**Knowledge based artificial neural network (KBANN) Algorithm**

**Abstract**

Hybrid learning is an area where theoretical knowledge of a domain and a set of related training examples are used to train the system for more accurate classification and better generalization. This kind of system works better because it uses two source of information which helps to cover the missing information in one source from other. In this way, system learns better than other systems which work with only one source of information. In this work, KBANN is one such an algorithm described in detail along with an example, which follows hybrid learning approach, it map domain theories to neural network and refine it using a learning algorithm.

**Introduction:**

Suppose, you are trying to teach someone to recognize a class of objects which it has never seen before then you may have two approaches as mentioned below:

1. Hand-built classifiers
2. Empirical learning

Hand-built classifiers: This teaching mechanism basically uses domain theory to teach about something rather than giving an extensive set of examples. In simple words, it's learning by being told.

Flaws: These methods have certain lacking like these systems are non-learning systems and assume that their domain knowledge is complete. But in real-world problems completeness and correctness are big challenges. Apart from this, writing so many rules for the domain theory makes the system slow. Also, modifications of rule are difficult.

Empirical learning: This teaching approach focus on providing a lot of examples to a person without giving a explanations of why these example or the other belongs to this class member.

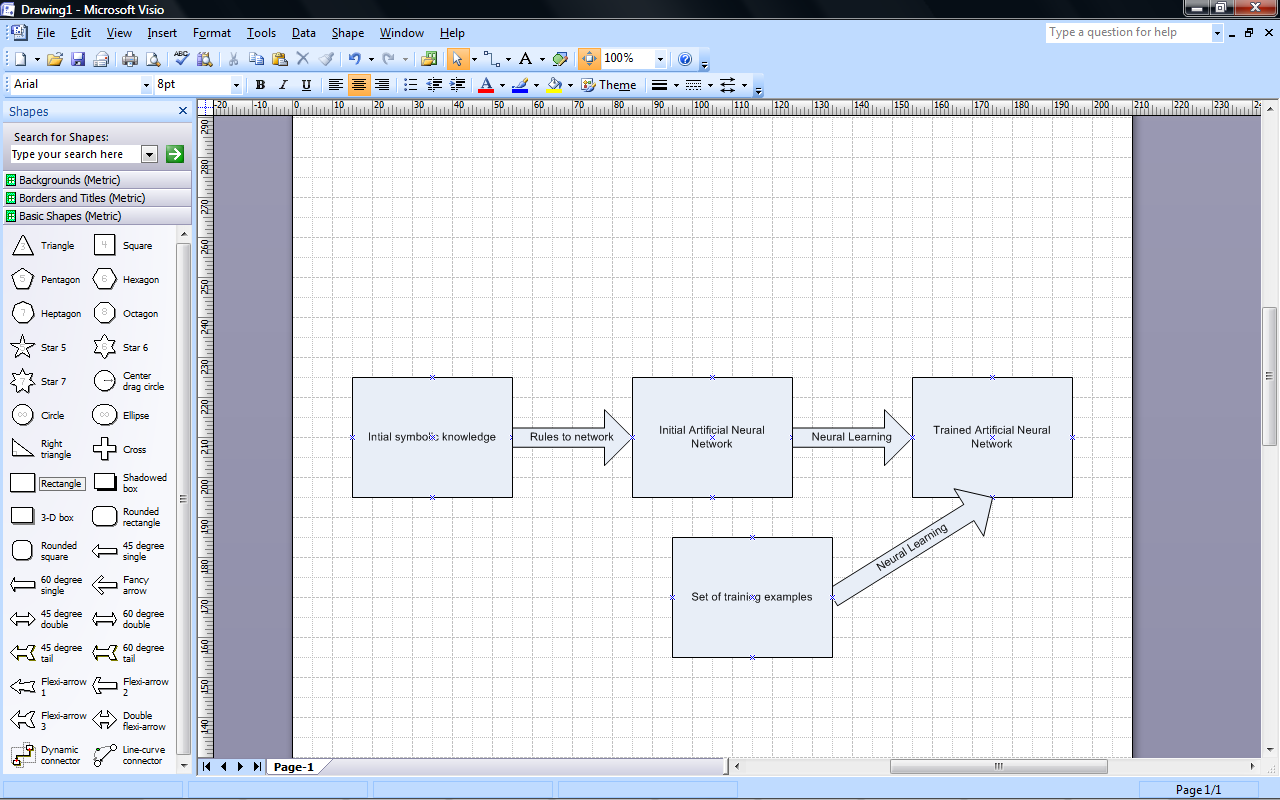
Flaws: Right and relevant feature selection based on the context is an important criteria for describing an object. In-appropriate feature selection may lead towards unknown description on an object. Complex features simplify learning but they are difficult, prone to error. While training, exceptional set of examples are not properly represented which results difficulty in handling uncommon cases.

Both these above mentioned teaching methods have flaws which do not encourage these methods to be generally applicable to learning tasks so some kind of hybrid system is needed which consist of both hand-built classifiers and empirical learning where a person learn both from theoretical and examples. One of such system is Knowledge based artificial neural network (KBANN), it is the successor of EBL-ANN algorithm.

**KBANN algorithm:**

Brief idea: As presented using the figure 1, in this algorithm symbolic rules or hand-built classifiers are inserted into a neural network. Then the network is more tuned using standard learning algorithm like (back propagation) using various training examples. This more tuned or refined neural network works with better accurate classifier. This algorithm is nothing but a hybrid approach.

Below is a block diagram of knowledge based artificial neural network algorithm.

**Fig. 1**: Flow diagram for KBANN algorithm[1]

In this algorithm, feed forward neural network is used with logistic activation function.

As presented in figure 1, there are majorly two components/algorithm in the KBANN i.e.

1. rules-to-network translator
2. a refiner

**🡪Rules-to-network algorithm:** The objective of this algorithm is to map the rules to the neural network. This rule translation simplifies the learning problem by specifying the important “derived’ features.

Below are the mappings:

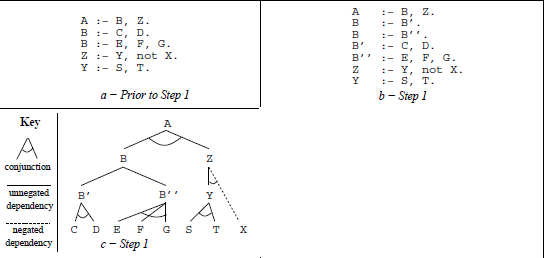
* Final conclusion 🡪 Output units
* Supporting facts 🡪 Input units
* Intermediate conclusion 🡪 Hidden units
* Dependencies 🡪 Weighted connections

The rules which need to be translated to KBANN neural networks are horn clauses having constraints like the rules must be propositional, acyclic and hierarchically structured.

Rule-to-network algorithm steps:

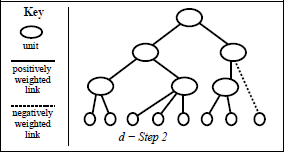
Step 1. Rewrite rules so that disjuncts are expressed as a set of rules that each have only one antecedent. Here in figure 3, dotted line represents the negated antecedents and solid line un-negated antecedents. Antecedents are connected by arcs which represents cojuncts i.e. AND/OR trees.

For example: A :- E, F, G can be writing as 🡪 (A :- A`), (A` 🡪 E, F, G)



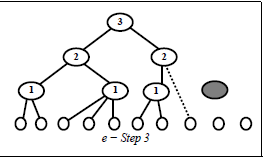
**Fig. 2** Step 1[1]

Step 2. Map the rule hierarchical structure directly to the neural network.



**Fig. 3** Step 2[1]

Step 3. Figure 3 represents labelling the units to their “level.”



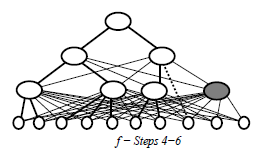
**Fig. 3** Step 3 [1]

Step 4. Add hidden units to learn derived features not specified in the initial rule or specific instruction from a user (*optional)*.

Step 5. Add units for known input features that are not referenced in the rules.

Step 6. Add links between all units that are separated by a level.

Step 7. Perturb the network by adding near-zero random numbers to all link weights and biases.



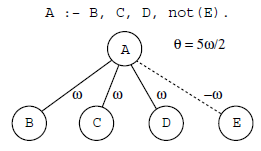
**Fig. 4** Step 4-5-6[1]

Rule-to-network translator sets the weight & bias of units for significant activation (near one or near zero) based on the deduction on domain knowledge.

* For positive antecedents +w
* For negative antecedents –w
* Bias as (P-1/2)\*w, where P is the no. +ve antecedents on a rule.

In the example mentioned in figure 5, the +ve antecedents have weights set as **w** and –ve antecedents sets the weight as **–w** and bias is set to (3-1/2)\*w = **5w/2**. If B, C, D has activation approximately equal to one and E approximately equal to zero then unit A will get activated since the threshold for activation of A unit is 5w/2 and we get a total of 3w. So unit encoding conjunction rules will get active when all +ve weighted links carry a signal near 1 and all –ve weighted links get the signal near zero.

In case of disjunctive rules, first we need to rewrite the rules as multiple rules with the same consequent and second the weights need to be set as w. Also, the bias need to be set as (½)w.



**Fig. 5** Conjunctive to neural network translation[1]

**🡪A refiner algorithm:**  In this part, the neural network is trained/refined using examples and learning algorithms like back propagation. After this refinement, the trained network model works more accurate than other machine learning algorithms.

Using back propagation algorithm for refinement of KBANN neural network creates some issue as they start with confident answers i.e. output units have either activation near zero or one. In this case, the changes to network will be very less regardless of the correctness of the answer which makes very difficult for a neural network to correct the aspects which cause the errors. KBANN uses cross entropy error function as mentioned below:



Along with error function, another term called regularization (penalizing the network for making changes in the original domain theory) is added. It tries to encourage the network to make the original weights remain unchanged. Below is the equation for regularizer:



Here  controls the ability of network to learn the training set and distance from the initial rules.

**KBANN example:** Let us take an example to understand the working of KBANN algorithm.

|  |  |
| --- | --- |
| **KBANN(domain theory, training examples)** | |
| In this example, domain theory is given as set of rules and various training examples are given to classify the object as cup and non-cups. In BottomlsFlat feature, there are 4 ‘x’ in cups and 4 ‘x’ is non-cups then it will be considered as Cups because for this attribute under cups category there 4 out of 4 ‘x’ present and in non-cups category, there are 4 out of 6 ‘x’ present.  Step1: Rewriting of rule is not needed as there is disjuncts present in the set of rules. Rewriting is done in such a way to form the hierarchical structure so that rule can be easily transformed to neural network.  Step2: In figure 7, rules are mapped  to neural network. For example, Cup node is connected to Stable node, Liftable node and OpenVessel. Stable node is connected to BottomlsFlat input feature and so on.. | **Fig 6**. Showing domain theory and training example[3] |
| Step3: Numbering can be done for units by their level, if needed.  Step4: Hidden units are not required in this example so not added.  Step5: Various input features are shown in figure 7 taken from training example (like Light, Expensive, HasHandle etc..). It is important to add these input features to correctly learning the concept as rule are not perfectly correct.  Step6: Links are added to the nodes at level n-1 to n levels shown in fig.7  Steps7: Weights are given as ‘+w’ to positive antecedents and ‘-w’ to negative antecedents and bias is given as (P-1/2)w. In figure 7, various types of line shows different weight configuration in neural network. | **Fig 7.** Showing build neural network using KBANN[3] |

**Applications area:**

KBANN has been applied in many areas such as it used in predicting the properties of alpha + beta titanium alloys [2]. In this paper, author has used knowledge based artificial neural network to predict tensile strength, yield strength, beta transus, specific heat capacity and density of titanium alloys. The neural network modelled has 10 nodes in one hidden layer and sigmoid function is used. It has also been used in medical area as a classifier for pulmonary embolism diagnosis. In this work, author has compared KBANN with other emperical learning algorithms such naive Bayes, Bayesian Belief Network, C4.5 decision tree algorithm, multilayer perceptron, boosting & bagging. KBANN found to perform better[4]. In the paper [5], the author has used KBANN algorithm for microwave components.

**References:**

1. Geoffrey G. Towell, Jude W. Shavlik,"Knowledge-Based Artificial Neural Networks", Appears in Artificial Intelligence, volume 69 or 70.Submitted 1/92, Final pre-publication revisions 8/94
2. P. S. Noori Banu\* and S. Devaki Rani, "Knowledge-based artificial neural network model to predict the properties of alpha+ beta titanium alloys", Journal of Mechanical Science and Technology 30 (8) (2016) 3625~3631
3. <http://jmvidal.cse.sc.edu/talks/indanalytical/kbannexample.xml?style=White>
4. G. Serpen∗, D.K. Tekkedil, M. Orra,"Aknowledge-based artificial neural network classifier for pulmonary embolism diagnosis", Elsevier Computers in Biology and Medicine 38 (2008) 204 – 220
5. P.M. Watson, K.C. Gupta, and R.L. Mahajan,"Development of Knowledge Based Artificial Neural Network Models for Microwave Components", IEEE MTT-S Digest, 1998