Cascaded Reasoning Network

Işık Kuntay[1♠] , Ayşin Taşdelen[2], Marie Li[3]

[1]Wayne State University, Department of Engineering, Detroit, MI USA
[2]Istanbul Bilgi University, Department of Computer Science, İstanbul, Türkiye
[3]California State University East Bay, Department of Statistics, Hayward, CA USA

ABSTRACT

In this paper we introduce a new approach to computational reasoning using an LSTM that has four inputs representing four different sources including one from a previous network and another from a knowledge base, called CRN. The network is able to extract the relevant information from the inputs and compress them to a concise, symbolic output that can become an input for the next network. This network thereby makes successive inferences in a cascaded manner, achieving a chain of reasoning with the end result as the answer to the main question, similar to how humans perform step by step reasoning to solve complex problems. CRN is also able to command the computer to compute a formula or provide more information. We demonstrate the efficacy of CRN by solving a modified version of the bAbI QA database. After the augmentation of the dataset with labels for intermediate reasoning, CRN achieved the score of 20/20 on all tasks in the bAbI, being first to do so.

Key Words: Artificial General Intelligence, Natural Language Processing, Reasoning, Neural Networks, LSTM

1. INTRODUCTION

In their book, Russel and Norvig [1] organize historical approaches to Artificial Intelligence in four categories: Thinking Humanly, Thinking Rationally, Acting Humanly, and Acting Rationally. They describe the approach of Thinking Humanly as an attempt to understand human minds. The approach of Thinking Rationally, on the other hand, is based on classical logic. While logic has a clear set of rules, the logical approach has the shortcomings of having to convert the entire domain being studied into symbolic notation, especially considering the large number of facts in a real-life situation. Planning languages such as PDDL [2] are results of the Thinking Rationally approach and utilize a combination of search algorithms and logical deductions to solve complex planning problems. Decision Making Systems, which are also logic-based, have been extensively used by the industry and government for decades [3]. The strength that enables rational systems to be trustworthy, namely their rigidness, however, is also the very weakness that limits them to the confines of the specific problems they were programmed to solve.

The success of logic-based systems is based on the successful representation of the problem domain in standardized form. In game environments, such as video games, chess, or Go, symbolic representation is straightforward and AI has enjoyed superhuman success in these fields. While winning over world champions in games like Go were impressive, these programs could not be classified within the scope of Artificial General Intelligence (AGI) but rather within scope of Narrow Intelligence [4].

We can think of humans having a sort of General Intelligence as opposed to the Narrow Intelligene of computers. In order to excel in real-world problems, computers need to be able to make sense of information that is not in standard form [5]. The advances of probabilistic methods such as Neural Networks (NN) have endowed computers with the ability to model highly unstructured data. Thinking humanly requires the ability to understand human language as language is the foundation of human intelligence that allows us to organize the otherwise vague concepts in our minds. Many advances have been made to this extent, among the most impactful one being the development of LSTMs [6]. However, for humans, concepts are not limited to words but are comprised of general memories pertaining to all five of our senses. Major advances have been made in NN processing of visual and audial data as well. However, there is yet to be a program that has the problem solving abilities of an average person.

The ability to reason about the relations between entities and their properties is central to generally intelligent behavior. Symbolic approaches to artificial intelligence are inherently relational, but suffer from the Symbol Grounding Problem (SGP) [7]. An AGI program must then be able to reason with the unstructured information it acquires and develop strategies based on reasoning on a wide variety of inputs. The quest is to find a way to process highly unstructured data from the real world and make use of grounded reasoning while avoiding SGP.

2. BACKGROUND – REASONING NETWORKS

Recently several studies have used NNs in identifying relations between objects and making inferences from a

♠Corresponding author, e-mail: xkuntay@yahoo.com

series of observations [8, 9, 10]. These models made use of syntetic datasets that were specifically developed for training networks to reason. The bAbI is one such Question and Answer (QA) dataset that is purely text-based [11]. One of some models tested with the bAbI [12, 13] is the Relation Network (RN) of Google DeepMind, which scored 18/20 in the tests. This model has impressive versatility, being applicable to both visual [14] and textual datasets, and a relatively simple design. It consists of two groups of networks; one of parallel networks that accept informational inputs and one of three MLP units in series that outputs the answer.

The overall RN model is expressed as:

$$\mathrm{RN}(O) = f_\phi\left(\sum_{i,j} g_\theta\left(o_i, o_j,\right)\right) \tag{1}$$

where the input is a set of “objects” O = {$o_1, o_2, ..., o_n$}, $o_i \in$ R m is the i th object, and f_φ and g_θ are functions with parameters φ and θ, respectively. We will sometimes denote g_θ as g(x). In the RN paper, f_φ and g_θ are MLPs (fully connected dense layers), and the parameters are learnable synaptic weights. RNs have three notable strengths: they learn to infer relations, they are data efficient, and they operate on a set of objects – a particularly general and versatile input format – in a manner that is order invariant [13].

Cascaded Reasoning Networks

In this paper, we present a model similar to RN, albeit with fundamental differences, called the Cascaded Reasoning Network (CRN), which takes four text strings as input and yields one string as output. The four inputs are: Question Sentence, Information Sentence, Fact Sentence, and Relation Sentence. The Question Sentence is the question that is asked to the network to which we want to find the correct answer. The Information Sentence is one of the information sentences given to the network in the problem description. The information sentences could be on a sequence of events or just a list of facts about the objects in question.

The Fact Sentence contains information that the network can learn to inquire from a library of general information. In this study we used a single placeholder string, “NONE”, for fact sentences since this was designed for future development. Ideally, a second NN should be trained to provide the CRN with the proper facts by searching the internet. This can be compared to the external information source in the Chinese room argument by John Searle [15].

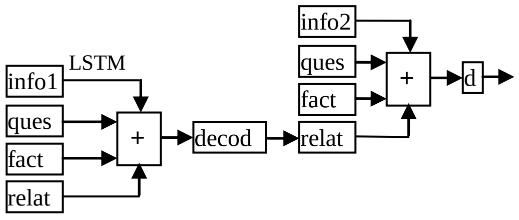

Figure 1. The architecture of two-layer CRN.

The Relation Sentence is like a summary of all the information the network collected from previous runs. It is similar in concept to the long term memory in LSTMs, except that it is in symbolic format instead of a state vector. The Relation Sentence can also be controlled by a specialized network and only include most current information relevant to the question. The output of the CRN is either a symbolic relation equation to be used in the next CRN or a command to the main software to acquire more information. Thus, CRN can communicate with the rest of the software via its output.

Properties of CRN

Reasoning in NNs is a statistical phenomenon where a certain pattern in the inputs results in a certain probability distribution in a list of outputs. The NN is trained to find the optimum for the objective function:

$$\mathrm{Q}(\theta) = \sum_{j=1}^{M}\sum_{i=1}^{N} log\left(P\left(C_i | X_j, \theta\right)\right) \tag{2}$$

where θ are the model parameters, *O*(θ) is the objective function, j is the index for M inputs, X_j is input number j, i is the index for N classes, C_i is the output class number i, P is the probability of observing class C_i given the input X_j and parameters θ.

In any reasoning network, including RNs, reasoning will start with identifications, since no reasoning can be made before distinguishing between objects. Although we cannot translate the matrix coming out of each g(x) function of RN into features conceivable for humans, we can guess that the network will try to first identify the objects in question. This information will most likely be included in the embedding matrix o_i, and g(x) will be able to yield a comparison to whether the two objects are close or not. If n objects with k features each are in consideration, there are a total of n.k features. If combinations of all features in pairs are considered, the number of combinations will be:

$$N_c = \frac{(n.k)!}{(2!\,(n.k-2)!)} \tag{3}$$

This is a large number, in the order of $(n.k)^2$. To reason about all these pairs requires considering them in pairs again, which yields an order of $(n.k)^4$. If we continue further reasoning, the order grows in powers of 2. The function for second-level reasoning can be expressed as:

$$\mathrm{RN}(O) = f_\phi\left(\sum_{ij}\sum_{l,k} g_\gamma\left(g_\theta\left(o_i, o_j\right), g_\theta\left(o_l, o_k\right)\right)\right) \tag{4}$$

where each layer of g(x) functions adds a power of 2 to the number of combinations to be reasoned about.

For sophisticated questions that require many steps of reasoning, neither will the network be able to handle the resulting possibilities, nor will we be able to provide sufficient number of examples for it to overcome the ambiguities. Therefore, somewhere along the graph the network needs to be guided in a logical direction. By doing this, the number of combinations in the reasoning process can be restricted.

In RN, the network may or may not be following human reasoning patterns. It could simply be evaluating numerical combinations, such as the number of times a certain word has appeared before another, and given thousands of examples concludes the most probable answer. As such, we do not have much control over the reasoning pattern of the network. If, on the other hand, the network is shown how to take smaller steps at a time, it can handle significantly more sophisticated problems. The main purpose of the network should be extraction of data from sophisticated human language and putting this into a framework similar to those of classical logic-based algorithms.

The RN converges to an answer and leaves the intermediate g(x) networks as “hidden” to the user. The output of the RN is the answer to the main question and can be used as an input for another network. However, since the questions in QA databases are very specific, the answer may not be of much use for the next level. It makes sense to teach the network simpler reasoning patterns and how to perform them sequentially. This is what CRN was developed for.

3. MODEL

There are three main differences between CRN and RN:

1.) CRN uses independently trained functions for the reasoning. In other words, the g(x) networks are trained specifically for intermediate reasoning. The output of one can be fed as an input to the other in a cascaded fashion.

2.) One of the inputs of CRN, the Relation Sentence, is in symbolic format. The symbolic notation bears concise information extracted and filtered from the information sentence in the previous step.

3.) We represent words for subjects, objects, and certain verbs with placeholders. This is to take advantage of the symmetry between the sentences. This significantly reduces network size requirements without the loss of information.

The advantage of teaching each g(x) function how to reason with the input from another g(x) function is that we can cascade as many functions as we like without having to worry about the growing number of weight matrices in the network. There are three methods to accomplish this:

a.) Training the g(x) networks such that they yield outputs that mimic human way of thinking in symbolic form.

b.) Training the $g_\theta(o_i,o_j)$ networks as part of a three-layer overall network and later connecting multiples of the middle g(x) network with initialized weights to construct a single, large, and sequential network that is to be trained again for the final output.

c) A blend of the two methods above, where the network has several layers connected with the hidden layers and trained in its entirety for most time but it is possible to train each layer individually using “training attachments”.

Figures 1 and 2 depict the above-described methods for connecting the networks sequentially. Figure 1 shows method (a), which is essentially the CRN architecture. It is a simple network with individual modules trained to output symbolic results that are to be input to the next step. The second one depicts method (b). Figure 3 depicts a hybrid of CRN and RN where the state vectors of three CRNs are concatenated and input to an MLP (i.e. Dense) layer, displayed here for comparison.

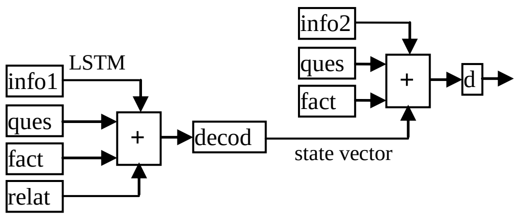

Figure 2. CRN with state vector as input.

Essentially, the design in Figure 2 consists of g(x) functions connected sequentially through their state vectors and the final network outputs the answer. Constructing such a network would require a g(x) network that takes as input the embedding matrix, or state vector, of another g(x) network and outputs another state vector that is to be fed into the next g(x). To obtain such a g(x) function, we would first need to construct a network with 3 layers, or levels, of g(x) such as in Figure 4. At the first level, basic relations between objects are considered such as “person A went to the location Y” and “person A got object G”. At second level, combinations of relations will be considered, such as “object G at location Y”. At each level, the number of combinations increases, since we have more combinations of where each person and object can be.

The nodes at the middle layer in Figure 4 would each be a g(x) network accepting and outputting state vectors. If these were concatenated and input into the final MLP layer of an RN, the resulting network would have a significant number of parameters. The outputs of the g(x) functions would not make sense for humans as they will be state vectors comprised of numbers. Until a visualizing method similar to Deconvolutional Networks [16] is invented, we have no way of looking into the inner works of RNs or other reasoning networks and we can only make assumptions. If we assumed that the reasoning patterns of the middle layers are duplicable and that this network could be trained, the g(x) networks in the middle could then be duplicated as additional layers, making an even more sophisticated network. The weights in these middle layers could then be fixed and the entire network retrained as such for more sophisticated tasks.

♠Corresponding author, e-mail: xkuntay@yahoo.com

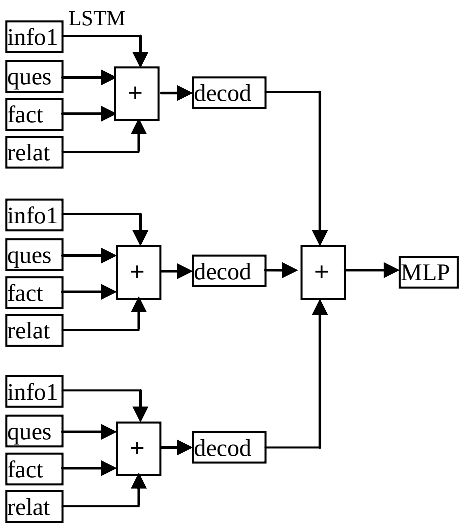

Figure 3. A hybrid of CRN and RN architectures.

One of the main reasons of our decision to use the first method was the fact that it enabled us to have control over the reasoning process. As humans, we have a certain logical process that is well established. While the method we use requires the manual production of training material, the effort is paid off with networks that know how to reason from each other's output and therefore can be cascaded multiple levels without computational problems. As the information sentences are processed by the CRN along with the facts and relations, information that is useful for answering the question is extracted and output in concise symbolic form. Representing a long sentence with just two symbols could be considered as a form of data compression. Successive data compression can allow the network to solve sophisticated problems.

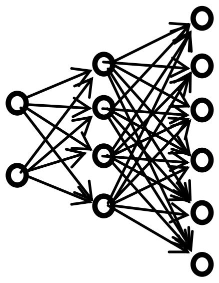

Figure 4. Representation of higher levels of reasoning.

As explained in method (c), however, an intermediate exists. The CRN can be made deeper with 3-5 layers connected in series by hidden layer outputs and trained towards a solving more sophisticated problems. Then, each layer can also be individually trained by shutting off the output of other layers and attaching an output layer to its hidden layer. We call this a "training attachment". This can be thought of as tapping into the output of each hidden layer and guiding the decision process of the combined network with given steps of reasoning. It is not too different from the method used in this paper except that the connections between layers are through their hidden layers.

The third difference between CRNs and RNs is the representation of nouns and verbs in each dataset with placeholders. Not all nouns and verbs were substituted with placeholders, however. Common words such as "take" or "get" were not substituted. The choice was based on the frequency of use for each word in the training set. The purpose of using placeholders is to take advantage of the symmetry between pairs of questions and answers. Considering, for example, the information sentence "Sharon went to the dining room" along with the question sentence "Where did Sharon go?", the network would still yield the correct answer if it were trained to answer "Where did X go?" with the information sentence "X went to the D" provided that a dictionary saved the information "X = Sharon" and "D = Dining Room". There is no loss of information when we conserve the actual words corresponding to the placeholders and train the network with the abstract symbolic form. When we print the results, we convert back to the original names using a reverse dictionary.

Perhaps the most important aspect of CRNs, the ability to reason with a wide range of vocabulary, arises from the utilization of placeholders. It would be unreasonable to expect a network to be able to learn many different tasks along with a large vocabulary. What the network needs is the significance of each word in the context of the question at hand. For example, if the network needs to answer where the football is, it does not need to know if the football is round or not. On the other hand, if the network is asked what will happen if a box is on top of a football, it will need to know that the football can roll. If these facts are supplied to the network as needed, it will have enough capacity to learn the task that it is expected to do, namely to be able to reason. In short, the CRN is trained to learn more by itself.

We want symbolic input and output from our reasoning network because we want it to be able to communicate with a knowledge-base (KB), which we see as essential for a network to make autonomous decisions for more sophisticated tasks (Figure 5). The CRN should be able to inquire information from the KB by means of a command from its output. This can only be achieved if we specifically train the network with the necessary labels. This requires several functions working in tandem. The output from one function becomes the input of the next. The equations for the functions operating in the CRN are expressed as follows:

$$\mathrm{CRN}(O) = f_\phi(\mathrm{I}_{j+1}, \mathrm{Q}, \mathrm{F}_{j+1}, \mathrm{R}_{j+1}) \quad (5)$$

$$\mathrm{R}_{j+1} = f_\phi(\mathrm{I}_j, \mathrm{Q}, \mathrm{F}_j, \mathrm{R}_j) \quad (6)$$

$$\mathrm{F}_j = \mathbf{k}_\sigma(\mathrm{I}_j, \mathrm{Q}, \mathrm{R}_j) \quad (7)$$

where CRN(O) is the final network that outputs the answer to the question Q, f_ϕ is the decoder network that outputs a string based on the inputs **I**, **Q**, **F**, and **R**, which respectively correspond to information, question, fact, and relation sentences in matrix forms. R_j, is the relation sentence at the jth level. The cascaded process arises from the relation between R_j and $R_{j+1.}$ Each network uses the output of the last one. F_j is the fact sentence at the jth level. Ideally F_j is a function k_σ of I, Q, and R. This would be a black-box function, meaning it would not be differentiable.

The CRN communicates with a KB through this k_σ function and acquires facts that are relevant to solving the problem (i.e. for Q = "Why did the orange roll down the hill?" a fact, F, could be "round objects roll"). The CRN also modifies the KB with commands given in its output (i.e. for "Mary went to the dining room" the KB can be modified as "Mary in dining-room). An exterior function monitors the output of the CRN and executes the command given once it is detected.

Another fundamental difference between CRN and the RN is having a Decoder instead of an MLP classifier. This is not a big difference, however, as the Decoder can be replaced with a stack of MLP layers. The two approaches differ in the representation of the target sequence, and hence the size of the final MLP layer. The Decoder served our purpose well, but at times the output was noisy and therefore the succeeding networks received noisy inputs.

It is also worthwhile to note that the CRN network we used for the experiments presented in this paper consists of only 4 encoders with 256 units each and a decoder with 1024 units, significantly lower than the number of units in other reasoning networks. To accommodate more tasks in a wider scope, size of LSTMs, especially that of the Relation Sentence, can be increased.

4. RESULTS

We applied the CRN architecture to a modified version of the bAbI. This dataset contains tasks, or questions that require different types of reasoning grouped in the 20 tasks such as single supporting fact, two supporting fact, yes-no question, counting, basic conference, conjunction, time reasoning, basic deduction, basic induction, path finding, and agents motivations [11]. Each task comes with a list of information sentences followed by a question and the answer to it as the label. In our application, we used an augmented version of the bAbI where there is a label following each information sentence. These are the expected compressed information extracted by the network with the combined evaluation of the information, question, fact, and relation sentences.

Table 1 below shows a sample test run. The information sentences are input one by one into the CRN. The CRN is trained to ask questions to the KB. The KB then gives the answer in a specific format. Each word corresponds to a token. For tokens, we used one-hot coding. No word embeddings were used, mainly for simplicity. The outputs from the four LSTMs are concatenated and fed into the decoder unit. The CRN also guides its own reasoning process through its output. In the first run in Table 1, the CRN outputs the question "where was Mary" in symbolic form consisting of 3 tokens. The "QUESTION:" token triggers a second run with the same question and info sentences but the relation sentence augemented with the answer to the inquiry, which leads to the answer to the main question.

Table 1. Sample Test Run on CRN.

Run 1	
Question	Where was the football before the bathroom
Info	Mary went to the bathroom
Fact	NONE
Relation	football with Mary
Output	QUESTION: WHERE-WAS Mary
Run 2	
Question	Where was the football before the bathroom
Info	Mary went to the bathroom
Fact	NONE
Relation	football with Mary ; Mary was-in office
Output	ANSWER: office

With the additional training data for step-by-step reasoning, simplification of sentences using single word for all synonyms, and the representation of nouns and verbs with symbolic placeholders, our model succeeded on all of the 20 tasks. The training was done on 100 samples per task. Given its efficiency in representation and evaluation, the network is able to learn many tasks simultaneously with modest amount of data.

5. DISCUSSION AND CONCLUSIONS

We have not tested our network for any visual inputs. This could be done in a fashion similar to that in RN. Incidentally, standard CNNs already have inference capabilities as they conduct a collective analysis of features to infer the object in the picture. It is possible to design a very deep architecture and ask highly complicated questions to the network and with sufficient data and time it may indeed find relations between objects on its own, but the data and time required would most likely be infeasible due to the size of the network. The question "what size is the triangular prism?" will take an order of magnitude less data and time than "how many other objects are there the same size as the triangular prism?". The ambiguity in the second question is an order of magnitude more than the one in first, suggesting that we need an order of magnitude more data for accurate results.

The CRN was designed to be able to reason many steps autonomously such that it could solve very difficult tasks requiring hundreds of reasoning steps. In simple cases, there is no need for additional inference to answer the question. For example for the question "Who picked up the football?" given the information sentence "Sandra picked up the football", there is no need to use more than one level of reasoning. In such cases, CRN has no advantage over RN. The advantage is exists for more sophisticated and diversified tasks.

We acknowledge that the main challenge for CRNs is the production of training data. Since CRN operates on 4 sentences, the data will have many combinations. This is not different than creating data for other QA

♠Corresponding author, e-mail: xkuntay@yahoo.com

networks, however, as all QA datasets require significant amount of labor. Since the internet does not usually have these types of data readily available, they are manually produced. Manual production of the CRN data is slightly more involved than that for RN or other QA network, but we see this as a logistic rather than computational challenge and one that is worth working on if AGI is the main objective.

We note that the visual component of reasoning is essential for solving most engineering problems and must eventually be incorporated into any reasoning algorithm. The overall reasoning can once again be divided into simpler steps and the CRN trained on the embedding matrix, or state vector, of a CNN. In this case, the information sentence can be replaced with this embedding matrix. The CRN can make successive runs using this output along with the output of the preceding CRN and eventually lead to the main answer.

Replacing numbers with placeholders allows CRNs to perform abstract math. Even the simplest NN has a weight matrix, which makes NNs inefficient for simple math operations. For instance, to add 15 and 25, a NN would need to perform matrix multiplication. It is more reasonable to have a parsing network identify numbers and replace them with symbols, while the NN is trained to output the result in symbolic form that is to be run with real numbers by the computer itself. We also support the use of a specialized NN for each task as this is much more efficient than training a single network to multi-task. A question classifying network can easily assign each task to the corresponding network. While this was not performed in this paper, for future research this is the path we intend follow.

In Figures 5 and 6 we demonstrate the importance of the communication between the CRN and KBs. Programming with KBs is a well-established field. Our main goal is to convert the real world data into a set of KBs. Many techniques, such as Random Forests [17] and Bayesian Networks [18], can be applied to solve complex problems with elaborate KBs that contain relevant and dynamic information on each of the objects. While not explicitly depicted in the figures, there are sources of general knowledge, such as internet, for retrieving information. These are considered as external Kbs, different than the KBs that pertain to specific objects and people. General information is to be supplied by networks trained to find this information on the internet.

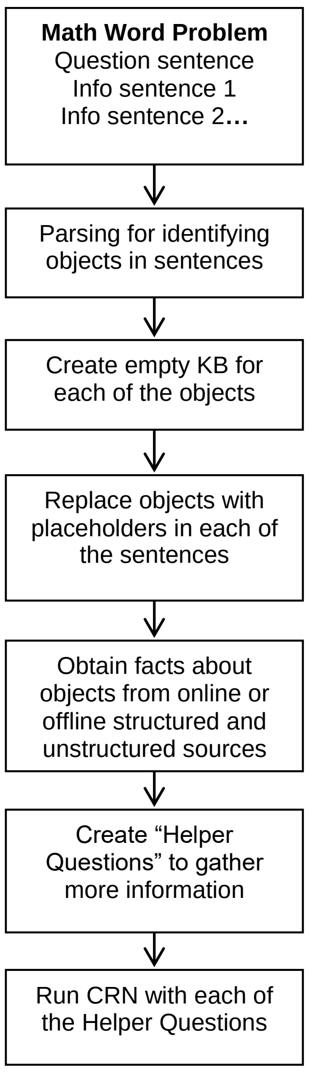
Math Word Problem
Question sentence
Info sentence 1
Info sentence 2...
Parsing for identifying objects in sentences
Create empty KB for each of the objects
Replace objects with placeholders in each of the sentences
Obtain facts about objects from online or offline structured and unstructured sources
Create "Helper Questions" to gather more information
Run CRN with each of the Helper Questions

Figure 5. Flowchart of operations for solving math word problems with the help of CRN.

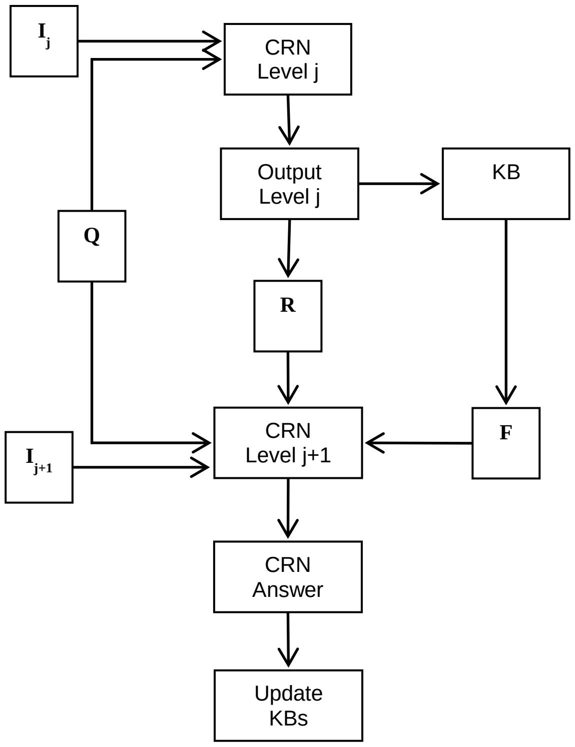

Figure 6. The flow of information through a CRN.

Below is a typical fourth grade math problem. As it involves understanding of and reasoning about concepts, it is a good test for AGI. The problem consists of information and question sentences.

Information sentences: “There are 4 doctors working in a clinic. Each doctor has 2 nurses assisting them. There are two receptionists, Jay and Molly, working at the reception.”

Question sentence: “How many people are working in the clinic?”

Apparently this problem requires the answering of such intermediary questions as:

1. Who are considered “people”?
2. Is “assisting doctors” a type of work?
3. What is the mathematical significance of “each”?
4. Is “reception” part of the “clinic”?

These are questions beyond any NN can simultaneously answer without the help of a KB. For this reason the interaction between NN and KB is essential. With sufficient training examples crafted with the common thought process of people and with adequate MLP units, the CRN can learn to ask the right questions. Another network, which is to be developed, can search for the answers in text-based databases and turn them into symbolic notation such as “doctors KIND-OF PEOPLE”. This way, the network does not necessarily need to know the exact words in the sentence. It just needs to know certain relationships between them.

Training the NN with each specific word would inundate its weight matrices. This is the case with numbers as well. Without symbolic representation, the above problem would require the NN to perform the simple operation of 4*2 + 2 = 10 using trained weights. The CRN on the other hand is trained to substitute the numbers with symbols and output an abstract math equation such as IN1*IN2 + IN3, where IN1, IN2, and IN3 are numbers of doctors, nurses, and receptionists.

REFERENCES

[1] Russell S. J., and Norvig P., “Artificial Intelligence: A Modern Approach”, ***Pearson Education Limited,*** (2010).

[2] McDermott, D., Ghallab M., Howe A., Knoblock C., Ram A., Veloso M., Weld D., and Wilkins D., "PDDL: The Planning Domain Definition Language", ***Yale Center for CVC Technical Report***, TR98003/DCS TR1165, (1998).

[3] Sprague, R, "A Framework for the Development of Decision Support Systems", ***MIS Quarterly***, Vol. 4, No. 4, pp.1-25, (1980).

[4] Franz A., “Artificial general intelligence through recursive data compression and grounded reasoning”, **arXiv**:1506.04366, (2015).

[5] Rosenbloom P. S., Demski A., and Ustun V., “The Sigma Cognitive Architecture and System: Towards Functionally Elegant Grand Unification”, ***Journal of Artificial General Intelligence***, 1946-0163, 1-103, (2016).

[6] Gers F. A., Schmidhuber J., and Cummins F. "Learning to Forget: Continual Prediction with LSTM", ***Neural Computation***, 12(1): 2451 – 2471, (2000).

[7] Harnad, S., “The Symbol Grounding Problem”, ***Physica D***, 42: 335-346, (1990).

[8] Agrawal A., Lu J., Antol S., Mitchell M., Zitnick C. L., Batra D., and Parikh D., “VQA: Visual Question Answering”, ***International Conference on Computer Vision***, 2380-7504, (2016).

[9] Scarselli F., Gori M., Tsoi A. C., Hagenbuchner M., and Monfardini G.. “The graph neural network model” ***IEEE Transactions on Neural Networks***, 20(1): 61 – 80, (2009).

[10] Li Y., Tarlow D., Brockschmidt M., and Zemel R., “Gated graph sequence neural networks”, **arXiv**:1511.05493, (2016).

[11] Weston J., Bordes A., Chopra S., Rush A. M., Merriënboer B., Joulin A., Mikolov T., “Towards AI-Complete Q & A: A Set of Prerequisite Toy Tasks”, **arXiv**:1502.05698, (2015).

♠Corresponding author, e-mail: xkuntay@yahoo.com

[12] Berant J., Chou A., Frostig R., and Liang P.. “Semantic parsing on freebase from question-answer pairs”, ***EMNLP***, 1533–1544, (2013).

[13] Santoro A., Raposo D., Barrett D. G. T., Malinowski M., Pascanu R., Battaglia P., Lillicrap T., “A simple neural network module for relational reasoning”, **arXiv**:1706.01427, (2017).

[14] Johnson J., Hariharan B., Maaten L. v. d., Fei-Fei L., Zitnick C. L., and Girshick R., “Clevr: A diagnostic dataset for compositional language and elementary visual reasoning”, **arXiv**:1612.06890, (2017).

[15] Searle, J., "Chinese room argument", ***Scholarpedia***, 4 (8): 310, (2009).

[16] Zeiler M. D., Fergus R., “Visualizing and Understanding Convolutional Networks”, **arXiv**:1311.2901, (2013).

[17] Breiman, L., “Random Forests”, ***Machine Learning,*** 45: 5, (2001).

[18] Margaritis D., "Learning Bayesian Network Model Structure from Data", Carnegie Mellon University School of Computer Science, ***Ph.D. Thesis***, Pittsburgh, (2003).

[19] Battaglia P., Pascanu R., Lai M., Rezende D. J., “Interaction networks for learning about objects, relations and physics.”, **arXiv**:1612.00222, (2016).