Temporal Graph Approach for Facebook's Babi Toy Question Answering Tasks

Project Title	Temporal Graph Approach for Facebook's Babi Toy Question Answering Tasks
Group Number	18
Name	ThammeGowda Narayanaswamy
USCID	2074-6694-39
Email	tnarayan@usc.edu
Collaborators	• Ravi Raju Krishna • Aditya Ramachandra Desai
Code Repository	https://github.com/raviraju/NLP_QA_Project

1. PROJECT OVERVIEW

Natural language question answering (QA) is an interface that allows humans interact with machines in a natural way to retrieve necessary information or perform an action. Any application that has natural language input and an output can be transformed into a kind of QA system. In the recent days, the question answering interfaces are being embedded into consumer electronics such as smartphones (Google Now in Android, Siri in Apple iOS, Cortana in Microsoft Windows 10), televisions and its remotes (Amazon Fire TV), household music systems (Amazon Alexa, Google Home), Cars (Google, Tesla, Apple) and this list is growing. Considering the above facts, we believed it is a good time for students like ourselves to have hands on experience with QA systems on behalf of the mini project for the Applied NLP coursework.

However, developing a QA system is a challenging problem with high incentives. It is is a standard problem in Artificial Intelligence (AI). It is believed that building a sane general purpose system with question answering capabilities like human mind is as hard as solving the general AI problem itself. Considering these facts, we started to frame a simpler problem statement for a team of three grad students to achieve in a few weeks of part time work. We reduced the complexities of general question answering systems by narrowing and focussing to a simulated world. We chose a synthetic dataset from Facebook AI Research named bAbI toy tasks [1] by Weston et al. The dataset had 20 different question types, we choose to solve three question

types due to the shorter timeframe of the mini project. These questions had stories in a simulated worlds based on actions performed by the actors. The dataset is designed to test the natural language understanding and reasoning capabilities.
Here is an example:

```
1 Mary moved to the bathroom.
2 Sandra journeyed to the bedroom.
3 Mary got the football there.
4 John went to the kitchen.
5 Mary went back to the kitchen.
6 Mary went back to the garden.
7 Where is the football?    garden        3 6
```

Fig 1. A sample two facts story

During the course of this project, we reviewed several techniques for constructing question answering systems and understood the challenges involved in them. At the high level, the NLP and AI tasks involved are Natural Language Understanding (NLU), Knowledge Representation(KR), Retrieval and Reasoning (RR), and Question Intent classification, Answer generation etc. We explored some of the state of the art machine learning techniques for constructing QA systems. We came up with a simpler and easily explainable method using temporal knowledge representation and inference based on graph traversal. We also constructed interactive graphs to visualise the internal states of knowledge base. Finally, we compared the results with the baseline provided in the literature (covered at the 'Evaluation' section).

2. MY PRIMARY RESPONSIBILITY

During the course of this project, I explored and reviewed some of the state-of-the-art techniques for constructing question answering systems. Weston et al [1] provide a set of prequisite toy tasks for AI-complete question answering. Taking advantage of my past expertise in machine learning domain, I explained and offered initial guidance to the other team members about the key techniques in statistical learning methods such as objective functions, convex optimisations and feature space transformations. I also explored the work of Weston et al [2] related to generic architecture for question answering using memory based networks and an example implementation using recurrent neural networks(RNN) such as Long Short-Term Memory (LSTM) networks. The following section provides an overview of my findings:

2.a Memory Networks for Question Answering

In the paper [2], the authors describes the following 4 components for building a natural language question answering system:

1) I - Input Feature Map Maps natural text to feature space
2) G - Generalization: Updates the status of the story incrementally

3) O - Output Feature Map: Retrieves candidate answers to given question
4) R - Response: Produces the final response in the expected format

In the next step, I tested an implementation based on LSTMs. However I realized that getting correct answers to questions is not enough for a AI completeness. The missing part was the ability to explain the steps taken by the QA system to induce/deduce/conclude an answer is also necessary requirement for explaining to the users as well as debugging the faulty answers. Since the reasoning in neural networks is mereley the low level numerical calculations done in inexplicable feature spaces, we considered alternative approaches to solving the same task. I proposed an approach based on the temporal graph knowledge representation (originally called timeline representation) which is a lot simpler to understand and to provide proofs for the produced answers. Even though we moved away from neural network approach, architecture is influenced by the key components described above. The following section describes this approach in detail.

2.b Temporal Graph Knowledge Representation for Question Answering

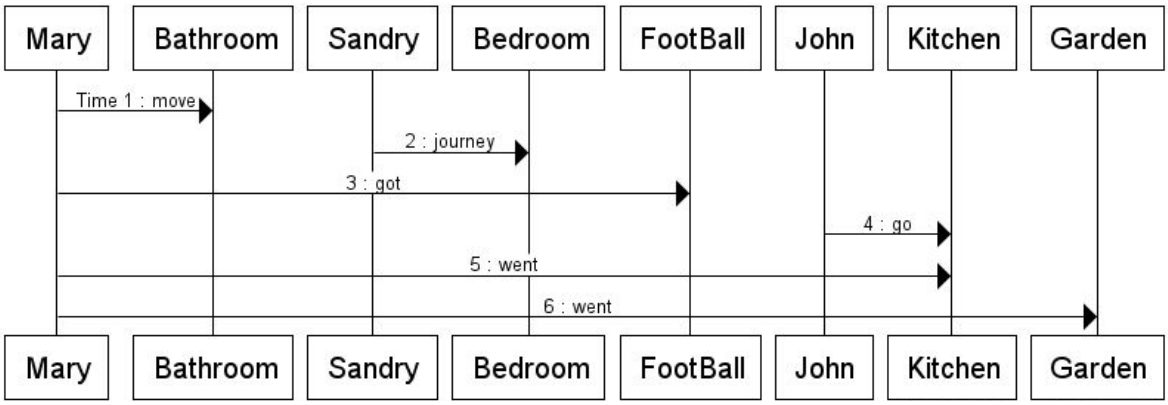
A key feature of our project is temporal graph knowledge representation. Having a powerful knowledge representation using temporal graph data structure to remember the facts, we are able to capture the rich meaning of the statements in the story. It helped to capture every information required to answer the challenge questions perfectly. Also, by taking advantage of the graph traversal techniques, we are able to answer all questions correctly.
The steps to construct the temporal graph are as follows:

I. ***Parsing the story lines (Collaborator: Aditya Ramachandra Desai)***
All the english statements from the dataset were parsed and normalized into (TIME, SUBJECT, VERB, OBJECT) tuples. TIME is the timestamp of the statement which is an autoincrementing integer for the chosen synthetic dataset. The SUBJECT is chosen by tagging the part of speech for each token and picking the first noun(NN* POS) from the statement. Similarly, OBJECT was chosen from the second noun in the sentence. The VERB is chosen form of the base form of the verb by lemmatizing the verb tokens.

II. ***Updating the graph:***
Given the (TIME, SUBJECT, VERB, OBJECT) tuples, we added SUBJECT and OBJECT as nodes in the graph if they do not already exists. Then VERB is added as an edge between SUBJECT and OBJECT at the index TIME. Note that these are the update statements made to the simulated knowledge base and inspired by how we believe humans update the knowledge in mind by cognition. Below is the temporal graph constructed for an example story

Fig 2. Sample Temporal Graph for the story:
1 Mary moved to the bathroom.
2 Sandra journeyed to the bedroom.
3 Mary got the football there.
4 John went to the kitchen.
5 Mary went back to the kitchen.
6 Mary went back to the garden.

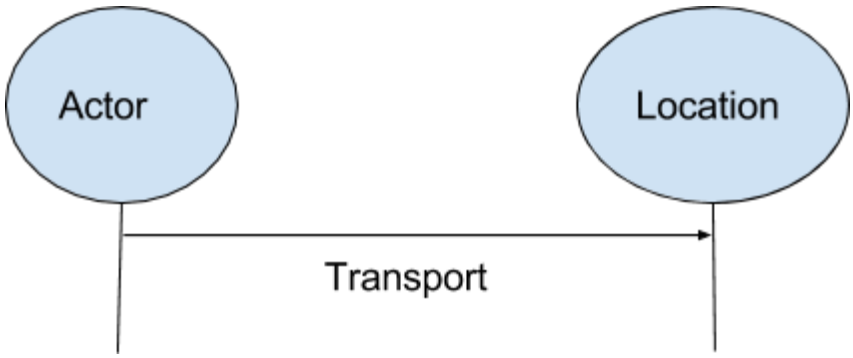
7 Where is the football? garden 3 6



III. ***Traversing the Graphs:***

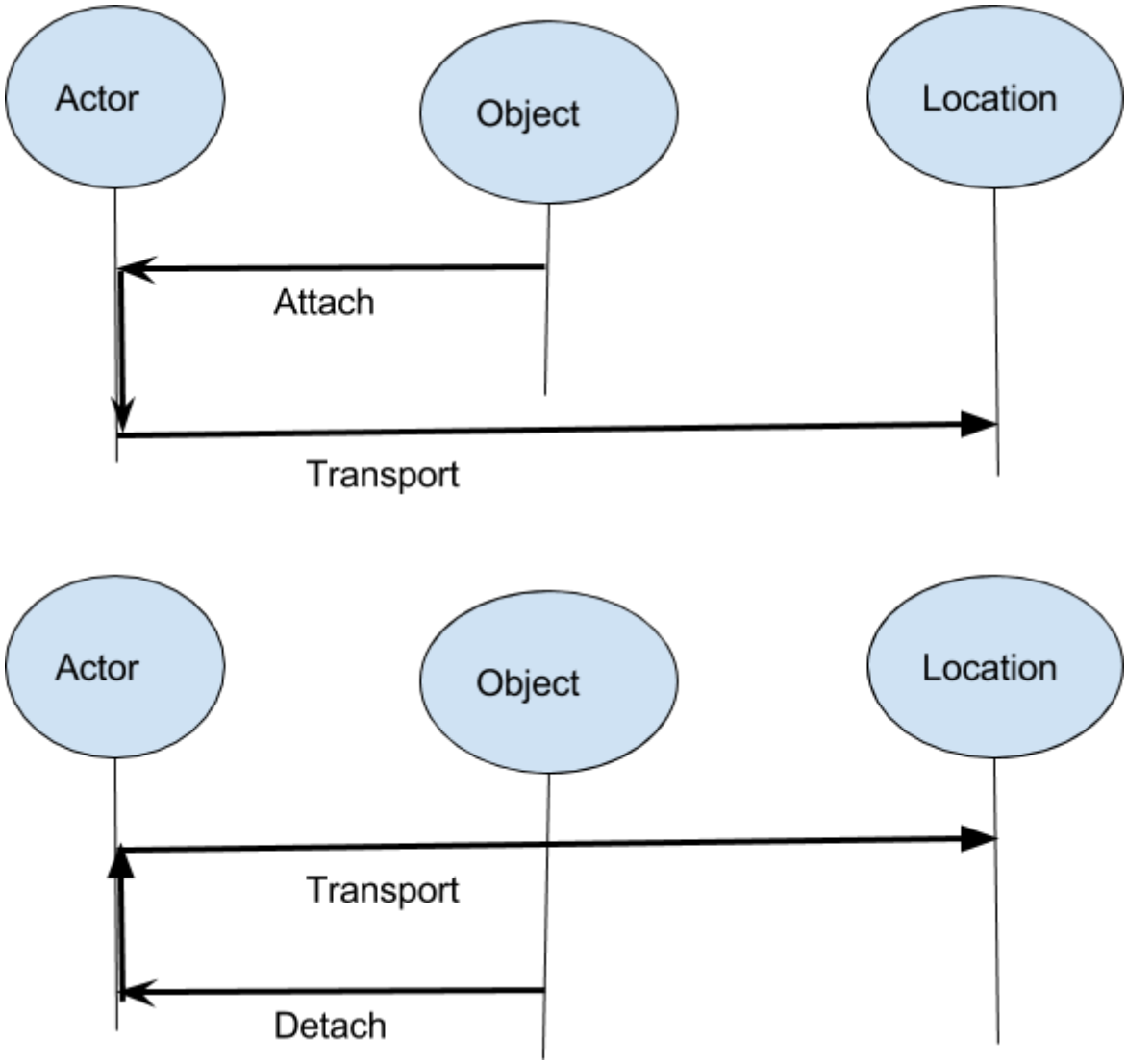
The traversal for single supporting fact questions are different than that for two supporting fact questions.

For single supporting fact questions, example: “Where is the actor?”, the traversal includes the actor node, an action edge of type ‘transport’ and then a ‘location’ node. Since the dataset had no ambiguities, the transport node always connected an actor node to location node. We took advantage of this fact and thus did not have to build a classifier for the nodes. However a classifier was required to build for classifying the edge types which are described in the later section. The final answer for a one supporting fact question is the most recent location node (decided based on time) at the time of question.



For the two supporting questions, the traversal includes two types:

i) If the most recent edge on the object node is of type ‘Attach’, the traversal walks forwards in the time dimension on the connecting actor. ii) If the most recent edge in the Object node is a ‘Detach’ action, then the traversal walks backwards in time on the connecting actor node.



Traversal for the Yes-No type is similar to the above except an additional preprocessing and a post-processing step. In the preprocessing step, the Yes-No question was converted to simple 'Where is' question on the subject. In the post processing the answer transformed to boolean Yes/No by checking against the object in the question. For example, consider a question "Is Mary in the office?". The preprocessing step converts it into "Where is Mary?". The post-processing step transforms the answer to "Yes" if the answer is same as "office" or returns "No" otherwise.

2.c. Evaluation:

The dataset had seperate labelled examples for training and testing set. Each of these had 1000 Question answer pairs based on the stories in between. I wrote an evaluator script to compute the statistics about the ratio of correct answers to the total number of questions on the test dataset. Our evaluation results compared with the baseline system from [3] are as follows:

	FB LSTM	*Stephen Merity's RNN*	*Our score*
1 Supporting Fact Type	50%	52.1%	100%
2 Supporting Fact Type	20%	37.0%	100%
Yes-No Type	48%	50.7%	100%

As indicated by the statistics in above table, RNNs have a baseline of about 50% for the Single supporting fact and Yes-No types questions. The two supporting type questions have very low baseline since they are relatively complex to infer the answer. On the otherside, our approach scored a perfect 100% correctness. The credit for such a higher score goes to the rich knowledge representation using temporal graphs, inference using deterministic graph traversal procedures and the lack of ambiguities in the chosen synthetic dataset.

3. OTHER PROJECT WORK

My other contributions to the project are:

- **Integration and Code cleanup:**
  I coordinated with the other members of the team to have the code cleaned and nicely integrated to have an end to end executable system.
- **Interactive temporal graphs:** Construction of interactive graphs for demonstrating and understanding the internal state modification of Temporal Graphs. I also constructed an ipython notebook to demonstrate the graphs [6].
- **Action Classifier** (Collaborator: Ravi Raju Krishna): Built a classifier for classifying action verbs into three different classes, namely - *attach, detach and transport.* Action classifier was a key decision maker in graph traversal. Actions are based on the verbs in the story lines.
  For example :
  - Mary moved to the office → transport
  - Mary picked up the apple → attach
  - Mary dropped the apple → detach

4. ONLINE RESOURCES:

- **Babi toy QA tasks Dataset**
  Purpose : the Question answer dataset for the project
  URL : https://github.com/facebook/bAbI-tasks
- **Stanford CoreNLP**
  Purpose : Parsing the stories, part of speech tagging, lemmatization
  URL : http://stanfordnlp.github.io/CoreNLP/
- **Networkx**
  Purpose : Graph library for constructing temporal graph
  URL : https://networkx.github.io/
- **Matplotlib**
  Purpose : Visualization
  URL : http://matplotlib.org/

- **Keras Baseline Implementation and performance**
  Pupose : Baseline performance comparison
  URL : http://smerity.com/articles/2015/keras_qa.html

5. REFERENCES

[1] Weston, J., Bordes, A., Chopra, S., Rush, A. M., van Merriënboer, B., Joulin, A., & Mikolov, T. (2015). Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*.

[2] Weston, J., Chopra, S., & Bordes, A. (2014). Memory networks. *arXiv preprint arXiv:1410.3916*.

[3] Merity, S., *Question answering on the Facebook bAbi dataset using recurrent neural networks* http://smerity.com/articles/2015/keras_qa.html

[4] Karpathy, A., *The Unreasonable Effectiveness of Recurrent Neural Networks* http://karpathy.github.io/2015/05/21/rnn-effectiveness/

[5] Toutanova, K., Klein, D., Manning, C. D., & Singer, Y. (2003, May). Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1* (pp. 173-180). Association for Computational Linguistics.

[6] Demo Notebook
https://github.com/raviraju/NLP_QA_Project/blob/master/NERs_and_Graph/notes/Demo.ipynb