StepGame: A New Benchmark for
Robust Multi-Hop Spatial Reasoning in Texts

Zhengxiang Shi[1], Qiang Zhang[2], Aldo Lipani[1]

[1]University College London
[2]Zhejiang University
zhengxiang.shi.19@ucl.ac.uk, qiang.zhang.cs@zju.edu.cn, aldo.lipani@ucl.ac.uk

Abstract

Inferring spatial relations in natural language is a crucial ability an intelligent system should possess. The bAbI dataset tries to capture tasks relevant to this domain (task 17 and 19). However, these tasks have several limitations. Most importantly, they are limited to fixed expressions, they are limited in the number of reasoning steps required to solve them, and they fail to test the robustness of models to input that contains irrelevant or redundant information. In this paper, we present a new Question-Answering dataset called StepGame for *robust multi-hop spatial reasoning in texts*. Our experiments demonstrate that state-of-the-art models on the bAbI dataset struggle on the StepGame dataset. Moreover, we propose a Tensor-Product based Memory-Augmented Neural Network (TP-MANN) specialized for spatial reasoning tasks. Experimental results on both datasets show that our model outperforms all the baselines with superior generalization and robustness performance.

1 Introduction

Neural networks have been successful in a wide array of perceptual tasks, but it is often stated that they are incapable of solving tasks that require higher-level reasoning (Ding et al. 2020). Since spatial reasoning is ubiquitous in many scenarios such as autonomous navigation (Vogel and Jurafsky 2010), situated dialog (Kruijff et al. 2007), and robotic manipulation (Yang, Lan, and Narasimhan 2020; Landsiedel et al. 2017), grounding spatial references in texts is essential for effective human-machine communication through natural language. Navigation tasks require agents to reason about their relative position to objects and how these relations change as they move through the environment (Chen et al. 2019). If we want to develop conversational systems able to assist users in solving tasks where spatial references are involved, we need to make them able to understand and reason about spatial references in natural language. Such ability can help conversational systems to successfully follow instructions and understand spatial descriptions. However, despite its tremendous applicability, reasoning over spatial relations remains a challenging task for existing conversational systems.

Earlier works in spatial reasoning focused on spatial instruction understanding in a synthetic environment (Bisk et al. 2018; Tan and Bansal 2018; Janner, Narasimhan, and Barzilay 2018) or in a simulated world with spatial information annotation in texts (Pustejovsky et al. 2015), spatial relation extractions across entities (Petruck and Ellsworth 2018) and visual observations (Anderson et al. 2018; Chen et al. 2019). However, few of the existing datasets are designed to evaluate models' inference over spatial information in texts. A spatial relational inference task often requires an conversational system to infer the spatial relation between two items given a description of a scene. For example, imagine a user asking to a conversational system to recognize the location of an entity based on the description of other entities in a scene. To do so, the conversational system needs to be able to reason about the location of the various entities in the scene using only textual information.

BAbI (Weston et al. 2016) is the most relevant dataset for this task. It contains 20 synthetic question answering (QA) tasks to test a variety of reasoning abilities in texts, like deduction, co-reference, and counting. In particular, the *positional reasoning* task (no. 17) and the *path finding* task (no. 19) are designed to evaluate models' spatial reasoning ability. These two tasks are arguably the most challenging ones van Aken et al. (2019). The state-of-the-art model on the bAbI (Le, Tran, and Venkatesh 2020) dataset almost perfectly solve these two spatial reasoning tasks. However, in this paper, we demonstrate that such good performance is attributable to issues with the bAbI dataset rather than the model inference ability.

We find four major issues with bAbI's tasks 17 and 19: (1) There is a data leakage between the train and test sets; that is, most of the test set samples appear in the training set. Hence, the evaluation results on the test set cannot truly reflect models' reasoning ability; (2) Named entities are fixed and only four relations are considered. Each text sample always contains the same four named entities in the training, validation, and test sets. This further biases the learning models towards these four entities. When named entities in the test set are replaced by unseen entities or the number of such entities increases, the model performance decreases dramatically (Chen et al. 2020a). Also, relations such as top-left, top-right, lower-left, lower-right are not taken into consideration; (3) Learning models are required to reason only over

one or two sentences in the text descriptions, making such tasks relatively simple. Palm, Paquet, and Winther (2018) pointed out that multi-hop reasoning is not necessary for the bAbI dataset since models only need a single step to solve all the tasks, and; (4) It is a synthetic dataset with a limited diversity of spatial relation descriptions. It thus cannot truly reveal the models' ability in understanding textual space descriptions.

In this paper, we propose a new dataset called *StepGame* to tackle the above-mentioned issues and a novel *Tensor Product-based Memory-Augmented Neural Network* architecture (TP-MANN) for multi-hop spatial reasoning in texts.

The StepGame dataset is based on crowdsourced descriptions of 8 potential spatial relations between 2 entities. These descriptions are then used as templates when generating the dataset. To increase the diversity of these templates, crowdworkers were asked to diversify their expressions. This was done in order to ensure that the crowdsourced templates cover most of the natural ways relations between two entities can be described in text. The StepGame dataset is characterized by a combinatorial growth in the number of possible description of scenes, named *stories*, as the number of described relations between two entities increases. This combinatorial growth reduces the chances to leak stories from the training to the validation and test sets. Moreover, we use a large number of named entities and require multi-hop reasoning to answer *questions* about two entities mentioned in the stories. Experimental results show that existing models (1) fail to achieve a performance on the StepGame dataset similar to that achieved on the bAbI dataset, and (2) suffer from a large performance drop as the number of required reasoning steps increases.

The TP-MANN architecture is based on tensor product representations (Smolensky 1990) that are used in a recurrent memory module to store, update or delete the relation information among entities inferred from stories. This recurrent architecture provides three key benefits: (1) it enables the model to make inferences based on the stored memory; (2) it allows multi-hop reasoning and it is robust to noise, and; (3) the number of parameters remains unchanged as the number of recurrent layers in the memory module increases. Experimental results on the StepGame dataset show that our model achieves state-of-the-art performance with a substantial improvement, and demonstrates a better generalization ability to more complex stories. Finally, we also conduct some analysis of our recurrent structure and demonstrate its importance for multi-hop reasoning.

2 Related Work and Background

2.1 Related Work

Reasoning Datasets. The role of language in spatial reasoning has been investigated since the 1980s (Pustejovsky 1989; Logan and Sadler 1996; Gershman and Tenenbaum 2015; Tversky 2019), and reasoning about spatial relations has been studied in several contexts such as, 2D and 3D navigation (Bisk et al. 2018; Tan and Bansal 2018; Janner, Narasimhan, and Barzilay 2018; Yang, Lan, and Narasimhan 2020), and robotic manipulation (Landsiedel et al. 2017).

However, few of the datasets used in these works are used to evaluate systems' spatial reasoning ability in texts.

The bAbI (Weston et al. 2016) dataset consists of several QA tasks. Solving these tasks require logical reasoning steps and cannot be solved by simply word matching. Of particular interest to this paper are tasks 17 and 19. Task 17 is about positional reasoning while task 19 is about path finding. These two tasks can be used to evaluate the spatial inference ability of learning models. However, the bAbI dataset has several issues as mentioned above: the data leakage, the fixed named entities and expressions, and the lack of a need to perform multi-hop reasoning.

Multi-Hop QA Datasets. The multi-hop QA tasks require reasoning over multiple pieces of evidence and focus on leveraging the connections between entities to infer a requested property of a set of them. Commonly-used multi-hop QA datasets are HotpotQA (Yang et al. 2018), ComplexWebQuestions (Talmor and Berant 2018), and QAngaroo (Welbl, Stenetorp, and Riedel 2018). The proposed StepGame dataset is different from these datasets. The StepGame dataset focuses on spatial reasoning, which requires machine learning models to infer the spatial relations among the described entities. Moreover, multi-hop QA datasets usually require no more than two reasoning steps, while the StepGame dataset can require as many as 10 reasoning steps.

Reasoning Models. There are three types of reasoning models: memory-augmented neural networks, graph neural networks, and transformer-based networks. Works of the first type augment neural networks with external memory, such as End to End Memory Networks (Sukhbaatar et al. 2015), Differential Neural Computer (Graves et al. 2016), and Gated End-to-End Memory Networks (Liu and Perez 2017). These models have shown remarkable abilities in tackling difficult computational and reasoning tasks. Works of the second type use graph structure to incorporate a stronger relational inductive bias (Battaglia et al. 2018). Santoro et al. (2017) introduced Relational Networks (RN) and demonstrated strong relational reasoning capabilities with a shallow architecture by modelling binary relations between entity pairs. Palm, Paquet, and Winther (2018) proposed a graph representation of objects and models multi-hop relational reasoning using a message passing mechanism. Works of the third type use transformers. Although transformers have been proven successful in many NLP tasks, they still struggle with reasoning tasks. van Aken et al. (2019) analyzed the performance of BERT (Devlin et al. 2019) on bAbI's tasks and demonstrated that most of BERT's errors come from task 17 and 19 which require spatial reasoning. Meanwhile, Dehghani et al. (2019) demonstrated that standard transformers cannot perform as well as memory-augmented networks on the bAbI dataset. Moreover, it is important to note that most of the errors of their proposed Universal Transformer come also from task 17 and task 19 of the bAbI dataset, which matches our observations on other transformer-based models. Therefore, spatial reasoning tasks are arguably the most challenging tasks in the bAbI dataset.

Tensor Product Representation. The Tensor Product Representation (TPR) (Smolensky 1990) is a technique for encoding symbolic structural information and modelling symbolic reasoning in vector spaces by learning to deconstruct natural language statements into combinatorial representations (Chen et al. 2020b). TPR has been used for tasks that require deductive reasoning abilities and it is able to represent entire problem statements to solve math questions in natural language (Chen et al. 2020b) and generate natural language captions from images (Huang et al. 2018).

Schlag and Schmidhuber (2018) proposed a gradient-based RNN with third-order TPR (TPR-RNN), which creates a vector space embedding of complex symbolic structures by tensor products and stores these learned representations into a third-order TPR-like memory. Self-Attentive Associative Memory (STM) (Le, Tran, and Venkatesh 2020) utilizes a second-order item memory and a third-order TPR-like relational memory to simulate the hippocampus, achieving state-of-the-art performance on the bAbI dataset. Despite a gain in the performance on bAbI compared to TPR-RNN, STM uses a much larger number of trainable parameters and takes a longer time to converge in practice. Our model size is four times smaller than STM.

Recently, Schlag, Munkhdalai, and Schmidhuber (2021) compared a concatenated memory $M \in \mathbb{R}^{2d \times d}$ with a 3-order memory $M \in \mathbb{R}^{d^2 \times d}$, and experimental results indicate a drop in performance when a concatenated memory is used. However, neither STM nor TPR-RNN processes information at the paragraph level and allows later modifications after the first information is stored, as done in our model. Both STM and TPR-RNN use an RNN-like architecture where each sentence in a paragraph is stored recurrently. This may result in a long-term dependency problem (Vaswani et al. 2017) where necessary information would not interact with each other. To solve this issue, an explicit mechanism to update relational information between entities at the end of each story is introduced in our model.

2.2 Background

The Tensor Product Representation (TPR) is a method to create a vector space embedding of complex symbolic structures by tensor product. Such representation can be constructed as follows:

$$M = \sum_i f_i \otimes r_i = \sum_i f_i r_i^\top = \sum_i (\boldsymbol{f} \otimes \boldsymbol{r})_{ii}, \quad (1)$$

where M is the TPR, $\boldsymbol{f} = (f_1, \ldots, f_n)$ is a set of n filler vectors and $\boldsymbol{r} = (r_1, \ldots, r_n)$ is a set of n role vectors. For each role-filler vector pair, which can be considered as an entity-relation pair, we *bind* (or store) them into M by performing their outer product. Then, given an unbinding role vector u_i, associated to the filler vector f_i, f_i can be recovered by performing:

$$M u_i = \left[\sum_i f_i \otimes r_i \right] u_i = \sum_i \alpha_{ij} f_i \propto f_i \quad (2)$$

where $\alpha_{ij} \neq 0$ if and only if $i = j$. It can be proven that the recovering is perfect if the role vectors are orthogonal to each other. In our model, TPR-like *binding* and *unbinding* methods are used to store and retrieve information from and to the TPR M, which we will call memory.

3 The StepGame Dataset

To design a benchmark dataset that explicitly tests models' spatial reasoning ability and tackle the above mentioned problems, we build a new dataset named StepGame inspired by the spatial reasoning tasks in the bAbI dataset (Weston et al. 2016). The StepGame is a contextual QA dataset, where the system is required to interpret a story about several entities expressed in natural language and answer a question about the relative position of two of those entities. Although this reasoning task is trivial for humans, to equip current NLU models with such a spatial-ability remains still a challenge. Also, to increase the complexity of this dataset we model several form of *distracting noises*. Such noises aim to make the task more difficult and force machine learning models that are trained on this dataset to be more robust in their inference process.

3.1 Template Collection

The aim of this crowdsourcing task is to find out all possible ways we can describe the positional relationship between two entities. The crowdworkers from Amazon Mechanical Turk were provided with an image visually describing the spatial relations of two entities and a request to describe these entities' relation. This crowdsourcing task was performed in multiple runs. In the first run, we provided crowdworkers with an image and two entities (e.g., A and B) and they were asked to describe their positional relation. From the data collected in this round, we then manually removed bad answers, and showed the remaining good ones as positive examples to crowdworkers in the next run. However, crowdworkers were instructed to avoid repeating them as an answer to our request. We repeated this process until no new templates could be collected. In total, after performing a manual generalization where templates discovered for a relation were translated to the other relations, we collected 23 templates for left and right relations, 27 templates for top and down relations, and 26 templates for top-left, top-right, down-left, and down-right relations.

3.2 Data Generation

The task defined by the StepGame dataset is composed of several story-question pairs written in natural language. In its basic form, the story describes a set of k spatial relations among $k + 1$ entities, and it is structured as a list of k sentences each talking about 2 entities. The relations are k and the entities $k + 1$ because they define a chain-like shape. The question requests the relative position of two entities among the $k + 1$ ones mentioned in the story. To each story-question pair an answer is associated. This answer can take 9 possible values: *top-left*, *top-right*, *top*, *left*, *overlap*, *right*, *down-left*, *down-right*, and *down*, each representing a relative position. The number of edges between the two entities in the question ($\leq k$) determines the number of hops a model has to perform in order to get to the correct answer.
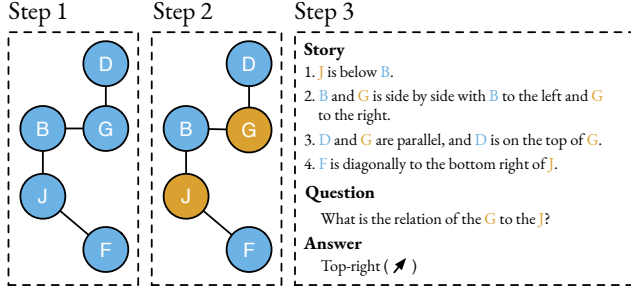
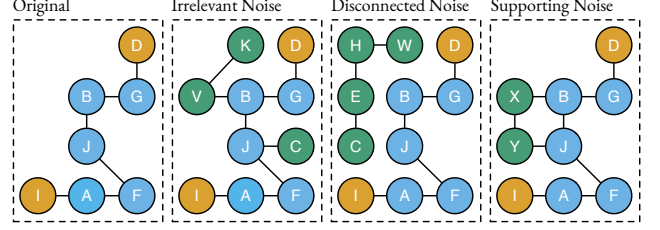Figure 1: An example of the generation of a StepGame sample with $k = 4$.

Figure 2: On the left-hand side we have the original chain. Orange entities are those targeted by the question. Beside, we show the same chain with the addition of noise. In green we represent irrelevant, disconnected and supporting entities.

To generate a story, we follow three steps, as depicted in Figure 1. Given a value k and a set of entities $\mathcal{E}$:

Step 1. We generate a sequence of entities by sampling a set of $k + 1$ unique entities from $\mathcal{E}$. Then, for each pair of entities in the sequence, k spatial relations are sampled. These spatial relations can take any of the 8 possible values: top, down, left, right, top-left, top-right, down-left, and down-right. Because the sampling is unconstrained, entities can overlap with each other. This step results in a sequence of linked entities that from now on we will call a chain.

Step 2. Two of the chain's entities are then selected at random to be used in the question.

Step 3. From the chain generated in Step 1, we translate the k relations into k sentence descriptions in natural language. Each description is based on a randomly sampled crowdsourced template. We then shuffle these k sentences to avoid potential distributional biases. These shuffled k sentence descriptions is a called a story. From the entities selected in Step 2, we then generate a question also in natural language. Finally, using the chain and the selected entities, we infer the answer to each story-question pair.

Given this generation process we can quickly calculate the complexity of the task before using the templates. This is possible because entities can overlap. Given k relations, $k + 1$ entities sampled from $\mathcal{E}$ in any order (●), 8 possible relations between pairs of entities with 2 ways of describing them (●), e.g., A is on the left of B or B is on the right of A, a random order of the k sentences in the story (●), and a question about 2 entities with 2 ways of describing it (●), the number of examples that we can generate is equal to:

$$(k + 1)! \binom{|\mathcal{E}|}{k + 1} \cdot 16^k \cdot \frac{k!}{2} \cdot 2 \binom{k + 1}{2}. \qquad (3)$$

The complexity of the dataset grows exponentially with k. The StepGame dataset uses $|\mathcal{E}| = 26$. For $k = 1$ we have 5,200 possible samples, for $k = 2$ we have more than 23 million samples, and so on. The sample complexity of the problem guarantees that when generating the dataset the probability of leaking samples from the training set to the test set diminishes with the increase of k. Please note that these calculations do not include templates. If we were to considering also the templates, the number of variations of the StepGame would be even larger.

3.3 Distracting Noise

To make the StepGame more challenging we also include noisy examples in the test set. We assume that when models trained on the non-noisy dataset make mistakes on the noisy test set, these models have failed to learn how to infer spatial relations. We generate three kinds of distracting noise: *disconnected*, *irrelevant*, and *supporting*. Examples of all kinds of noise are provided in Figure 2. The irrelevant noise extends the original chain by branching it out with new entities and relations. The disconnected noise adds to the original chain a new independent chain with new entities and relations. The supporting noise adds to the original chain new entities and relations that may provide alternative reasoning paths. We only add supporting noise into chains with more than 2 entities. All kinds of noise have no impact on the correct answer. The type and amount of noise added to each chain is randomly determined. The detailed statistics for each type of distracting noise are provided in the Appendix.

4 The TP-MANN Model

In this section we introduce the proposed TP-MANN model, as shown in Figure 3. The model comprises three major components: a question and story encoder, a recurrent memory module, and a relation decoder. The encoder learns to represent entities and relations for each sentence in a story. The recurrent memory module learns to store entity-relation pair representations into the memory independently. It also updates the entity-relation pair representations based on the current memory and stores the inferred information. The decoder learns to represent the question and using the information stored in the memory recurrently infers the spatial relation of the two entities mentioned in the question.

It also has been shown that learned representations in the TPR-like memory could be orthogonal (Schlag and Schmidhuber 2018). We use an example to illustrate the inspiration behind this architecture. A person may experience that when she goes back to her hometown and sees an old tree, her happy childhood memory about playing with her friends under that tree might be recalled. However, this memory may not be reminisced unless triggered by the old tree appearance. In our model, unbinding vectors in the decoder module
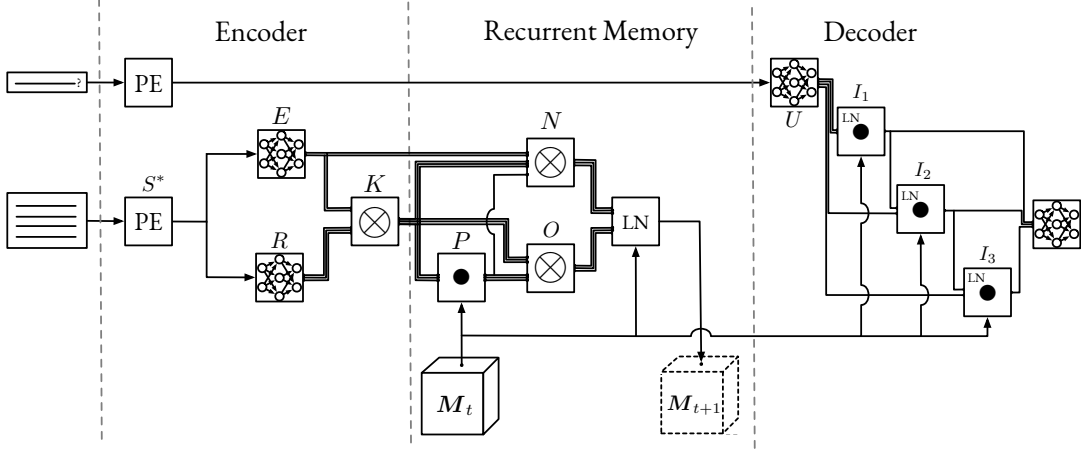
Figure 3: The TP-MANN architecture. PE stands for positional encoder, the sign in the box below the symbol E represents a feed-forward neural network, the $\otimes$ sign represents the outer-product operator, the $\bullet$ sign represents the inner product operator, and LN represents a layer normalization. The $\otimes$, $\bullet$, and LN boxes implement the formulae as presented in Section 4. Lines indicate the flow of information. Those without an arrow indicate which symbols are taken as input and are output by their box.

play the role of the old tree in the example, where unbinding vectors are learned based on the target questions. The decoder module unbinds relevant memories given a question via a recurrent mechanism. Moreover, although memories are stored separately, there are integration processes in brains that retrieve information via a recursive mechanism. This allows episodes in memories to interact with each other (Kumaran and McClelland 2012; Schapiro et al. 2017; Koster et al. 2018).

Encoder. The input of the encoder is a story and a question. Given a input story $S = (s_1, \ldots, s_m)$ with m sentences and a question q both described by words in a vocabulary V. Each sentence $s_i = (w_1, \ldots, w_n)$ is mapped to learnable embeddings $(w_1^*, \ldots, w_n^*)$. Then, a positional encoding (PE) is applied to each word embedding and then averaged together $s_i^* = \frac{1}{n} \sum_{j=1}^{n} w_j^* \cdot p_j$, where $\{p_1, \ldots, p_n\}$ are learnable position vectors, and $\cdot$ is the element-wise product. This operation defines $S^* \in \mathbb{R}^{m \times d}$, where each row of S^* represents an encoded sentence and d is the dimension of a word embedding. For the input question we convert it to a vector $q^* \in \mathbb{R}^d$ in the same way. For each sentence of the story in S^*, we learn entity and relation representations as:

$$E_i = f_{e_i}(S^*), i = 1, 2, \tag{4}$$
$$R_j = f_{r_j}(S^*), j = 1, 2, 3, \tag{5}$$

where f_{e_i} are feed-forward neural networks that output entity representations $E_i \in \mathbb{R}^{m \times d_e}$ and f_{r_j} are feed-forward neural networks that output relation representations $R_j \in \mathbb{R}^{m \times d_r}$. Finally, we define three search keys K as:

$$K_1 = E_1 \otimes R_1, \tag{6}$$
$$K_2 = E_1 \otimes R_2, \tag{7}$$
$$K_3 = E_2 \otimes R_3, \tag{8}$$

where $K_1, K_2, K_3 \in \mathbb{R}^{m \times d_e \times d_r}$. Keys will be used to manipulate the memory in the next module and retrieve potential existing associations for each entity-relation pair.

Recurrent Memory Module. To allow stored information to interact with each other, we use a recurrent architecture with T recurrent-layers to update the TPR-like memory representation $M \in \mathbb{R}^{d_e \times d_r \times d_e}$, where M contains trainable parameters. Through this recurrent architecture, existing episodes stored in memory can interact with new inferences to generate new episodes. Different from many models like Transformer (Vaswani et al. 2017) and graph-based models (Kipf and Welling 2017; Velickovic et al. 2018) where adding more layers in the model leads to a larger number of trainable parameters, our model will not increase the number of trainable parameters as the number of recurrent-layers increases.

At each layer t, given the keys Ks, we extract pseudo-entities Ps for each sentence in S^*. In the first layer ($t = 0$), since there is no previous information existing in memory M_0, the model just converts each sentence in S^* as an episode and stores them in it (M_1). Then at the later layers ($t > 0$), pseudo-entities Ps build bridges between episodes in the current memory M_t and allow them to interact with potential entity-relation associations.

$$P_{jt} = K_j \otimes M_t, j = 1, 2, 3, \tag{9}$$

where $P_{jt} \in \mathbb{R}^{m \times d_e}$. We then construct the memory episodes needed to be updated or removed. This is done after the first storage at $t = 0$ so that all story information is already available in M. These old episodes, $O_{jt} \in \mathbb{R}^{d_e \times d_r \times d_e}$, will be updated or removed to avoid memory conflicts that may occur when receiving new information:

$$O_{jt} = K_j \otimes P_{jt}, j = 1, 2, 3 \tag{10}$$

Afterwards, new episodes, N_1, N_{2t} and $N_3 \in \mathbb{R}^{d_e \times d_r \times d_e}$, will be added into the memory:

$$N_1 = K_1 \otimes E_2, \tag{11}$$
$$N_{2t} = K_2 \otimes P_{1t}, \tag{12}$$
$$N_3 = K_3 \otimes E_1. \tag{13}$$

Then we apply this change to the memory by removing (subtracting) old episodes and adding up the new ones to the now dated memory M_t:

$$M_{t+1} = \text{LN}(M_t + \\ + N_1 + N_{2t} + N_3 - O_{1t} - O_{2t} - O_{3t}), \quad (14)$$

where LN is a layer normalization.

Decoder. The prediction is computed based on the constructed memory M at the last layer and a question vector q. To do this we follow the same procedure designed by Schlag and Schmidhuber (2018):

$$U_j = f_j^u(q), j = 1, 2, 3, 4, \quad (15)$$

where f_1^u is a feed-forward neural network that outputs a d_e-dimensional unbinding vector, and f_2^u, f_3^u, f_4^u are feed-forward neural networks that output d_r-dimensional unbinding vectors. Then, the information stored in M will be retrieved in a recurrent way based on unbinding vectors learned from the question:

$$I_1 = \text{LN}(M_T \cdot U_1) \cdot U_2, \quad (16)$$

$$I_2 = \text{LN}(M_T \cdot I_1) \cdot U_3, \quad (17)$$

$$I_3 = \text{LN}(M_T \cdot I_2) \cdot U_4, \quad (18)$$

$$\hat{v} = \text{softmax}(W_o \cdot \sum_{j=1}^{3} I_j). \quad (19)$$

A linear projection of trainable parameters $W_o \in \mathbb{R}^{|V| \times d_e}$ and a softmax function are used to map the extracted information into $\hat{v} \in \mathbb{R}^{|V|}$. Hence, the decoder module outputs a probability distribution over the terms of the vocabulary V.

5 Experiments and Results

In this section we aim to address the following research questions: (**RQ1**) What is the degree of data leakage in the datasets? (**RQ2**) How does our model behave with respect to state-of-the-art NLU models in spatial reasoning tasks? (**RQ3**) How do these models behave when tested on examples more challenging than those used for training? (**RQ4**) What is the effect of the number of recurrent-layers in the recurrent memory module? Before answering these questions, we first present the material and baselines used in our experiments.

5.1 Material and Baselines

In the following experiments we will use two datasets, the bAbI dataset and the StepGame dataset. For the bAbI dataset we only focus on task 17 and task 19 and use the original train and test splits made of 10 000 samples for the training set and 1 000 for the validation and test sets. For the StepGame dataset, we generate a training set made of samples varying k from 1 to 5 at steps of 1, and a test set with k varying from 1 to 10. Moreover, the test set will also contain distracting noise. The final dataset consists of, for each k value, 10 000 training samples, 1 000 validation samples, and 10 000 test samples.

	Task 17	Task 19	Mean
RN	97.33±1.55	98.63±1.79	97.98
RRN	97.80±2.34	49.80±5.76	73.80
STM	97.80±1.06	99.98±0.05	98.89
UT	98.60±3.40	93.90±7.30	96.25
TPR-RNN	97.55±1.99	99.95±0.06	98.75
Ours	**99.88±0.10**	**99.98±0.04**	**99.93**

Table 1: Test accuracy on the task 17 and 19 of the bAbI dataset: Mean±Std over 5 runs.

We compare our model against five baselines: Recurrent Relational Networks (RRN) (Palm, Paquet, and Winther 2018), Relational Network (RN) (Santoro et al. 2017), TPR-RNN (Schlag and Schmidhuber 2018), Self-attentive Associative Memory (STM) (Le, Tran, and Venkatesh 2020), and Universal Transformer (UT) (Dehghani et al. 2019). Each model is trained and validated on each dataset independently following the hyper-parameter ranges and procedures provided in their original papers. All training details, including those for our model, are reported in the Appendix.

5.2 Training-Test Leakage

To answer **RQ1** we have calculated the degree of data leakage present in bAbI and the StepGame datasets. For the task 17, we counted how many samples in the test set appear also in the training set: 23.2% of the test samples are also in the training set. For task 19, for each sample we extracted the relevant sentences in the stories (i.e., those sentences necessary to answer the question correctly) and questions. Then we counted how many such pairs in the test set appear in the training set: 80.2% of the pairs overlap with pairs in the training set. For the StepGame dataset, for each sample we extracted the sentences in the stories and questions. The sentences in the story are sorted in lexicographical order. Then we counted how many such pairs in the test set appear also in the training set before adding distracting noise and using the templates: 1.09% of the pairs overlap with triples in the training set. However, such overlap is all produced by the samples with $k = 1$, which due to their limited number have a higher chance of being included in the test set. If we remove those examples, the overlap between training and test sets drops to 0%.

5.3 Spatial Inference

To answer **RQ2** and judge the spatial inference ability of our model and the baselines we train them on the bAbI and the StepGame datasets and compare them by measuring their test accuracy.

In Table 1 we present the results of our model and the baselines on the task 17 and 19 of the bAbI dataset. The performance of our model is slightly better than the best baseline. However, due to the issues of the bAbI dataset, these results are not enough to firmly answer RQ2.

In Table 2 we present the results for the StepGame dataset. In this dataset, the training set is without noise but the test set is with distracting noise. In the table we break down the performance of the trained models across k. In the last column

Model	k=1	k=2	k=3	k=4	k=5	Mean
RN (Santoro et al. 2017)	22.64±0.25	17.08±1.41	15.08±2.58	12.84±2.27	11.52±1.73	15.83
RRN (Palm, Paquet, and Winther 2018)	24.05±4.48	19.98±4.68	16.03±2.89	13.22±2.51	12.31±2.16	17.12
UT (Dehghani et al. 2019)	45.11±4.16	28.36±4.50	17.41±2.18	14.07±2.87	13.45±1.35	23.68
STM (Le, Tran, and Venkatesh 2020)	53.42±3.73	35.96±4.45	23.03±1.83	18.45±1.87	15.14±1.56	29.20
TPR-RNN (Schlag and Schmidhuber 2018)	70.29±3.03	46.03±2.24	36.14±2.66	26.82±2.64	24.77±2.75	40.81
Ours	**85.77±3.18**	**60.31±2.23**	**50.18±2.65**	**37.45±4.21**	**31.25±3.38**	**52.99**

Table 2: Test accuracy on the StepGame dataset: Mean±Std over 5 runs.

Model	$k = 6$	k=7	k=8	k=9	k=10	Mean
RN (Santoro et al. 2017)	11.12±0.96	11.53±0.70	11.21±0.98	11.13±1.00	11.34±0.87	11.27
RRN (Palm, Paquet, and Winther 2018)	11.62±0.80	11.40±0.76	11.83±0.75	11.22±0.86	11.69±1.40	11.56
UT (Dehghani et al. 2019)	12.73±2.37	12.11±1.52	11.40±0.92	11.41±0.96	11.74±1.07	11.88
STM (Le, Tran, and Venkatesh 2020)	13.80±1.95	12.63±1.69	11.54±1.61	11.30±1.13	11.77±0.93	12.21
TPR-RNN (Schlag and Schmidhuber 2018)	22.25±3.12	19.88±2.80	15.45±2.98	13.01±2.28	12.65±2.71	16.65
Ours	**28.53±3.59**	**26.45±2.95**	**23.67±2.78**	**22.52±2.36**	**21.46±1.72**	**24.53**

Table 3: Test accuracy on StepGame for larger ks (only on the test set). Mean±Std over 5 runs.

we report the average performance across k. Our model outperforms all the baseline models. Compared to Table 1, the decreased accuracy in Table 2 demonstrates the difficulty of spatial reasoning with distracting noise. It is not surprising that the performance of all five baseline models decreases when k increases, that is, when the number of required inference hops increases. We also report test accuracy on test sets without distracting noise in the Appendix.

5.4 Systematic Generalization

To answer **RQ3** we generate new StepGame test sets with $k \in \{6, 7, 8, 9, 10\}$ with distracting noise. We then test all the models jointly trained on the StepGame train set with $k \in \{1, 2, 3, 4, 5\}$ as in the Section 5.3. We can consider this experiment as a zero-shot learning setting for larger ks.

In Table 3 we present the performance of different models on this generalization task. Not surprisingly, the performance of all models degrades monotonically as we increase k. RN, RRN, UT and SAM fail to generalize to the test sets with higher k values, while our model is more robust and outperforms the baseline models with a large margin. This demonstrates the better generalization ability of our model, which performs well on longer stories never seen during training.

5.5 Inference Analysis

To answer **RQ4**, we conduct an analysis of the hyperparameter T, the number of recurrent-layers in our model. We jointly train TP-MANN on the StepGame dataset with k between 1 and 5 with number of T between 1 and 6 and report the break down test accuracy for each value of k. These results are shown in the left-hand side figure of Figure 4. The test sets with higher k benefit more from a higher number of recurrent layers than those with lower k, indicating that recurrent layers are critical for multi-hop reasoning. We also analyze how the recurrent layer structure affects systematic generalization. To do this we also test on a StepGame test set with k between 6 and 10 with noise. These ks are larger than the largest k used during training. These results are
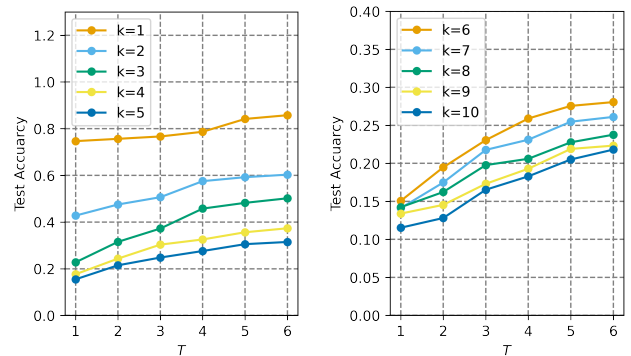

Figure 4: Analysis of TP-MANN's number of recurrent-layers (T). The x-axis is T with which the model has been trained. Each line represents a different value of k of the StepGame dataset.

shown in the right-hand side figure in Figure 4. Here we see that as T increases, the performance of the model improves. This analysis further corroborates that our recurrent structure supports multi-hop inference. It is worth noting, that the number of trainable parameters in our model remains unchanged as T increases. Interestingly, we find that the number of recurrent-layers needed to solve the task is less than the length of the stories k suggesting that the inference process may happen in parallel.

6 Conclusion

In this paper, we proposed a new dataset named StepGame that requires a robust multi-hop spatial reasoning ability to be solved and mitigates the issues observed in the bAbI dataset. Then, we introduced TP-MANN, a tensor product-based memory-augmented neural network architecture that achieves state-of-the-art performance on both datasets. Further analysis also demonstrated the importance of a recurrent memory module for multi-hop reasoning.

References

Anderson, P.; Wu, Q.; Teney, D.; Bruce, J.; Johnson, M.; Sünderhauf, N.; Reid, I. D.; Gould, S.; and van den Hengel, A. 2018. Vision-and-Language Navigation: Interpreting Visually-Grounded Navigation Instructions in Real Environments. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, 3674–3683. IEEE Computer Society.

Battaglia, P. W.; Hamrick, J. B.; Bapst, V.; Sanchez-Gonzalez, A.; Zambaldi, V.; Malinowski, M.; Tacchetti, A.; Raposo, D.; Santoro, A.; Faulkner, R.; et al. 2018. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*.

Bisk, Y.; Shih, K. J.; Choi, Y.; and Marcu, D. 2018. Learning Interpretable Spatial Operations in a Rich 3D Blocks World. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*. AAAI Press.

Chen, C.-H.; Fu, Y.-F.; Cheng, H.-H.; and Lin, S.-D. 2020a. Unseen Filler Generalization In Attention-based Natural Language Reasoning Models. In *2020 IEEE Second International Conference on Cognitive Machine Intelligence (CogMI)*, 42–51. IEEE.

Chen, H.; Suhr, A.; Misra, D.; Snavely, N.; and Artzi, Y. 2019. TOUCHDOWN: Natural Language Navigation and Spatial Reasoning in Visual Street Environments. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE.

Chen, K.; Huang, Q.; Palangi, H.; Smolensky, P.; Forbus, K. D.; and Gao, J. 2020b. Mapping natural-language problems to formal-language solutions using structured neural representations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, Proceedings of Machine Learning Research. PMLR.

Dehghani, M.; Gouws, S.; Vinyals, O.; Uszkoreit, J.; and Kaiser, L. 2019. Universal Transformers. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics.

Ding, D.; Hill, F.; Santoro, A.; and Botvinick, M. 2020. Object-based attention for spatio-temporal reasoning: Outperforming neuro-symbolic models with flexible distributed architectures. *arXiv preprint arXiv:2012.08508*.

Gershman, S.; and Tenenbaum, J. B. 2015. Phrase similarity in humans and machines. In *Proceedings of the 37th Annual Meeting of the Cognitive Science Society, CogSci 2015, Pasadena, California, USA, July 22-25, 2015*. cognitivesciencesociety.org.

Graves, A.; Wayne, G.; Reynolds, M.; Harley, T.; Danihelka, I.; Grabska-Barwińska, A.; Colmenarejo, S. G.; Grefenstette, E.; Ramalho, T.; Agapiou, J.; et al. 2016. Hybrid computing using a neural network with dynamic external memory. *Nature*.

Huang, Q.; Smolensky, P.; He, X.; Deng, L.; and Wu, D. 2018. Tensor Product Generation Networks for Deep NLP Modeling. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics.

Janner, M.; Narasimhan, K.; and Barzilay, R. 2018. Representation Learning for Grounded Spatial Reasoning. *Transactions of the Association for Computational Linguistics*.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Koster, R.; Chadwick, M. J.; Chen, Y.; Berron, D.; Banino, A.; Düzel, E.; Hassabis, D.; and Kumaran, D. 2018. Big-loop recurrence within the hippocampal system supports integration of information across episodes. *Neuron*, 99(6): 1342–1354.

Kruijff, G.-J. M.; Zender, H.; Jensfelt, P.; and Christensen, H. I. 2007. Situated dialogue and spatial organization: What, where... and why? *International Journal of Advanced Robotic Systems*, 4(1): 16.

Kumaran, D.; and McClelland, J. L. 2012. Generalization through the recurrent interaction of episodic memories: a model of the hippocampal system. *Psychological review*, 119(3): 573.

Landsiedel, C.; Rieser, V.; Walter, M.; and Wollherr, D. 2017. A review of spatial reasoning and interaction for real-world robotics. *Advanced Robotics*, 31(5): 222–242.

Le, H.; Tran, T.; and Venkatesh, S. 2020. Self-Attentive Associative Memory. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, 5682–5691. PMLR.

Liu, F.; and Perez, J. 2017. Gated End-to-End Memory Networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Valencia, Spain: Association for Computational Linguistics.

Logan, G. D.; and Sadler, D. D. 1996. A computational analysis of the apprehension of spatial relations.

Palm, R. B.; Paquet, U.; and Winther, O. 2018. Recurrent Relational Networks. In Bengio, S.; Wallach, H. M.; Larochelle, H.; Grauman, K.; Cesa-Bianchi, N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, 3372–3382.

Petruck, M. R. L.; and Ellsworth, M. J. 2018. Representing Spatial Relations in FrameNet. In *Proceedings of the First International Workshop on Spatial Language Understanding*, 41–45. New Orleans: Association for Computational Linguistics.

Pustejovsky, J. 1989. Language and Spatial Cognition. *Computational Linguistics*, 15(3).

Pustejovsky, J.; Kordjamshidi, P.; Moens, M.-F.; Levine, A.; Dworman, S.; and Yocum, Z. 2015. SemEval-2015 Task 8: SpaceEval. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*. Denver, Colorado: Association for Computational Linguistics.

Santoro, A.; Raposo, D.; Barrett, D. G. T.; Malinowski, M.; Pascanu, R.; Battaglia, P. W.; and Lillicrap, T. 2017. A simple neural network module for relational reasoning. In Guyon, I.; von Luxburg, U.; Bengio, S.; Wallach, H. M.; Fergus, R.; Vishwanathan, S. V. N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, 4967–4976.

Schapiro, A. C.; Turk-Browne, N. B.; Botvinick, M. M.; and Norman, K. A. 2017. Complementary learning systems within the hippocampus: a neural network modelling approach to reconciling episodic memory with statistical learning. *Philosophical Transactions of the Royal Society B: Biological Sciences*.

Schlag, I.; Munkhdalai, T.; and Schmidhuber, J. 2021. Learning Associative Inference Using Fast Weight Memory. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*.

Schlag, I.; and Schmidhuber, J. 2018. Learning to Reason with Third Order Tensor Products. In Bengio, S.; Wallach, H. M.; Larochelle, H.; Grauman, K.; Cesa-Bianchi, N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, 10003–10014.

Smolensky, P. 1990. Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial intelligence*, 46(1-2): 159–216.

Sukhbaatar, S.; Szlam, A.; Weston, J.; and Fergus, R. 2015. End-To-End Memory Networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, 2440–2448.

Talmor, A.; and Berant, J. 2018. The Web as a Knowledge-Base for Answering Complex Questions. In Walker, M. A.; Ji, H.; and Stent, A., eds., *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)*, 641–651. Association for Computational Linguistics.

Tan, H.; and Bansal, M. 2018. Source-Target Inference Models for Spatial Instruction Understanding. In McIlraith, S. A.; and Weinberger, K. Q., eds., *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, 5504–5511. AAAI Press.

Tversky, B. 2019. *Mind in motion: How action shapes thought*. Hachette UK.

van Aken, B.; Winter, B.; Löser, A.; and Gers, F. A. 2019. How Does BERT Answer Questions?: A Layer-Wise Analysis of Transformer Representations. In Zhu, W.; Tao, D.; Cheng, X.; Cui, P.; Rundensteiner, E. A.; Carmel, D.; He, Q.; and Yu, J. X., eds., *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, 1823–1832. ACM.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is All you Need. In Guyon, I.; von Luxburg, U.; Bengio, S.; Wallach, H. M.; Fergus, R.; Vishwanathan, S. V. N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, 5998–6008.

Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Vogel, A.; and Jurafsky, D. 2010. Learning to Follow Navigational Directions. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, 806–814. Uppsala, Sweden: Association for Computational Linguistics.

Welbl, J.; Stenetorp, P.; and Riedel, S. 2018. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics*, 6: 287–302.

Weston, J.; Bordes, A.; Chopra, S.; and Mikolov, T. 2016. Towards AI-Complete Question Answering: A Set of Prerequisite Toy Tasks. In Bengio, Y.; and LeCun, Y., eds., *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Yang, T.-Y.; Lan, A.; and Narasimhan, K. 2020. Robust and Interpretable Grounding of Spatial References with Relation Networks. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 1908–1923. Online: Association for Computational Linguistics.

Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W. W.; Salakhutdinov, R.; and Manning, C. D. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, 2369–2380. Association for Computational Linguistics.