HAL open science

Using Relational Concept Networks for Explainable Decision Support

Paolo De Heer, Jeroen Voogd, Kim Veltman, Patrick Hanckmann, Jeroen Van Lith

To cite this version:

Paolo De Heer, Jeroen Voogd, Kim Veltman, Patrick Hanckmann, Jeroen Van Lith. Using Relational Concept Networks for Explainable Decision Support. 3rd International Cross-Domain Conference for Machine Learning and Knowledge Extraction (CD-MAKE), Aug 2019, Canterbury, United Kingdom. pp.78-93, 10.1007/978-3-030-29726-8_6 . hal-02520050

HAL Id: hal-02520050

https://inria.hal.science/hal-02520050

Submitted on 26 Mar 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

Using Relational Concept Networks for Explainable Decision Support

Paolo de Heer, Jeroen Voogd, Kim Veltman, Patrick Hanckmann, and Jeroen van Lith

Netherlands Organisation for Applied Scientific Research (TNO) Defence, Safety & Security, Oude Waalsdorperweg 63, 2597 AK Den Haag, The Netherlands
{paolo.deheer,kim.veltman}@tno.nl

Abstract. In decision support systems, information from many different sources must be integrated and interpreted to aid the process of gaining situational understanding. These systems assist users in making the right decisions, for example when under time pressure. In this work, we discuss a controlled automated support tool for gaining situational understanding, where multiple sources of information are integrated.
In the domain of operational safety and security, available data is often limited and insufficient for sub-symbolic approaches such as neural networks. Experts generally have high level (symbolic) knowledge but may lack the ability to adapt and apply that knowledge to the current situation. In this work, we combine sub-symbolic information and technologies (machine learning) with symbolic knowledge and technologies (from experts or ontologies). This combination offers the potential to steer the interpretation of the little data available with the knowledge of the expert.
We created a framework that consists of concepts and relations between those concepts, for which the exact relational importance is not necessarily specified. A machine-learning approach is used to determine the relations that fit the available data. The use of symbolic concepts allows for properties such as explainability and controllability. The framework was tested with expert rules on an attribute dataset of vehicles. The performance with incomplete inputs or smaller training sets was compared to a traditional fully-connected neural network. The results show it as a viable alternative when data is limited or incomplete, and that more semantic meaning can be extracted from the activations of concepts.

Keywords: Symbolic AI · Neural Networks · Graph-based Machine Learning, · Explainability · Decision support.

1 Introduction

Military decision making often takes place in uncertain and complex situations, and can have a large impact on the opponent and also on civilians and cause collateral damage. Often the complete impact of a decision cannot be easily foreseen. In order to make the right choice, situational understanding is needed. To

obtain situational understanding, information from different sources needs to be integrated. For example, information from sensors (such as cameras, thermal sensors, but also social media and human observations) and expert knowledge e.g. obtained through experience, needs to be combined to build an understanding of the situation.

An example of a use-case where situational understanding is relevant is recognizing smuggling activities in a busy city. During a military mission in a hostile county, there is a lot of information that might be relevant to the mission, but not immediately linked to smuggling in an analyst's mind. Here, the types of vehicles in the area, the hours of active use of those vehicles, the locations and the people that suspected smugglers meet might play a large part in determining if there is an active smuggling operation going on. Even more general concepts might play a role as well, such as economy and law and order in the area, but also patterns of life, important dates, and the political situation. These factors can be described in flows of people, goods, and information, forming an intricate web of higher-level knowledge. Combining this knowledge in a semantic graph (see Figure 1) can more easily provide new insights to commanders about which information might be relevant to the current situation. For example, a correlation might be found between the detected vehicle type and the time and area of usage, indicating smuggling activities. The values for connections between and correlations of these concepts need to be found or learned in order to make useful predictions.

There are, however, several factors that limit understanding. There may be too little information on some aspects, there may be unreliable or intentionally false information, and there may even be too much information. In those situations, human analysts and decision makers may not be well suited for sifting through the data to find trustworthy and useful information and being able to integrate it into the decision-making process, especially given the time pressure and the potential severity of the consequences.

A possible solution for finding the correlations in the available data is using Artificial Intelligence (AI) approaches. In recent years, developments in neural networks (NNs) have made Machine Learning (ML) a popular method for solving problems in the field of AI. However, for the use-case discussed above, such methods lack a critical property. When making decisions with the help of decision support systems, users require that these systems can explain themselves. They want to be able to ask why the system indicates that something is very likely present in the environment. Black box methods, such as neural networks, are unsatisfactory because they struggle to provide additional insights besides its output. What is needed is a so-called white box approach, which is able to accept and convey knowledge in a for humans understandable way, and can answer 'why' questions.

Another disadvantage of ML is the possibility that the algorithms learn to use prevalent, but unexpected, aspects of the data, which may lead to incorrect decisions. A well-known example of the latter is an ML algorithm that learned to

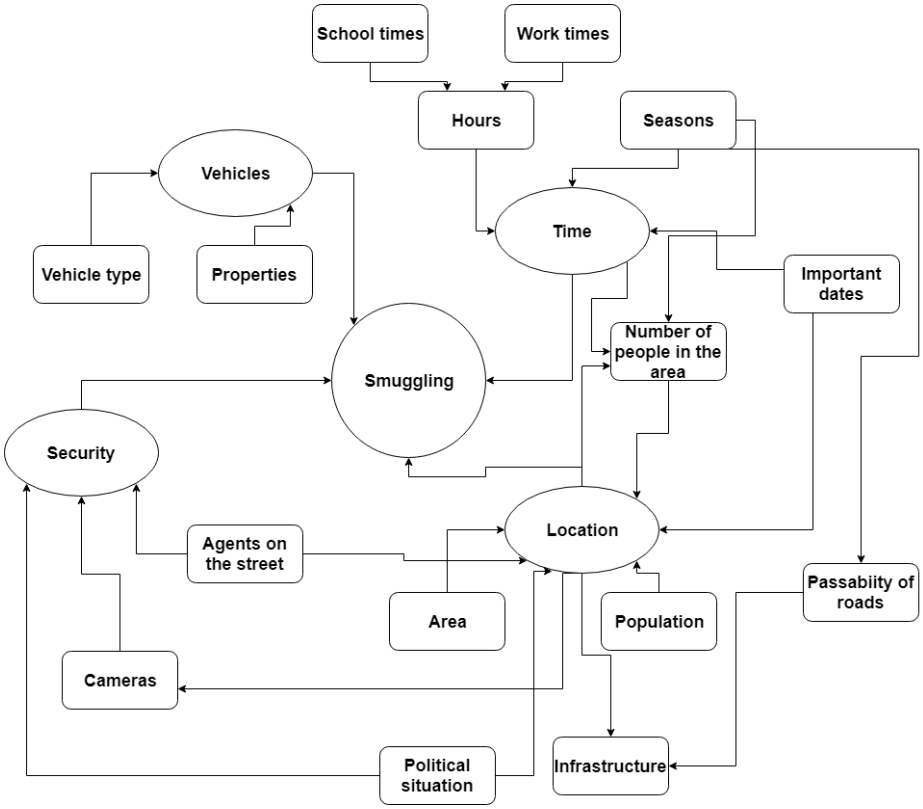

Fig. 1. A simplified semantic graph showing a number of the concepts that can play a role in a smuggling scenario.

classify wolves versus huskies based on the presence of snow instead of learning the discerning characteristics of these animals [11].

Recently, research has partly shifted towards creating AI tools that are able to explain why and how a certain conclusion came to be [13]. Van Lent [8] called this type of artificial intelligence 'Explainable AI'. In a system with explainable AI users may be able to analyze the cause of (faulty) results. Furthermore, by improving the explainability of intelligent software, it has been argued that the users of these types of applications start to trust the systems more [17]. Having trustworthy, meaningful and understandable decision support is critical in the military domain.

In this work, we seek to support analysts with automated tooling. The goal is to be able to understand important aspects of the environment in which a military operation takes place. Often these operations take place in unfamiliar terrain and only a global understanding of the situation is available. The best possible understanding is based on all available knowledge and data. The envisioned support tool integrates multiple sources of information, some of which

consist of data streams, some are symbolic in nature such as the knowledge of human experts. We combine sub-symbolic information and approaches (sensor data, machine learning) with symbolic knowledge and technologies (expert knowledge, semantic networks) to work towards a decision support system that produces better results with fewer data required for learning compared to traditional neural network approaches, and supports explainability.

In this paper we first present a brief overview of related work (section 2, and then discuss the RCN framework and its properties (Section 3). In Section 4 several experiments and their results are discussed. Lastly, in Section 5, our findings are discussed as well as some points of future work.

2 Related work

One of the key challenges of neuro-symbolic model integration is to find suitable ways of merging symbolic and sub-symbolic models into a unified architecture. A good overview of the neuro-symbolic approach is provided by Besold et al. [1]. Our neuro-symbolic approach was partly inspired by Raaijmakers et al. [10], who presented a method for explaining sub-symbolic text mining with ontology-based information. However, this method does not allow a priori expert knowledge to be incorporated in the text mining itself; it is used to explain the mining results afterward. Gupta et al. [4] used an ontology to shape the architecture of the sub-symbolic model, which enables the model to become more explainable since high-level knowledge is embedded in the architecture. The idea to shape the architecture based on a symbolic model is used in the approach discussed in this paper.

Sabour et al. [12] describe a method that comes close to our approach: capsule networks. A capsule is a group of neurons whose activity vector represents the instantiation parameters of a specific type of entity such as an object or an object part. Active capsules at one level make predictions for the instantiation parameters of higher-level capsules. When multiple predictions agree, a higher level capsule becomes active. Sabour et al. show that a discriminatively trained, multi-layer capsule system achieves state-of-the-art performance on MNIST [7] and is considerably better than a convolutional network at recognizing highly overlapping digits. Although this approach was used for image processing and our approach is about concepts, the idea of clusters of neurons signifying one concept with relations to other clusters inspired us and is reflected in how the activation of concepts works in the methods discussed in this paper.

For this proposal, building on the work of Voogd et al. [15], an automated situational understanding support tool is created, which uses a Relational Concept Network (RCN), see section 3.

Explainability in artificial intelligence is a topic that attracts much attention, see e.g Holzinger et al [5] for an overview. A benefit of our approach is that it supports explainability. The expert knowledge that is incorporated in the network makes it possible to explain the output in a for humans understandable way, as it is the symbolic knowledge that humans can comprehend, given an

appropriate user interface. The symbolic knowledge may contain relations that are causal in nature, but this is not guaranteed. Therefore our approach falls short of supporting explanations in terms of causality. But since it does refer to human understandable models, albeit mixed with link-strength based on data, it may be a way to obtain causability as defined in [5]. If in a future situation our approach is implemented as a decision support system, and the decisions together with observations of the results of acting on the decision are fed into the RCN, this could lead to detecting causality [9]. For our application domain, however, it is unfeasible to experiment freely with the decisions to actually obtain causality.

3 RCN Framework

The Relational Concept Network (RCN) consists of a directionally organized network of symbolic concepts (nodes) with relations (edges) connecting them based on expert knowledge. The concepts and their (logical) relations represent the domain knowledge provided by experts, ontologies, or other sources. The (activation) value of a concept is to be interpreted as the probability that this concept is present or relevant in the current situation.

Each concept has an internal predictor method to determine the concept's output activation based on its inputs (where an input comes from either external data or from another concept in the network model). This predictor method can range from a simple rule-based evaluation to a NN that has to be trained. The exact relation values between concepts are difficult to estimate for human. It is for example hard to determine how much the fact that a large car is moving fast would contribute to the chance that it is smuggling. Therefore, we use a sub-symbolic NN as each concept's internal method that, using training data, can learn the influence and strength of the relations, see Figure 2 for a simple example.

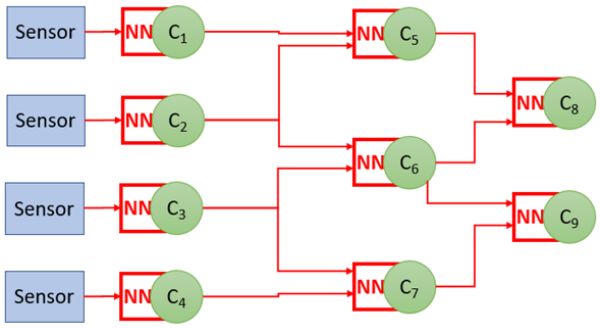

Fig. 2. A simple schematic representation of the RCN where symbolic concepts (C_x) have integrated neural networks.

Each concept is basically a small neural network with one neuron in its output layer: the concept. So, the whole RCN may be seen as one large neural network where some neurons have a concept name. Compared to traditional neural networks the expert knowledge effectively limits the number of links between neurons and therefore limits the state space of the NN, which should result in faster training and/or less need for training data.

A concept which represents a feature of the input data is called an *input concept*. The value is set using an external source, e.g. a sensor or human input (C_1 to C_4 in Figure 2). *Intermediate concepts* receive their input from concepts and provide their output to other concepts (C_5 to C_7). *End concepts* are concepts without dependents, i.e. its output is not being used by other concepts in the network (C_8 and C_9).

The RCN is built as a framework to offer functionalities to create the situational understanding support system. Currently, network-related functionalities are implemented allowing searching for concepts, paths between concepts, dependency analysis, etc. The framework also enables defining and training concepts based on NNs, and querying the (trained) concept outputs. In the following subsections, these two functionalities will be explained further.

3.1 Training the RCN

When training the RCN, first domain knowledge of the expert is implemented in the model through defining concepts and making connections between them. This knowledge may come from different sources. The expert knowledge may come from different experts at different moments. It may also come from an existing RCN from a previous application. It is expected that especially conceptually higher level knowledge is more general and may be re-usable even when applied in different locations and times, such as a previous military mission.

Training the RCN is accomplished by training the integrated NNs separately. A training sample for a single concept consists of a vector of input data (the features) and the associated true value (the label). Because these NN predictors are small and typically have only a few inputs, they can be trained and evaluated quickly. This allows fast feedback on the performance of the proposed connection. A disadvantage of this strategy is that individual predictors can only be trained when there is ground-truth training data available for that concept. If a new connection is made to a concept, only that concept needs to be retrained. If the RCN is extended with a new concept, that concept needs to be trained, but also directly dependent concepts need to be retrained. If an expert changes links between concepts, they may 'break' knowledge that was inserted by another expert. This will sort itself out in the training phase because links that are not supported by the data will receive little weight and therefore have little influence on the outcome when using the RCN.

After training all concepts, the system can be used to provide indications of relevant concepts based on new data in the form of queries.

3.2 Querying the Concepts

After training, the RCN can be used by analysts to examine which concepts are found in the current situation. To do this, data from sensors and other sources are fed into the RCN at the appropriate concepts. These concepts are then used to calculate higher-level concepts. After these calculations, the probability values of all concepts are known. The analyst examines concepts with a high probability value to gain insight into what might be going on.

If only a subset of the input data is available, i.e. some input concepts do not receive data from external sources, thanks to the decoupled nature of the RCN, some intermediate concepts may still be (recursively) inferred, creating a chain of predictions. Furthermore, if an intermediate or end concept has several inputs, these inputs may differ in how much they contribute to the probability value of that concept. The contribution of an input to the output is its feature importance. Inputs with low feature importance may be ignored, making it possible for more high-level concepts to determine their probability value, be it with higher uncertainty (see also "Graceful degradation" in section 4). Permutation Importance [6] is a method to determine the importance of individual features for a black box model. Since the method is model-agnostic it can be used with neural networks in concepts.

Users will - at least at first - be suspicious about the results produced by the RCN and want to be able to ask 'why' questions. The black box nature of a NN is not suited for these questions. The RCN supports explainability in the following way: if input is supplied to a trained RCN and then evaluated through the network of concepts, each concept obtains a probability value. Then, questions such as 'why does this concept have a high probability value?' can be answered by referring to which of the input concepts have a high probability and with what weight (i.e. the feature importance) it contributes to the concept under consideration, all the way back to the first layer of feature data, see Figure 5.

Besides the probability value of the concepts, it is also useful for the analysts to query the performance of the knowledge that was supplied by the experts. After training, a set of validation data is used to determine the performance of the concepts. The analyst can check if there is expert knowledge that does not fit well with the data. If so, the analyst has to determine whether the data is of sufficient quality, or if the knowledge should be enhanced, i.e. remove connections to concepts with a low feature importance, and add connections to concepts that may prove to have a higher importance.

4 Experiments and Results

In this section, we discuss several tests and their results obtained by the RCN to verify the functionality as described in the previous chapter. It is examined how well the RCN can be trained, how modular the resulting model is thanks to its decoupled nature, how well it performs using smaller datasets and what the effect is on explainability.

For the tests described below, we have used the Large-scale Attribute dataset (LAD) [18], which is an annotated image dataset of six different super classes of objects. For these tests, we only used the 'Vehicles' super class, which includes land/air/water and civilian/military vehicles. Each item in the dataset is annotated on several properties ranging from used materials or components to the usage scenario, as well as the subclass it belongs to (e.g. whether it's a car, an ambulance, a train or an airplane). Figure 3 illustrates an instance of the dataset. Note that we only use the annotations as concepts, not the images themselves.

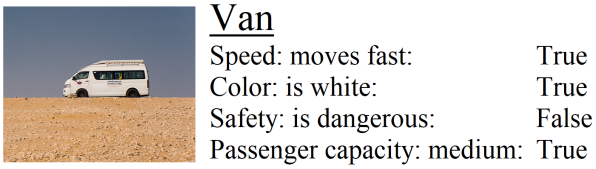

Fig. 3. A sample from the LAD dataset, with some of the many annotated properties.

Thanks to the diversity of the annotated properties in the LAD, it is possible to structure the properties into a hierarchy of more basic concepts (e.g. materials and components) to concepts with a higher abstraction or complexity (e.g. usage scenario, safety).

Combining Expert Knowledge with Data

Current AI techniques are not capable of fully learning both sub-symbolic and symbolic data, therefore the symbolic concepts and their logical relations are defined by an expert. In order to allow the expert to define the RCN by specifying concepts and relations, a graphical user interface (GUI) has been made that allows an expert to pick and place concepts (which are derived based on the labels in the data) and connect them based on expert rules. Note that an expert rule is not an if-then relation, but a combination of concepts that the user/expert has specified as relevant.

In Figure 4 an example is shown for the various vehicle types in the dataset, the resulting multi-layered hierarchy of connected concepts is a sparse network. In this example the user specified (among many other expert rules) that the relations needed for the concept "Function: can dive" are "Shape: is globular", "Shape: is ellipsoidal", "Parts: has a propeller". The RCN is then generated from this expert-made network and the user can start the learning phase. Afterwards, for a given input, the probabilities of higher level concepts and feature importance of the links can be visualized in the same GUI.

Trainability

An obvious requirement is that the system has to be able to learn, i.e. fit the

Fig. 4. The GUI to build the RCN and control the learning phase

structure in the data onto the structure defined by the relations between the concepts. This translates into the training of the sub-symbolic parts integrated into the concepts of the RCN.

As a performance measure, balanced accuracy [2] is used, which relates the average of combined precision and recall obtained on each class, corrected for any class imbalances. This provides a robust performance measure when dealing with varying amounts of data per class. If the network of concepts is close to what is present in the structure of the data, it should result in a well-trained RCN. i.e. a balanced accuracy score between 0.5 and 1.0.

To test this, the available data is split into a training set (80%) and a test set. As a test for this requirement we need to show that the RCN is capable of learning the desired concept relations. In addition, we will compare the performance of the RCN with a single traditional fully-connected NN (FCNN) that uses all concepts as input and provides predictions for all output concepts. The neural networks are trained using a standard multilayer perceptron [3] using 3 hidden layers of 8 hidden nodes each.

The results (see Figure 6 showing the averaged balanced accuracy score over all concepts) indicate that most of the concepts can be trained to a perfect score and the RCN performs well. However, compared to a traditional NN the overall performance of the RCN is lower. There are some possible explanations for these observations. The relations between concepts were chosen based on the concept labels, without looking at the data. Therefore some relations that were defined may not be supported by the data. Another relevant aspect is the fact that the FCNN might learn patterns in the data that can give the correct answer in the current context, but in fact have no connection to the actual (symbolic) relation, similar to the previously mentioned example of the animal classifier that relies on the presence of snow in an image [11]. Lastly, the NN architecture (number

and size of hidden layers) integrated in the concepts was chosen based on a rule of thumb and fixed for all concepts. However, some concepts may benefit from a larger or smaller architecture. Furthermore, when inspecting the dataset, some odd data points were observed. We do not know how the labels are assigned to the data, but there are some assignments that were inconsistent with what is semantically expected for that class. For example, some instances of the concept 'submarine' had no value for the concept 'function: can dive', which should be the case for all submarines.

Explainability

The RCN should allow a user to find out why a concept is indicated as strongly present in the data. To test this form of explainability, the RCN is first trained, then one single data point is applied to the input of the RCN and all probability values for the concepts are determined. Then a textual or graphical representation of how concepts are influenced by 'lower' concepts can be made. This can be examined by the user to check whether it makes sense.

In Figure 5, a textual example of explainability is shown for a specific concept ("aim: is for military"), the dependencies on lower level concepts and their strength are shown.

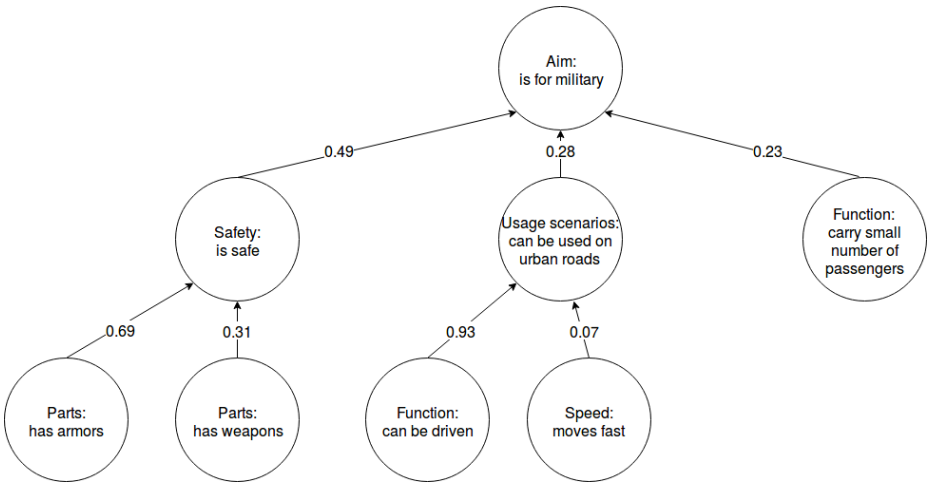

Fig. 5. The hierarchical structure of a concept, its sub-concepts and the feature importance values of those concepts that provide input for a higher-level concept.

Modularity

One of the advantages of the RCN compared to a NN is that it does not require all inputs and corresponding labels to be available at one time. The RCN can

be neatly decomposed in concepts with their input concepts up to the point for which data is available. It can thus be trained piece by piece.

To make sure this works, our current implementation allows that concepts can be trained independently (or in sets). Only those concepts for which there is data in the training set are trained. When data for a concept becomes available at a later date, that concept can then trained, keeping the rest of the RCN (and its trained concepts) in tact. Note that this is not the same as true incremental learning of the RCN where the sub-symbolic parts of the concepts get to learn step-by-step in a more or less random order.

Improve Prediction Performance with Small Training Sets

The use of expert knowledge in the RCN effectively reduces the phase space available compared to the use of a traditional NN. For small training datasets, and if the expert knowledge fits this data well, this should result in a higher performance after training than for a traditional NN.

To test this, smaller and smaller training datasets are used to train the three different configurations have been tested:

- $FCNN_{full}$: A fully Connected NN that uses all input and intermediate concepts of the RCN as input features,
- FNN_{inputs}: A fully connected NN that uses the same input concepts as the RCN, but does not use the intermediate concepts,
- RCN_{inputs}: The RCN; for performance measurements only input concepts are used, not the intermediate concepts.

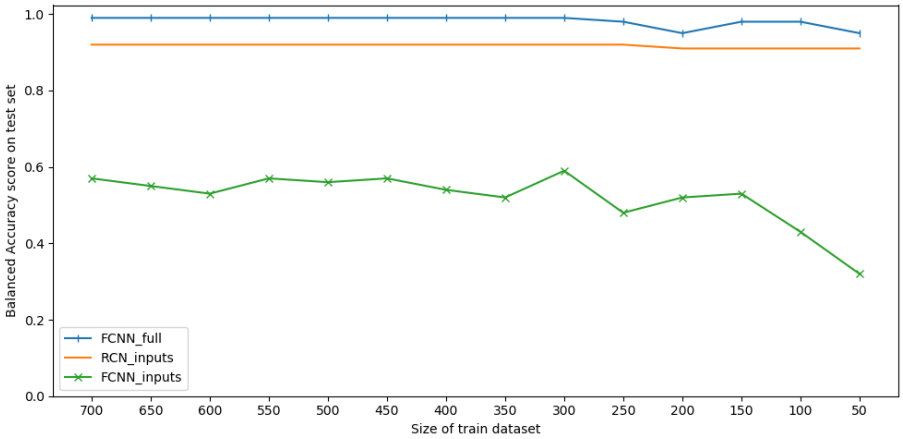

Fig. 6. The performance of different methods trained on the dataset, either using traditional fully connected NNs, or the RCN.

The results (see Figure 6) show that $FCNN_{full}$ scores a near-perfect classification on most of the sizes. There does seem to be slight drop in performance for smaller datasets, this small drop might indicate a sensitivity to the data present. However, as this drop is not clearly observed in the performance of the other algorithms, it might be the case that it is not the data itself that causes this decrease in balanced accuracy, but an effect of the small size of the dataset.

The $FCNN_{inputs}$ has a much lower score with a strongly decreasing score for smaller datasets. The RCN_{inputs} shows a slightly lower performance than $FCNN_{full}$, but its performance remains more stable at smaller datasets.

The scores of the RCN_{inputs} compared to $FCNN_{full}$ is lower because the $FCNN_{full}$ are free to include subtle relations that are not included in the expert rules. Additionally some expert rules are poorly chosen compared to the data used. The instability of both the FCNNs performances at small dataset sizes indicate that the expert rules in the RCN_{inputs} are meaningful for stability.

Graceful Degradation

In the previous paragraph the size of the training dataset was decreased, but each entry in the dataset still had values for all properties, i.e. all input and intermediate concepts were still present in each data point. In practice it may often be the case that for only some of the input concepts data is available. Where traditional NNs can only function if all input data is available, the RCN allows for some concepts to still be evaluated even if only a subset of input data is available.

A function can be defined purely based on the connections between the concepts in the RCN that returns all concepts that can be evaluated for a given subset of input data. This reachability function is evaluated by averaging over the many possible configurations of available input for the RCN. Two values are calculated: the percentage of output concepts that are reachable and the percentage of intermediate concepts that are reachable.

In Figure 7, the bottom graph (purple) shows that for traditional NN all input must be available. The percentage of output concepts that are reachable as a function of the percentage of available input concepts is the blue (middle) line, if the reachable intermediate concepts are included the red line (top) is observed. The higher the line the more concepts can evaluate their probability and become useful to an analyst.

After training it becomes clear that the feature importance can differ significantly, see Figure 5. This opens the possibility to disregard connections with a low feature importance, making the network more sparse and resulting in an even higher reachability for a given percentage of available input. The drawback, however, is an increased error in the output.

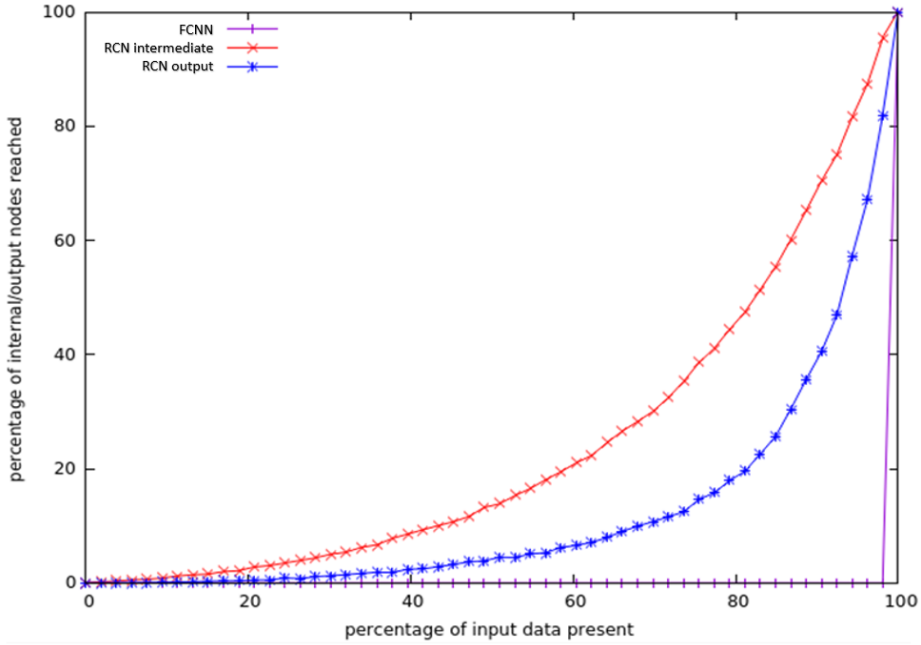

Fig. 7. The degradation of reachable concept outputs when decreasing the completeness of an input feature, shown for the traditional fully connected NNs, and the RCN.

5 Discussion and Conclusion

In the previous section, some of the properties that the combination of symbolic knowledge and sub-symbolic methods should have, are examined to determine if this can lead to a fruitful pairing for the situational understanding support use-case. Here we discuss the results and give directions for future work. We end with some conclusions.

In our current implementation, the human experts are the only source of symbolic knowledge in the form of expert rules, they interpret the results and make corrections if necessary, and finally provide their analysis results to a decision maker. Experiments with the interface show that it is easy to choose concepts and make links between them to implement expert rules and build a hierarchy of concepts.

In standard expert systems where symbolic rules can be entered and used, the maintenance of the collection of rules increases drastically [14,16]. In the RCN, adding or changing the expert rules does not lead to similar inconsistencies as can occur with expert systems because wrong connections will, during training, obtain a low weight. The neural network structure also allows for working with incomplete expert knowledge.

The tests on trainability show that the RCN is capable of training the sub-symbolic parts integrated in the concepts to match the structure in the data.

It was found that some concepts have a high performance and some a lower performance. The concepts that work well, implement expert rules that match well with the data. Low performing concepts are a signal for the user to examine why this is the case. There may be several reasons: they represent a wrong expert rule, the data is incomplete or has errors, or the structure in the data is too complex for the current rule given the size of the sub-symbolic part. If the last reason is the case, the sub-symbolic part can be increased, which would also require more data for training, or additional expert rules are required: more or different connections between concepts are necessary, and/or more concepts with accompanying expert rules need to be introduced.

Modularity is currently supported by collecting all relevant data at the concepts and then do a learning step for separate concepts. After training, the RCN can be used for inferring probabilities of higher level concepts in input data. During use, modularity is also supported, as the tests on graceful degradation show that it is possible to evaluate the probabilities for subsets of concepts, depending on the available input data and the connections from the expert rules. In real-world applications, it is more than likely that input data is only available for small sections of the RCN.

It is, however, desirable to have a continuous learning approach where incoming data with information on several hierarchical levels of the RCN can be used for learning. In our current implementation, we use the scikit-learn toolkit which supports incremental learning but this has not been tested yet.

The tests on performance as a function of training data set size indicate that the RCN's performance is less dependent on the amount of available training data compared to a NN. This is an important finding from the perspective of the use-case since it can be expected that at least for some concepts only little data will be available, making the use of a traditional NN infeasible. On the current dataset with imperfect expert rules, the RCN has a relatively high performance, close to that of a traditional FCNN trained on the same data, while providing the advantages of explainability and more flexibility in training data.

Although the basis for a decision support system that incorporates both symbolic and sub-symbolic data has been shown in the RCN approach, there are many things that can be enhanced. Hence, there are several plans for future implementations.

Currently, human experts are the only source of symbolic knowledge. This can be expanded by for example the use of ontologies that provide relevant domain knowledge. Often ontologies use well-defined relations between concepts. A way needs to be found to implement these in the RCN, and it needs to be studied how these can benefit a user by giving more insightful explainability. If implemented, the expert can also use this richer type of relations when specifying rules.

An additional source of symbolic knowledge may come from lessons learned from previous military operations that can be entered in the form of expert rules. It may even be possible to re-use the RCN as a whole. It is expected, however, that lower level concepts may require significant retraining. Higher level concepts

may be more general in nature and therefore transferable from one situation to the next.

If an expert rule performs badly while using good data, it needs correcting. This can be done by a kind of hypothesis testing: changing the expert rule, checking its performance, changing it again, etc. until a good performance has been obtained. This can partly be automated by having the RCN check many different combinations of connections to a concept and suggesting the best performing set to the expert. Additionally, the predictor, here a NN with a number of hidden layers and nodes in each layer, can be changed to better suit the complexity of the mapping it needs to learn.

Another element that can be improved in the fact that the current network is a directed graph: the relation between nodes only works one way. This limits the rules that can be entered. For many concepts in a domain, it may not be clear what causes what, only that they are even related. This requires bidirectional connections. A potential problem arises when anything can be connected to anything: the output of a concept can indirectly become its own input. This will have to be solved, for example by freezing activations or repeatedly resampling.

One important aspect that is currently missing in the RCN is an indication of the uncertainty in the output. Uncertainty should be available integrally from lowest level input to the output of the highest level concepts. This will be important for analysts to consider when building their situational understanding. If implemented, this may be used as a means of graceful degradation: if a concept has inputs with low feature importance for which no data is available, they may be ignored. This will, however, increase the uncertainty of the output, which should be visible to the user.

All in all, the methods discussed in this paper show that combining symbolic knowledge (from experts and expert systems) with sub-symbolic methods (such as neural networks and sensor data), can lead to a fruitful pairing. The combination of the flexibility of neural networks with the domain knowledge of experts results in a form of hybrid AI, where symbolic insights from an expert can be made more precise by training sub-symbolic networks using data. The main advantages are the flexibility this approach offers, explainability of the results, and a reduction of data requirements.

References

1. Besold, T.R., Garcez, A.d., Bader, S., Bowman, H., Domingos, P., Hitzler, P., Kühnberger, K.U., Lamb, L.C., Lowd, D., Lima, P.M.V., et al.: Neural-symbolic learning and reasoning: A survey and interpretation. arXiv preprint arXiv:1711.03902 (2017)
2. Brodersen, K.H., Ong, C.S., Stephan, K.E., Buhmann, J.M.: The balanced accuracy and its posterior distribution. In: 2010 20th International Conference on Pattern Recognition. pp. 3121–3124. IEEE (2010)
3. Gardner, M.W., Dorling, S.: Artificial neural networks (the multilayer perceptron)a review of applications in the atmospheric sciences. Atmospheric environment **32**(14-15), 2627–2636 (1998)

4. Gupta, U., Chaudhury, S.: Deep transfer learning with ontology for image classification. Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG), 2015 Fifth National Conference on. IEEE (2015)
5. Holzinger, A., L.G.D.H.Z.K.M.H.: Causability and explainability of ai in medicine. WIREs Data Mining Knowl Discov. e1312, doi:10.1002/widm.1312 (2019)
6. Korobov, M., L.K.: Permutation importance (2017), https://eli5.readthedocs.io/en/latest/blackbox/permutation_importance.html
7. LeCun, Cortes, B.: The mnist databasse of handwritten digits. http://yann.lecun.com/exdb/mnist/ (2010)
8. van Lent, M., F.W.M.M.: An explainable artificial intelligence system for small-unit tactical behavior. IAAI EMERGING APPLICATIONS (2004)
9. Pearl, J., M.D.: The book of why. New York, NY: Basic Books (2018)
10. Raaijmakers, B.: Exploiting ontologies for deep learning: a case for sentiment mining. Elsevier, Procedia Computer Science 00 (2018) 000000, SEMANTiCS 2018 14th International Conference on Semantic Systems (2018)
11. Ribeiro, M.T., Singh, S., Guestrin, C.: Why should i trust you?: Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. pp. 1135–1144. ACM (2016)
12. Sabour, Frosst, H.: Dynamic routing between capsules. 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA. (2017)
13. Samek, W., Wiegand, T., Müller, K.R.: Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. ITU Journal: ICT Discoveries, Special Issue No. 1 (2017)
14. Vargas, J.E., Raj, S.: Developing maintainable expert systems using case-based reasoning. Expert Systems **10**(4), 219–225 (1993)
15. Voogd, Hanckmann, d.H.v.L.: Neuro-symbolic modelling for operational decision support. NATO MSG-159 Symposium, STO-MP-MSG-159.3 (2018)
16. Wagner, W.P., Otto, J., Chung, Q.: Knowledge acquisition for expert systems in accounting and financial problem domains. Knowledge-Based Systems **15**(8), 439–447 (2002)
17. Wang, N., Pynadath, D.V., Hill, S.G.: Trust calibration within a human-robot team: Comparing automatically generated explanations. In: The Eleventh ACM/IEEE International Conference on Human Robot Interaction. pp. 109–116. IEEE Press (2016)
18. Zhao, B., Fu, Y., Liang, R., Wu, J., Wang, Y., Wang, Y.: A large-scale attribute dataset for zero-shot learning. CoRR abs/1804.04314, arXiv:1804.04314 (2018)