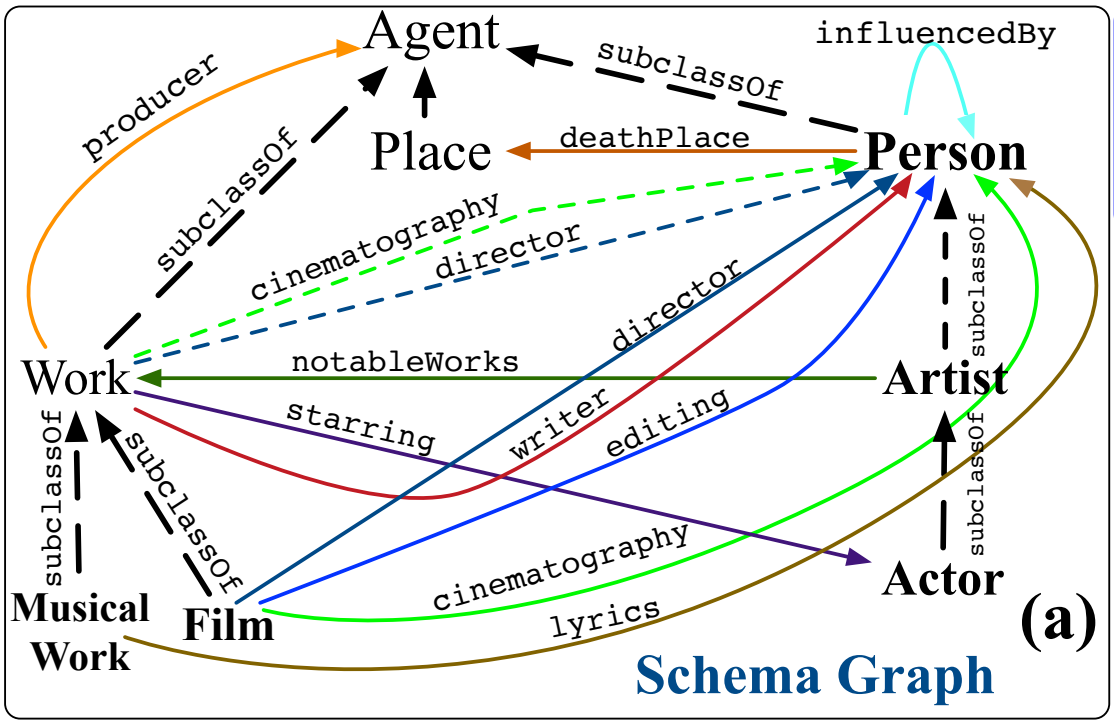
From facts to fact templates

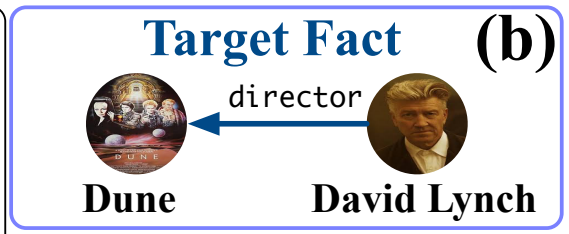


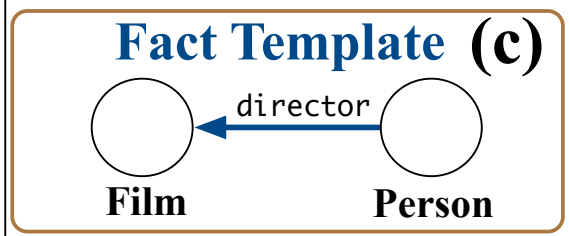


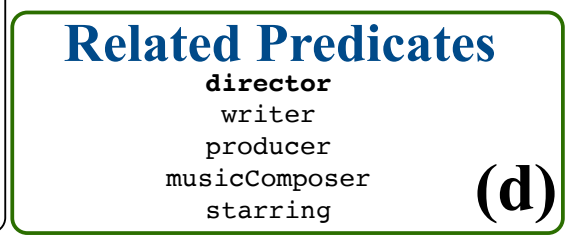


From facts to fact templates





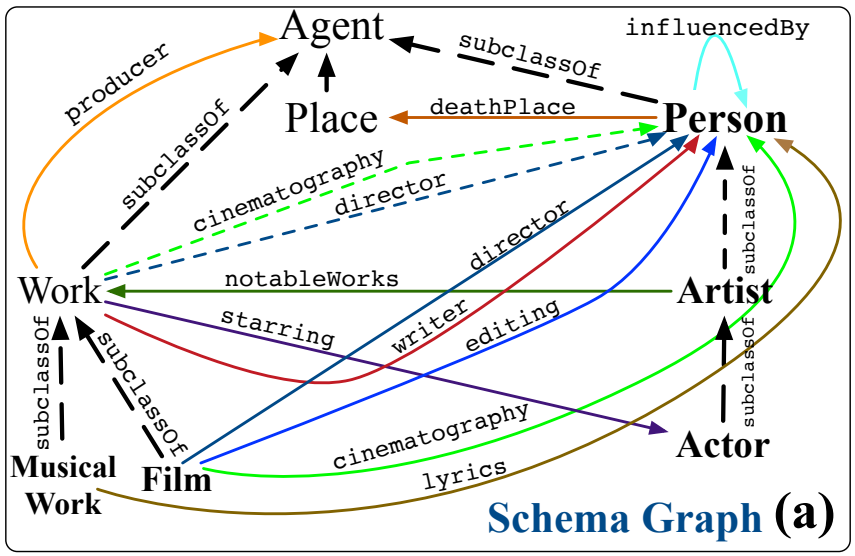


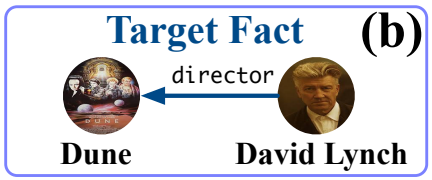


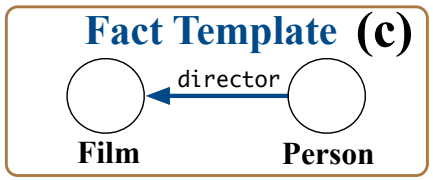
Predicate Relatedness

Relatedness allows to isolate the portion of the schema/data that is relevant to the target fact

Candidate Evidence Patterns



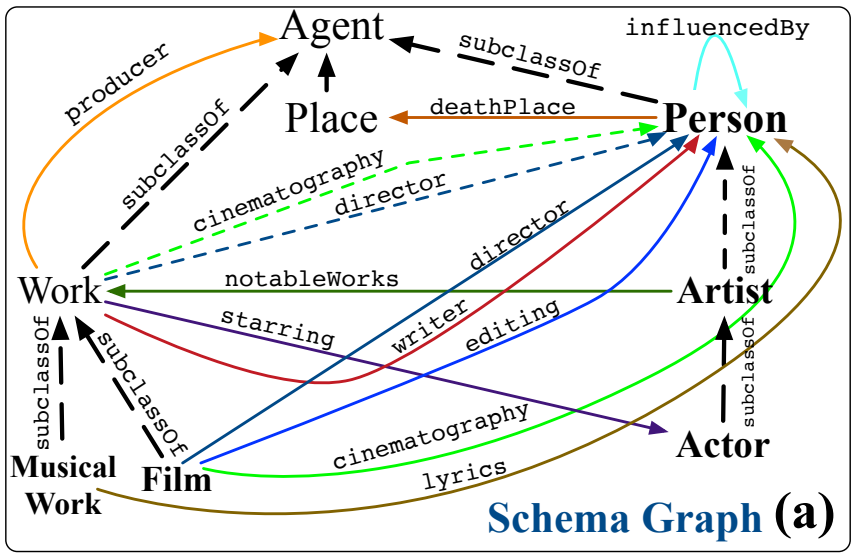


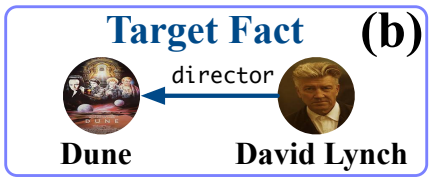


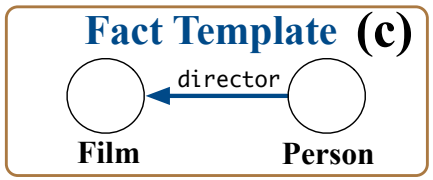
Related Predicates

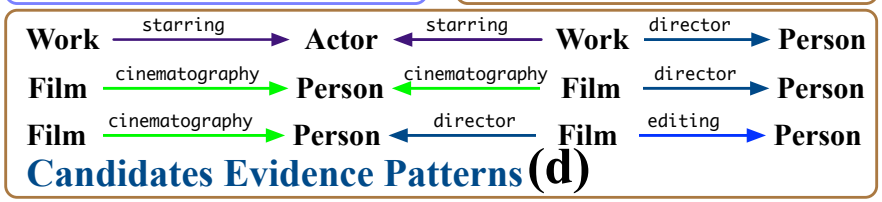
director
writer
producer
musicComposer
starring

Candidate Evidence Patterns





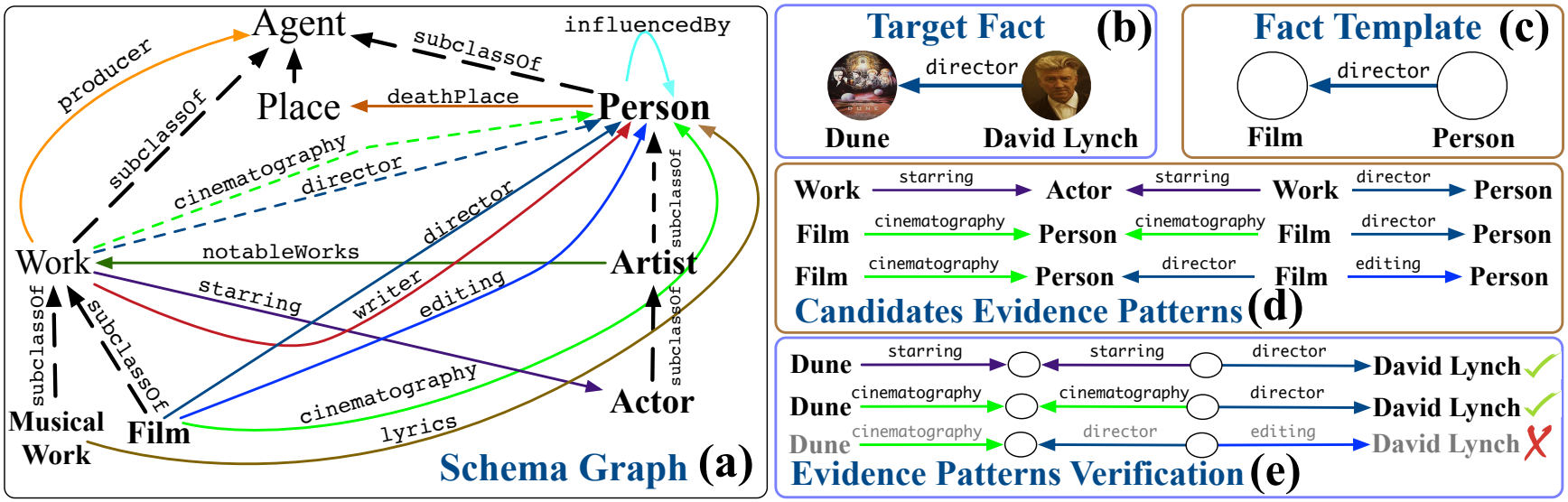




Candidate Evidence Patterns

- Schema-level paths of fixed length linking the entity types
- Generated via BFS with a priority queue based on relatedness

Candidate Evidence Patterns

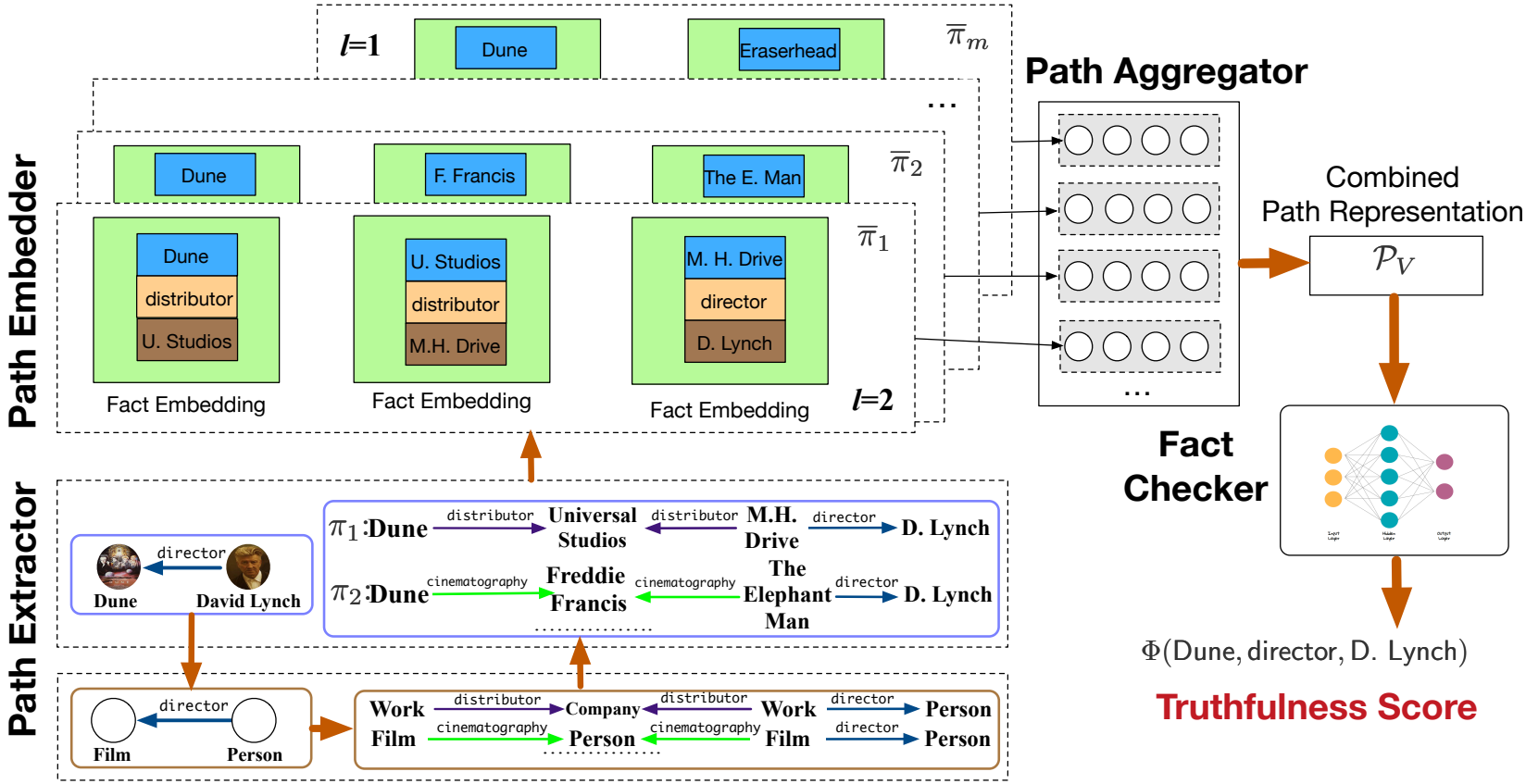


Use paths in the data as a source of evidence

Verify which patterns have a counterpart in the data

Bi-directional Depth First Search constrained to the pattern

Fact Embedding and Aggregator (FEA)



Use paths in the data as a source of evidence

Treat paths as sets of triples
Embed and aggregate paths according to different strategies

Path Embedding and Aggregation

Path Embedding:

- Given a triple t=(s, p, o), its embedding is t_E=EmbF(t).
- Aa path π=$\{t^1, t^2, \ldots t^l\}$ of length l including l facts is embedded as a sequence π_E=$[t_E^1, t_E^2, \ldots, t_E^l]$.

Path Embedding and Aggregation

Path Embedding:

- Given a triple t=(s, p, o), its embedding is t_E=EmbF(t).
- Aa path π=$\{t^1, t^2, \ldots t^l\}$ of length l including l facts is embedded as a sequence π_E=$[t_E^1, t_E^2, \ldots, t_E^l]$.

Path Aggregation:

1. **Average Pool:** concatenate the vector of the facts and perform an avg pooling: $\mathcal{P}_V^l = AvgPool([\oplus(\pi_i^l), \forall \pi_i^l \in \mathcal{P}^l])$

Path Embedding and Aggregation

Path Embedding:

- Given a triple t=(s, p, o), its embedding is t_E=EmbF(t).
- Aa path π=$\{t^1, t^2, \ldots t^l\}$ of length l including l facts is embedded as a sequence π_E=$[t_E^1, t_E^2, \ldots, t_E^l]$.

Path Aggregation:

1. **Average Pool:** concatenate the vector of the facts and perform an avg pooling: $\mathcal{P}_V^l = AvgPool([\oplus(\pi_i^l), \forall \pi_i^l \in \mathcal{P}^l])$
2. **Max Pool:** Use a dense neural network layer.

$$\mathcal{P}_V^l = MaxPool([\sigma(W_l \cdot \oplus(\pi_i^l) + b_l), \forall \pi_i^l \in \mathcal{P}^l])$$

where *MaxPool* is the one-dimension max pool operation.

Path Embedding and Aggregation

Path Embedding:

- Given a triple t=(s, p, o), its embedding is t_E=EmbF(t).
- Aa path π=$\{t^1, t^2, \ldots t^l\}$ of length l including l facts is embedded as a sequence π_E=$[t_E^1, t_E^2, \ldots, t_E^l]$.

Path Aggregation:

1. **Average Pool:** concatenate the vector of the facts and perform an avg pooling: $\mathcal{P}_V^l = AvgPool([\oplus(\pi_i^l), \forall \pi_i^l \in \mathcal{P}^l])$
2. **Max Pool:** Use a dense neural network layer.

$$\mathcal{P}_V^l = MaxPool([\sigma(W_l \cdot \oplus(\pi_i^l) + b_l), \forall \pi_i^l \in \mathcal{P}^l])$$

where *MaxPool* is the one-dimension max pool operation.

3. **LSTM Max Pool.** Treat a (vectorized) path as a sequence an employ an LSTM network to cater for sequential dependencies between facts in a path. After processing all paths a max pool operation produces the combined representation $\mathcal{P}_V$.

Learning task

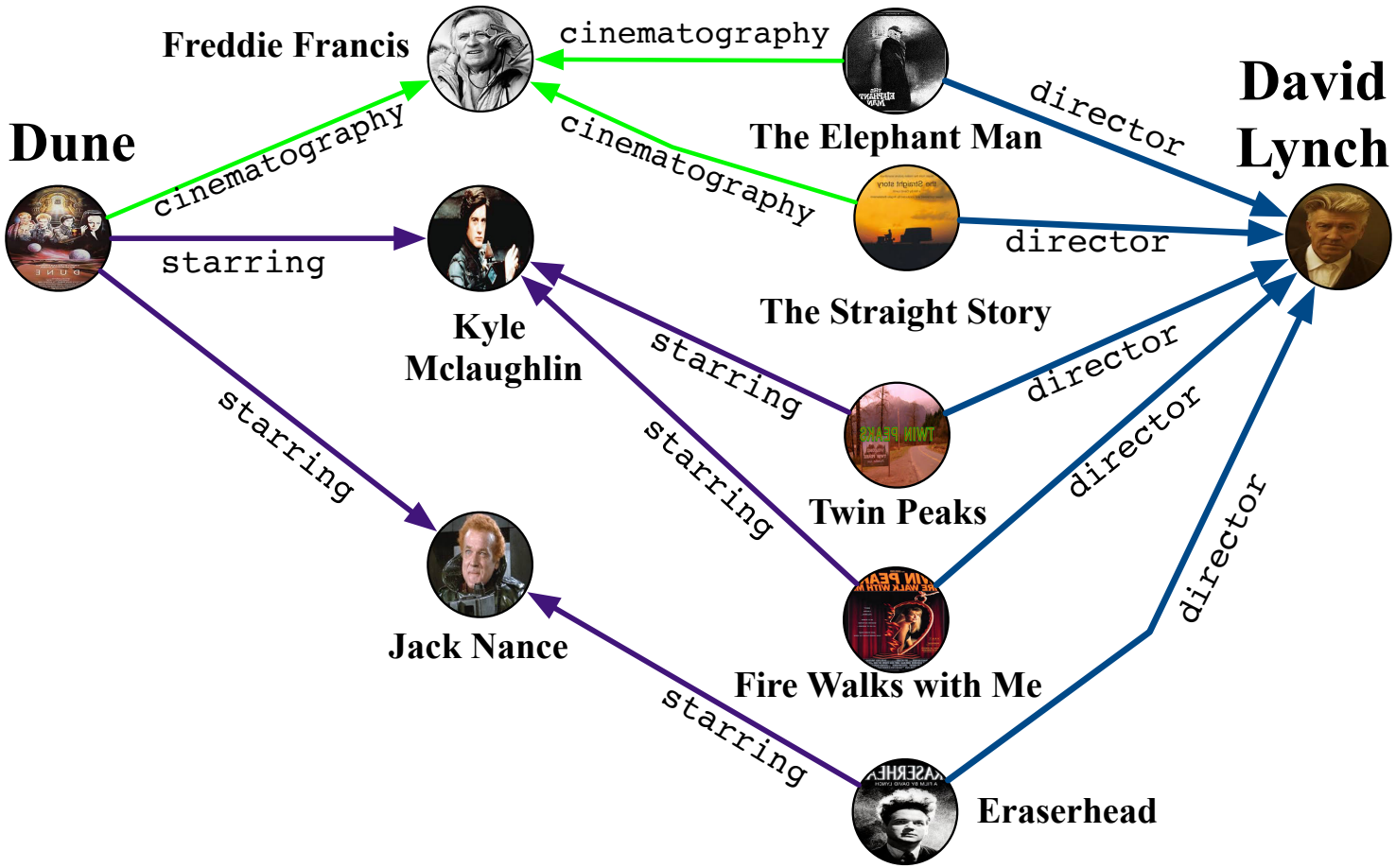
- Fact-checking problem as a binary classification problem
- True fact and a false fact are assigned 1 and 0 as target values, respectively.

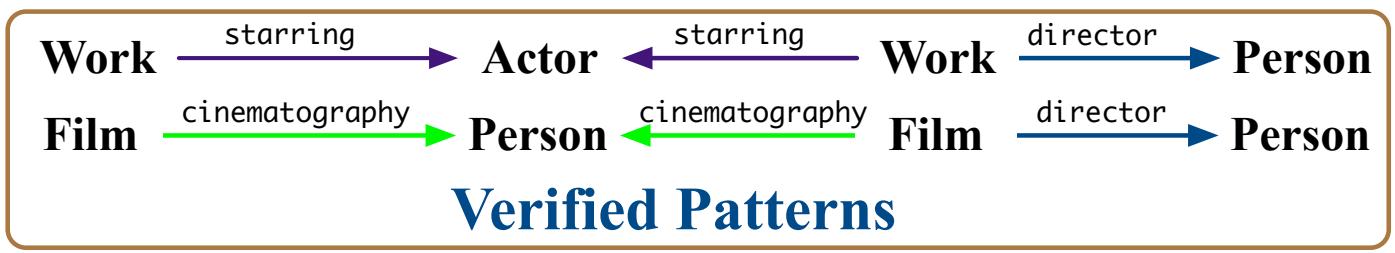
Optimize the negative log-likelihood objective function, which defined as follows:

$$\mathcal{L} = -\sum_{f^+ \in \mathcal{F}^+} log\ \hat{y}_{f^+} + \sum_{f^- \in \mathcal{F}^-} log(1 - \hat{y}_{f^-})$$

where $\mathcal{F}^+ = \{f^+ \mid y_{f^+} = 1\}$, $\mathcal{F}^- = \{f^+ \mid y_{f^-} = 0\}$ are the true (resp., false) facts.

Evidence Graph for the target fact





Experiments: setting

- **KG used**: DBpedia (24M triples, and 663 predicates)
- **Benchmarks**: 5 **real** (e.g., WSDM Cup Triple Scoring, Google Relation Extraction Corpora); 5 **synthetic** for which the ground truth is available; **BUCKLE** takes into account popularity, transparency, homogeneity, and functionality properties of the facts.
- **Competitors**: **CHEEP**, KStream (**KS**), KLinker (**KL**), PredPath (**PP**), Path Ranking Algorithm (**PRA**), **LEAP**,**TransE**
- **Metric**: AUC (A true fact obtains a score greater than a false fact)

Evaluation result

Approach	birthPlace (273/1092)	deathPlace (126/504)	almaMater (1546/6184)	nationality (50/200)	profession (110/440)
FEA-LSTMAggr	.93	.91	.81	.89	.99
FEA-MaxAggr	.90	.87	.80	.86	.97
FEA-AvgAggr	.91	.86	.81	.87	.99
CHEEP	.91	.87	.77	.85	.98
KStream	.82	.84	.75	.93	.93
KLinker	.91	.87	.78	.86	.93
PredPath	.86	.76	.83	.95	.92
PRA	.74	.75	.63	.83	.50
LEAP	.81	.74	.80	.91	.88
TransE	.54	.56	.66	.77	.82

Approach	author (93/558)	team (41/164)	director (78/4680)	keyPerson (201/1208)	spouse (16/256)
FEA-LSTMAggr	.93	.92	.99	.84	.98
FEA-MaxAggr	.90	.91	.97	.79	.94
FEA-AvgAggr	.92	.93	.99	.81	.98
CHEEP	.91	.91	.99	.82	.96
KStream	.92	.99	.83	.80	.86
KLinker	.96	.92	.88	.83	.91
PredPath	.99	.92	.84	.88	.87
PRA	.96	.91	.99	.87	.88
LEAP	.99	.89	.81	.82	.85
TransE	.80	.56	.82	.83	.79

Evaluation results: BUCKLE benchmark

Approach	Predicate	P	NP	R
FEA-LSTMAggr	**nearestCity**	.87	.67	.78
	foundedBy	.82	.67	.86
	manufacturer	.91	.88	.94
	employer	.70	.52	.71
FEA-MaxAggr	**nearestCity**	.79	.61	.72
	foundedBy	.77	.63	.79
	manufacturer	.88	.79	.86
	employer	.66	.47	.62
FEA-AvgAggr	**nearestCity**	.85	.64	.75
	foundedBy	.79	.63	.80
	manufacturer	.89	.88	.91
	employer	.68	.50	.67
CHEEP	**nearestCity**	.86	.61	.72
	foundedBy	.81	.62	.79
	manufacturer	.78	.86	.81
	employer	.67	.43	.64
PredPath	**nearestCity**	.84	.58	.69
	foundedBy	.80	.63	.81
	manufacturer	.55	.51	.53
	employer	.58	.38	.50
KLinker	**nearestCity**	.87	.66	.76
	foundedBy	.82	.67	.80
	manufacturer	.90	.85	.92
	employer	.69	.43	.66
LEAP	**nearestCity**	.41	.40	.41
	foundedBy	.69	.58	.71
	manufacturer	.68	.57	.64
	employer	.58	.42	.43
TransE	**nearestCity**	.49	.40	.43
	foundedBy	.75	.60	.75
	manufacturer	.72	.47	.70
	employer	.62	.46	.48

- P, NP, R: popular, non popular, and random entity pairs

Concluding Remarks

- **Summary**
  - Automatic Fact Checking approach using KGs
  - Learning model based on path embedding and aggregation
  - Good performance

Concluding Remarks

- **Summary**
  - Automatic Fact Checking approach using KGs
  - Learning model based on path embedding and aggregation
  - Good performance
- **Limitations**
  - Only facts of the form (s, p, o)
  - Coverage of the KG

Concluding Remarks

- **Summary**
  - Automatic Fact Checking approach using KGs
  - Learning model based on path embedding and aggregation
  - Good performance
- **Limitations**
  - Only facts of the form (s, p, o)
  - Coverage of the KG
- **Future Work**
  - Triples with temporal information
  - Learning approach based on Graph Neural Networks

Thankful to you I am