Dynamic Knowledge Graph Based Dialogue System

1 Project Overview

Real-world knowledge graphs often change over time. For instance, DBpedia \cite{1}, a multilingual knowledge graph extracted from content in Wikipedia, evolves as Wikipedia changes over time. There are a wide range of applications of dynamic knowledge graphs such as event forecasting \cite{2}, dialogue systems \cite{3, 4} and so on. In this project, we focus on how to build a dialogue system that can not only interact with others but also be aligned with a dynamic knowledge graph. In other words, the dialogue state should reflect the dynamics of a knowledge graph. For instance, if the relationship between two people changes from friends to competitors, the dialogue between them would also change due to this relationship change. Tuan et al. \cite{3} propose a novel task about grounding dialogue generation with dynamic knowledge graphs and collect a dataset for this task. However, they use an adaption method based on the transition matrix of the knowledge graph, which is undesirable and inflexible. It underuses the power of knowledge graphs.

In this project, we are more interested in how to design an embedding-based method to quickly adapt the model to the knowledge graph dynamics. Given the same input, we expect the dialogue system to generate different responses with different knowledge graphs. Besides, if the progress goes well, we would like to explore more about how to efficiently update the model as the changes in the knowledge graph would change so frequently to make training from scratch each time infeasible. It is a bonus for this project instead of a requirement. Finally, how to design automatic evaluation metrics to compare the baselines in this novel task is also an open question. Again, it is a bonus option.

2 Project Steps

1. Get acquainted with the related work \cite{2, 3, 4}.

2. Implement the zero-shot adaption method proposed in \cite{3}.

3. Improve the adaption to knowledge graph change via a model-based method.

4. Perform empirical evaluation to compare different baselines.

5. Explore more efficient updating tricks instead of training from scratch (bonus).

6. Explore more accurate evaluation metrics for this task (bonus).
3 Requirements

- Strong programming skills (proficiently with Pytorch or Tensorflow).
- Good knowledge of natural language processing, especially knowledge graphs.
- Acquaintance with related works including [2, 3].
- Good English especially in writing.

4 Postscript

- This project is for the master thesis. If you are looking for a semester project, we probably would adjust the workload to fit it into a semester project.
- Innovation and publication are highly encouraged.

References


