INTERNATIONAL COMPUTER SCIENCE INSTITUTE

1947 Center Street • Suite 600 • Berkeley, California 94704 • 1-510-642-4274 • FAX 1-510-643-7684

An application of a neural net for fuzzy abductive reasoning

Matthias Kaiser

TR-93-044

August 1993

Abstract

This is a description of a simple system that is able of performing abductive reasoning over fuzzy data using a back-propagation neural net for the hypothesis generation process.

I will first outline and exemplify the notion of abduction as a process of building hypotheses on the basis of a given set of data, evaluating them to find the best hypothesis and give explanation for the made selection. I extend this notion to account for abductive reasoning over fuzzy data. As an example I describe the classification of objects according to fuzzy sensory features into previously learned categories that were represented by a set of objects described by feature-value-pairs from which prototypes are detected which form the center of a category.

In the following a brief description of the back-propagation algorithm and a design of a demonstration system that is capable of carrying out abductive reasoning in a small example domain is given. The system is able to learn to classify kinds of fruit given certain feature-value-pairs and to detect the most prototypical feature-value-pair-clusters within a category. The trained neural net is used for the hypothesis generation process. It also provides very critical information for the evaluation and explanation of hypotheses. I then discuss the implementation of an evaluation and explanation component using the specific capabilities of the neural net.

1 Introduction

A very important kind of reasoning is the generation of hypotheses from certain data we are presented either through our perceptions or through communication. One recognizes, for example, objects according to shape,colour, size etc. From these data one makes a guess, and what is more, a hypothesis,what these data represent. To do this we rely on what we know of the world, we compare a set of features we perceive to sets of features of which we know to be premises for the hypothesis of certain objects. Such a hypothesis generation together with an evaluation of how good the hypothesis is (i.e.,how certain we are about it) and the generation of an explanation why we made this and not another hypothesis is called abductive reasoning. In this paper I want to model the process of abductive reasoning using a neural network, a so-called back-propagation network. The reason for the use of a neural network for a task like abductive reasoning is as follows:

Abductive reasoning relys on data sets where the data may vary in their values or even if they are present at all or not For example, not every ball has the same size, not every T-shirt the same colour, and we can recognize a bird even though it may not be able to fly. Similarly certain symptoms may or may not be present in a person having a certain type of disease. Neural networks have proven very successful in dealing with non-precise or fuzzy data. They not only can learn from examples to classify data sets but also evaluate the degree of membership (prototypicality) of a data set in respect to a given class.

In making hypotheses we do not often have a predefined algorithm for the generation of a hypothesis because the generation may depend critically on which data or data values are important for the hypothesis, for instance, the colour of a T-shirt is not important to identify it as a T-shirt, the colour of an animal is important to recognize it as say a tiger or a panther.Algorithms for hypothesis generation are sometimes hard to find, if at all, the ability to learn by examples, as neural networks can do is a good presupposition in order for a machine to make certain hypotheses. Neural networks found useful in the solution of problems requiring recognition of complex patterns and performing non-trivial mappings they learned by examples, are back-propagation networks.

In this paper I design a simple abduction system that can hypothesize a certain type of fruit given a certain set of feature-value-pairs on the basis of learned examples from which it formed prototypical knowledge. The values of the features may be fuzzy. Furthermore, the system will be able to evaluate generated hypotheses according to how typical the hypothesized object would be for a class (in my example a kind of fruit)and how far the made hypothesis is better as compared to alternatives.

From this the following disposition of the paper emerges: I first give an explanation and examplification of the concept of abduction and important implications. A critical role in the system will be prototypicality of data as well hypotheses. Thus we will be able to get information about the certainty of hypotheses on the basis of a

system's knowledge it aquired previously. I will describe prototypicality using fuzzy descriptions enhancing the concept of abduction to abduction in fuzzy domains.

I then turn to a short outline of the essential ideas of the back-propagation network. The major part of the paper is dedicated to the description and discussion of the abduction system and features I find interesting enough for further illumination. The following points constitute the disposition of the paper.

section 2: Abductive reasoning.
section 3: Description of the abduction model using back-propagation network.
section 4: Discussion.
section 5: Conclusion.

2 Abductive Reasoning

2.1 Abduction

Abduction is an inference from a set of given data describing something to an explanatory hypothesis that is considered to be the best explanation for the given data. One could say it is the best guess what the data stand for. Thus abduction can be seen as a kind of theory forming or interpretive inference (Harman, 1965). To form a theory about whether the sun turns round the earth or vice versa we have to proceed from the data (how planets move in respect to each other) to the inference which is the hypothesis that best explains these movements. Abduction plays a major role as an inference in science and in daily life since we perceive data constantly and seek hypotheses for what they are represent. Incidentally, abduction is the inference performed in detective stories where the genius of detection can find hypotheses explaining mysterious data, odd behaviour and actions of people. For examples see the books by Conan Doyle who,mistakenly confuses abduction with deduction.

The process of abduction can be characterized by the following pattern:

- D is a collection of data (facts, observations, givens)
- h is a hypothesis that explains D (would explain D if h is true)
- No other h is able to explain D as well as this h does
- Therefore h is probably true

Abduction is an inference that infers causes from effects. This is the reason why abduction is sometimes characterized as modus ponens turned backward (Charniak, McDermott , 1985).

The idea of abduction is that a body of data provides evidence for a hypothesis that satisfactorally explains the data more plausible. One can divide the process of abduction into three components:

a. Hypothesis generation: Given the data set D a hypothesis is generated as to what D represents.

b. Critical evaluation of hypothesis alternatives: This is if more than one hypothesis is possible to account for D as characterizes; then they are evaluated as to which accounts best for D.

c. Acceptance and explanation of the best hypothesis: This is an explanation why a chosen hypothesis can be accepted and how and why it best explains D and in as much D is sufficient for the inference of h.

Sometimes only phase 1 stated under a) has been referred to as abduction. Although the explanatory hypothesis can be simple, it is often the case that we have to deal with composite multi part hypotheses,as in theory forming,where there are mostly multiple parts being related to each other in various ways.But there is a problem with this view. The problem is that no feasible information processing strategy can afford to explicitly consider all possible combinations of potentially usable theory parts since the number of combinations would grow exponentially. To avoid such an explosion we must adopt a strategy which avoids generating all possible explanations by preselecting theory fragments to remove those that are implausible under the circumstances. My strategy combines a clean separation between the process of hypothesis generation and that of acceptance because it mixes degrees of critical evaluation into the process of the hypothesis generation. Thus phase 1 to 3 cannot be separated completely because they depend on and influence each other (see Josephson, 1993).

It is now time to give an example for a very simple kind of abduction. In fact, I will deal with very simple examples, since my goal is not to explore the notion of abduction to a large degree, but rather to use a certain device, a neural network to do abductive reasoning. Showing the specific advantages of neural networks I can utilize them for hypothesis generation, evaluation and explanation of data.

Imagine that one wants to hypothesize which kind of fruit is described by the following features (the data) that are preselected for the hypothesis generation process:

1. is_red(fruit) is_sweet(fruit) is_small(fruit) has_pit(fruit)

Given this set of data we can hypothesize that a cherry is the fruit being described by them.

Can we accept this hypothesis? To decide this we must ask which alternatives are possible. I know of no good alternatives, and I think I have sufficient data for this hypothesis, which if given, could make a difference, so I accept the cherry as hypothesis from our given feature set. We evaluate in that we ask which alternatives could exist given the above features. Perhaps a strawberry? But a strawberry has no pit, thus the hypothesis does not account for this feature. We can explain our hypothesis by comparing the feature set with that what we have learned about the cherry: this fruit is red, and as far as I know a cherry is red (supporting the hypothesis). This fruit has a pit, and as far as I know a cherry has a pit (supporting the hypothesis)....

In accepting the hypothesis we have to ask: Are the data on which we built our hypothesis sufficient for a certain hypothesis? This concerns the selection of data for the hypothesis. So we selected the feature "sweet" but not the feature "contains water" which also holds for cherries but is not relevant to distinguish it from other kinds of fruit. I will have not much to say about data selection in this paper,but this is a very important question . The opposite question of the sufficiency also plays a role, namely: Is the set of data actually relevant? The consequence of having to much data that are not relevant is waste of computation time. I probably would still be able to hypothesize a cherry when the fact is_sweet(fruit) is not given. Therefore the hypothesis remains the same despite of a varying set of data. On the other hand, the lack of one feature may make a difference in the hypothesis. For example:

2. $\dfrac{\text{is\_yellow(fruit) is\_sweet(fruit) is\_elongated(fruit)}}{\text{banana}}$

3. $\dfrac{\text{is\_yellow(fruit) is\_sweet(fruit) is\_elongated(fruit) has\_seeds(fruit)}}{\text{pear}}$

(Please neglect for the sake of the example that a banana is more elongated than a pear. I address fuzzy data shortly.)

If a fruit has_seeds(fruit) is missing we would rather hypothesize a banana than a pear. Here we see how critical the preselection process for hypothesis generation, even as in such a small example, may be. A smart hypothesis generator could ask for more information if it knows that it gets preselected information, and if it knows that a certain information could make a difference if provided.

Another example is presented in 4. and 5.:

4. $\dfrac{\text{is\_yellow(fruit) is\_sweet(fruit) has\_seeds(fruit)}}{\text{apple}}$

5. $\dfrac{\text{is\_red(fruit) is\_sweet(fruit) has\_seeds(fruit)}}{\text{apple}}$

Even though the data sets contain a different colour we hypothesize that an apple is described because we know red apples and yellow apples. But, if we get the description:

6. is_orange(fruit), is_sweet(fruit), has_seeds(fruit)

we would rather hypothesize that what is described is an orange and not an apple, because orange apples are, at least, atypical.

What we see from the above examples is:

a. given a set of data we can make a certain hypothesis of what is described by them

b. the absence or presence of a certain datum may or may not affect the hypothesis

c. the variation of a datum belonging to the same category (e.g. colours) may or may not influence the hypothesis

From this we can see that it is a difficult task, and might not be "fruitful" at all for some cases, to formulate explicit rules that would constitute a good hypothesis generator relying on the traditional AI paradigms.

The abduction system I introduce in this paper will be able to make hypotheses on the basis of learned prototypical examples consisting of a set of prototypical data (as premises) and a hypothesis that is treated as prototypical hypothesis emerging from the given premises. What will be actually measured providing evidence for one hypothesis rather than for another is to what degree the data given supply a hypothesis i.e., to which prototypical hypothesis the network has learned it, comes closest, and how big the difference is between a hypothesis and an alternative. However, there might be more than one exemplar in a category e.g. yellow and red apples which may be equally prototypical, so the system will have to learn not only one exemplar as "the prototype" but it will have to know more exemplars. Also, it might learn non-prototypical exemplars as explicit borderline cases of a category. This can support a good explanation of hypotheses, as will be discussed later.

It is important to stress that the process of abduction is an ampliative i.e., at the end of such a process,after we found a best hypothesis,we may have more information than we had before. In the abduction process, the information of the premises is transcended and new information is generated. In deduction, we have a contrasting case to this: here we extract information in form of the conclusion which was already contained in the premises. To put it in slogan-like words one could say: "Deductions are truth preserving, abductions are truth producing". Another important feature of abduction as compared to deduction is that the conclusion of an abduction can have more certainty than any of its premises. A deduction, on the other hand, is never stronger than its weakest link. For example, if an apple looks like an apple, but is unusually sour, we nevertheless tend to call this sour object an apple if it has all the other features of an apple rather than calling it a lemon. The fact that makes abduction an attractive concept to model using a neural network is that an abduction emerges from premises; however, no single premise is essential to recognizing/constructing the hypothesis. This is the major contrast to more traditional kinds of reasoning that are based on deduction. Now I turn to the explanation of some notions that are critical ideas standing behind the philosophy and the design of the system that extend the notion of abduction presented so far. This is the permission of fuzzy data as inputs and the production of fuzzy hypotheses as outputs of the hypothesis generator,the neural net.

To build a system as outlined above to generate hypotheses from given data I must first address which data are needed to make certain hypotheses in a given domain. My example domain will be a part of the category of fruit. What data are needed to distinguish one kind of fruit from another? A kind of frame with the slots which stand for certain features I find relevant for this task that can be instantiated by the values must be defined. For information on schematization into categories and subcategorization in cognition see Lakoff 1987, Langacker 1987, Talmy 1983.

Correspondingly, for the notion of frames in AI see Minsky 1981.

The model will provide a very simple frame represented by a vector with the following slots:

- degree of sweetness, sourness, greenness, yellowness, orangeness, redness, smallness, elongatedness
- presence of pits,seeds

It is important to note that this frame for fruit is intended only as an example that demonstrates the performance of the model, and is not intended to be an accurate frame for all fruit classification.

2.2 Abduction and Prototypicality

Consider two questions that are critical conditions for the system:

a. What does the system regard as being a typical exemplar of a kind of fruit (or class in general) from what it has learned explicitly by examples?

b. To which extend is the set of data given to generate a hypothesis a typical description of an exemplar of a certain fruit (or class in general)?

Of course the second question is dependent on the first. If we see an orange, we judge its typicality for the class orange according to which exemplars of oranges we have seen so far. From the examples we saw which were assured as being oranges we formed a certain kind of prototype of an orange, the image of an orange, that forms the center of the class orange. Eleanor Rosch's work on the phenomenon of prototypicality in categorization (Rosch 1973, 1977, 1978) has attracted the attention of several researchers in cognitive science and in the domain of reasoning. The attraction which prototype-based categorization has held can be seen as stemming from an awareness of serious problems encountered with the idea of defining natural categories by lists of necessary and sufficient conditions for membership. The basic issue here is that some members of a category are better exemplars of the category than others; they are in some manner more central, more prototypical of the category. For example, most people would agree that a robin is a better example of a bird than an ostrich or a kiwi, just as a cherry is a more typical example of a fruit than an avocado. Under a theory of categorization which considers membership in a category to be contingent only on meeting a set of necessary and sufficient conditions, there is no room for such a notion of gradation of membership in a category. This seems important in considering the process of abduction. We can, and want to distinguish between several degrees of prototypicality of hypotheses i.e. how far does a set of given data lead to a prototypical hypothesis and how far does the hypothesis generated differ from the prototype according to the degree of prototypicality of the data. If we have some data and generate from these a hypothesis we want to know the distance between

the typicality of the apple that is hypothesized from the data just given and an apple that is a sure center example of apples. And we want to have an explanation why the other apple is perhaps not so typical an apple. Maybe it is a bit small or a bit too sour or ....

The way of dealing with degrees of typicality is to use fuzzy values. (Zadeh, 1992) The basic idea of fuzzy logic is as follows: The predicates of the classical predicate calculus which have been the components of lists of the old "necessary and sufficient condition categorization" are considered in respect to strictly defined domains of individuals. They are clearly characterized by their domains i.e., by the class of all objects of the domain of individuals or the class of all k-tuples of objects of the domain of individuals when k-place predicates with $k > 1$ are considered. One can say for every individual whether it is a member of all the individuals having a certain predicate or not just as in the old categorization view which was based on this calculus as its formal language. In fuzzy logic the duality principle is surrendered stating that it is either true or false for an individual to be member of the extention of a certain predicate. Statements about the degree of truth of the fact that an individual is a member of a predicate using the range of real number between and including 0 and 1. Quasi-truth values can be interpreted as typicality values, the value 1 denoting the highest possible typicality for an individual for a class, 0 denotes that the individual is not a member of the class.

With the use of the notion of fuzzy sets we can assign to every datum we use for the hypothesis generation and to every hypothesis that is generated a typicality value which is a value giving information about the degree of membership of the datum/hypothesis according to its category or, saying, that the typicality value denotes the distance of the datum/hypothesis to the center of its category. The distance can be computed by substraction of the typicallity value from 1 denoting the center of the category. It is possible for us to extend the process of abduction to handle fuzzy inputs and to evaluate fuzzy outputs which is a very important step.This is because categorization relys on fuzzy factors and data which are often fuzzy when provided by sensors.I will illuminate this later.

In the following I want to simulate the process of abductive reasoning by means of a back-propagation network.

3 Modelling abductive reasoning using back-propagation

3.1 The Back-propagation Model

Applications that must perform a complex data translation which have no predefined mapping function to describe the translation process or those that must provide a best hypothesis as output when presented with fuzzy or noisy input data, are the kinds of problems that are very difficult to solve for more traditional "sequential" problem solving paradigms but are very well suited to be solved by a neural network

as a parallel distributed processing device.

One of the most popular and useful types of such a device is the back-propagation network. This type of network was first described by Werbos (Werbos,1974) and later by Parker (Parker,1985) but was made popular in a broader range by Rumelhart (Rumelhart et al,1986). It is a multi-layered perceptron using the supervised mode of learning. Let me describe how it operates to give the flavour of how it is capable of solving the kind of problem to which it is applied in this paper. I will not cover the mathematical description of the algorithm, for this refer to the above.

The task of the net is to learn a predefined set of input/output pairs.In our case, the inputs are vectors representing the instantiations of the fruit frame where the actual values in the vectors are instantiations of the various frame slots. For example the slot "sweetness" might be instantiated by the value 0.8. The output vector contains slots for every kind of fruit considered. To a certain frame instantiation, the net eventually will answer by highlighting the node standing for the fruit we described in the input vector. The input/output pairs the net learns will serve as examples for later use when it is confronted with inputs it has not learned. It will compare those inputs with all it has learned to make the best hypothesis to which fruit the input vector description fits best. It will not respond in a classical logical, i.e. dual sense, but in a fuzzy fashion. It will output the degree to which it,from what it has learned,is justified to hypothesize a certain fruit type. On the other hand,it may not only highlight one fruit node but alternative fruit nodes as well, if an input has features that are considered to be characteristic for more than one kind of fruit.

In general, the net learns a set of input/output pairs by using a two-phase propagate-adapt-cycle. After an input pattern an instantiated frame in form of a vector has been applied to the nets first neuron layer, it is propagated through each succeeding layer until an output is generated. This output vector is then compared to the desired output which is a vector representing the kind of fruit the input data describes. An error signal is computed for each output neuron. These error signals are then propagated backward from the output neuron layer to each node in the intermediate neuron layer that contributes directly to the output. However, each neuron in the intermediate neuron layer receives only a portion of the total error signal based on the relative contribution the neuron made to the output. This process is repeated layer by layer until each neuron in the net has received an error signal describing its relative contribution to the total error. Based on the error signal that has been received, connection weights are updated by each neuron to cause the net to converge toward a state which allows all training patterns, all data-fruit example pairs, to be encoded. In this process, the neurons in the intermediate layer(s) organize themselves such that different neurons learn to recognize different features of the total input space i.e., the set of all frame instantiations we provide,in order for the net to learn to map onto certain kinds of fruit. After the learning process is completed, the net will be able to recognize frame instantiations that are not identical with any it was trained.It will map it to a vector representing a certain kind of fruit, to be more precise, it will map it to the kind of

fruit that is best described by the provided frame instantiation. Thus the net can handle fuzzy or incomplete data.

Now see how the application of this type of neural nets works in abductive reasoning.

3.2 Hypothesis Generation

To perform abductive reasoning with a neural network we have to interpret input vectors as data sets and output vectors as hypotheses.

The process of hypothesis generation becomes an interpretation of classification of data sets into resulting hypotheses which is not a classification in the sense of classical logic, a la "set belongs to class C or set does not belong to the class C" but the distinction between classes can be fuzzy which actually is just what the idea of prototype theory is. I do not only distinguish whether an object belongs/does not belong to a certain category, but rather to which degree it belongs to a certain category. In other words, how typical it is for this category and also how typical it would be for alternative categories. For the data, each property (sweet, red, has_pits etc.) is assigned one node in the input layer of the net. This means that the input layer of the net is interpreted as representing the fruit frame discussed above.

In the output vector, each fruit name corresponds to a node in the output layer of the network. The network thus maps a set of properties (slot instantiations in the frame) that are activated in the input layer to a node that is interpreted as a fruit name in the output layer. If the net is presented a frame instantiation for which it did not learn a mapping, it provides the best hypothesis to which fruit name a set of data emerges plus all other values of the other nodes, so that an evidence value for every fruit is given. This is a real hypothesis generation as a component of the process of abduction.

For the simulation of hypothesizing kinds of fruit given a set of properties I suggested the following properties to have relevance and thus correspond to nodes (interpreted as frame slots) in the input layer of the net:

sweet, sour, green, yellow, orange, red, pits, seeds, small, elongated

As can be seen,ten nodes are needed in the input layer. These are the kinds of data the net will use to make hypotheses about what fruit it is given. The point is, that various kinds of fruit vary in just these features. Given a set of these features, the net first learns to map such a feature set into an output vector that can be interpreted as standing for a certain fruit name. I want the net to learn to distinguish the following kinds of fruit:

cherry, apple, pear, banana, orange, peach, lemon, grape

This means that I need eight nodes in the output layer.

The properties and the fruit names could be encoded more cleverly with less nodes in the input and output layers but I am more interested in showing the idea how abduction can be simulated, not how to use neural networks inexpensively.

The net has to learn the following mappings which are meant to stand for prototypical mappings,the features to which the net learns the fruit associations which are regarded to stand for quite prototypical exemplar of the respective kind of fruit. I use this just as an example. Recall that prototypical instances of categories are instances in which there is strong evidence for the category from each of a number of different factors, each of which provides partial support for category membership.

sweet, red, pit, small	$\rightarrow$	cherry
sweet, sour, yellow, seeds	$\rightarrow$	apple
sweet, yellow, seeds, elongated	$\rightarrow$	pear
sweet, yellow, elongated	$\rightarrow$	banana
sweet, sour, orange, seeds	$\rightarrow$	orange
sweet, orange, pit	$\rightarrow$	peach
sour, yellow, seeds, elongated	$\rightarrow$	lemon
sweet, green,seeds, small, elongated	$\rightarrow$	grape

These mappings denote the centers of the fruit categories that are associated with them.

If in the input vector a feature is mentioned,the according node in the input layer gets the value > 0,or otherwise 0. I expect the net after learning the mappings,to "turn on" the corresponding fruit node that is higher than others. It shall show a value close to 1.0 for a node in the output layer when I provide an input vector it has explicitly learned as certain hypothesis. If the vector that is provided has not been learned it will give values of more than one fruit node varying in the interval between 0 and 1.

$$0.0 < value < 1.0$$

Here the strongest hypothesis is the fruit node having the highest value. Especially interesting is that we can work with fuzzy values in the input layer, being interpreted as "quite sweet", "a little sour" or "somewhat greenish", etc. If we want to represent a greenish-yellow colour, we would turn on the green node to 0.3 and the yellow to 0.6. We can specify not only whether or not a feature is given but also to which degree. We can therefore model fuzzy data. Practically the system can handle variations in typicality of the input data to provide correct output data. This is important considering the varying degree of the colours of several apples.The system, if it learns prototypes, will be able to deal with a certain (learnable) degree of non-typicality. These properties of the network for abduction are very advantageous when dealt with data directly from sensors. These data are often of an analog rather than a discrete kind but can be handled ably by the net.

Now look at some real results obtained by testing the net. I use a net with 10 neurons in the input layer, 10 neurons in the hidden layer and 8 neurons in the output

layer with a momentum of 0.9 and a learnrate of 0.7. I permitted a maximal total error of 0.01 and an individual error of 0.001. The system needed 184 iterations to converge to a normalized system error of 0.009955. Here are the values of the output nodes when the net had to recognize a prototypical apple having the properties:

sweet, sour, yellow, has_seeds

These nodes were turned on with a value of 1.0, but I also could have used more fuzzy values, as I will show later.

cherry	=	0.007701
apple	=	0.961034
pear	=	0.020421
banana	=	0.003291
orange	=	0.017989
peach	=	0.000234
lemon	=	0.015550
grape	=	0.000168

For a typical lemon with the properties:

sour, yellow, seeds, elongated

cherry	=	0.000667
apple	=	0.026010
pear	=	0.031520
banana	=	0.000561
orange	=	0.039731
peach	=	0.000102
lemon	=	0.955704
grape	=	0.021238

Testing the net on an apple that has a greenish colour and is not so sweet,but also not so sour (green = 0.3, yellow = 0.6, sweet = 0.8, sour =0.7) I get the following results:

cherry	=	0.003971
apple	=	0.227421
pear	=	0.016623
banana	=	0.000058
orange	=	0.051117
peach	=	0.004662
lemon	=	0.023074
grape	=	0.001057

The apple is by far the best hypothesis. Now I test the net on a red apple which has not been presented in the lean phase, yellow is 0, red is 1:

cherry	=	0.018427
apple	=	0.041060
pear	=	0.005367
banana	=	0.000016
orange	=	0.315589
peach	=	0.004339
lemon	=	0.014070
grape	=	0.002197

The net does not recognize this as an apple anymore. It will be made to recognize it by including such an apple into the learning set. This done these are the results:

cherry	=	0.007133
apple	=	0.970771
pear	=	0.001155
banana	=	0.000007
orange	=	0.006833
peach	=	0.002473
lemon	=	0.009231
grape	=	0.000727

The more kinds of fruit and different examples of fruit the net has learned the more precise the hypotheses are. I want to stress once more how easy it is to make the net more knowledgible, by simply adding more input/output vectors to the example set without having to program anything.

3.3 Evaluation and explanation of hypotheses

Once the net generated the scale of hypotheses given a set of test data from which the best i.e., having the highest score = typicality can be chosen, I want to know how this solution is to be evaluated in respect to alternative hypotheses and how this choice of the best hypothesis can be explained and verified with this solution. A disadvantage of the net is that it gives no explicit explanation how it arrived at a particular solution. What I want to know is at least:

a. What is the score of the best hypothesis?

b. What is the degree of evidence that the chosen hypothesis is the "best"?

c. Which data gave evidence and which did not support the hypothesis?

d. Does a value of a feature in the data set presented exceed the values for this feature for a certain fruit that is hypothesized from what the net had learned explicitly?

The questions a) to d) are taken to be the guideline for our further development of the system. I treat a) and b) as belonging to the evaluation component while c) and d) are considered as questions about the explanation of hypotheses.

3.3.1 Evaluation

A nice fact concerning the hypotheses made is that the net provides a degree of typicality for a certain class. If the best hypothesis is for example 0.9, we can be sure that the data describe a prototype of a learned fruit. If it is 0.6, it is still a good hypothesis; if it is 0.3, but still the best hypothesis, we can be sure that of all the net knows it is an acceptable hypothesis if the other alternatives are worse, but one should be careful: there is danger that the knowledge of the net is not sufficient to make a reliable hypothesis in this case, because the test example differs too much from the data describing things it has learned. Compare this problem with that of seeing an unknown animal. We make guesses about what it could be. Say, according to a best hypothesis generated from our knowledge, "Could be a dog". Obvious is the phenomenon in children calling a cow "dog" because they have not yet learned the concept of "cow" explicitly, and from what they know it fits the concept of a dog rather than that of a bird. By giving a typicality value, the net answers question a) since this value is a degree of the absolute quality of the hypothesis generated. The value obtained from the net is a hypothesis as a typicality value for the corresponding fruit.

Besides the absolute value of hypothesis consider the distance of this value to alternative hypotheses as formulated in question b). The smaller it is, the more relatively uncertain it is compared to the alternative. The distance between two alternative hypothesis values can be interpreted as a degree of preference of the higher value over the lower. If If this value of relative preference is low, the certainty that this hypothesis is the best compared to the other is low, doubt is high.If doubt is high, then we want to know why, we want an explanation as to how the system came to the hypotheses it made.

3.3.2 Explanation of hypotheses

In order to give a detailed explanation of its choice of a certain hypothesis the system must proceed from a global evaluation that gives a degree of typicality and certainty to an explanation of the typicality of every single datum upon which the made hypotheses rest, thus turning the process of hypothesis forming backward by asking what are the premises and to which degree are they compatible with the corresponding prototype of the category represented by the hypothesis? For example, if we get a

feature	test	prototype	difference
sweet	0.8	1.0	-0.2
sour	0.7	1.0	-0.3
green	0.3	0.0	0.3
yellow	0.6	1.0	-0.4
orange	0.0	0.0	0.0
red	0.0	0.0	0.0
has_pits	1.0	1.0	0.0
has_stone	0.0	0.0	0.0
small	0.0	0.0	0.0
elongated	0.0	0.0	0.0

Table 1: Hypothesis explanation comparism to prototype (fruit: apple).

hypothesis that the data we fed to the net provide evidence to represent a banana we want for every datum an explanation in how far it contributes to this evidence. There are three steps in solution I offer to this problem. (Actually I will make a tradeoff of all these solutions to combine the best of each.)

The Traditional approach. We could build a simple pattern-matcher that detects in which properties and how far the actual input vector and the vector that was learned to stand for a certain hypothesis differ. The differences shown for every datum. This makes it necessary to store all data to be accessable. For example, the following is the analysis for the apple I tested:

hypothesis:	apple
typicality:	0.227421
relative certainty:	apple = 0.227421,
	lemon = 0.023074

These are the results of the global hypothesis evaluator.

Table 1 looks quite acceptable. Besides the disadvantage of storing all the data the net has learned, there is another one: If the net learns from more than one example of a fruit as many protocols as examples will be produced. This can become confusing and not very fruitful. To avoid a flood of results a way must be found to compare the hypothesis with the object, seen from its features, that is closest to the test object.

Indirect Measurement of Distances between closest learned Object and Test Object by Search in the Input Space. To find the distance between an actual input vector and a vector that stands for a data set the system has been trained on is the following: the input nodes are changed gradually while the effect of the change is meassured as to how far the resulting output value is converging

or diverging towards/from the value for the highest typicality i.e. the output the learned examples produce. By gradually changing the values of the input vector and measuring the result,the data for a learned example can be regenerated.

Using this method we do not have to store the examples that have been learned explicitly, rather we regenerate the one most similar to the test data. We also do not get a whole bunch of explanation because we use only one - the best possible learned example for the explanation or perhaps the second best for certain purposes too.

The question of how precise the regeneration of the input vector of the fruit is depends on how big the steps are chosen when we make little changes in the input node values during the convergence process.After finding the input vector producing the highest value in the corresponding output node, wthe system has to compute the difference between the actual input vector and the regenerated example input vector and then see how far the single data of our actual object differ from its closest learned example. Using this method in combination with a program that does the comparism, the result is a solution in the form of the example of an explanation given above. In the apple example, the closest learned example for the fuzzy test apple I applied, is of course the yellow apple, because the test apple does not have red at all.

There is one addition to the explanation component that should be made : it would be desirable if the system could tell how far the difference between a feature of the test example and a feature of a learned example is typical or in a sense, tolerable. If, for example, the smallest apple the system has learned is an apple with a degree of smallness of 0.3, and the test example has a smallness of 0.9, the size of a cherry, the system should give information that this is untypical. It should inform that a test object has a feature with a value less or greater than any object of the same kind of fruit it has learned. The realization of this ability leads to the third step.

Remembering extreme values. In the learning process the net stores for each fruit that has been presented the extreme values it finds for every feature explicitly in a kind of "memory for the extreme". For example, if the sweetest apple of ten apples presented to the system has a sweetness of 1.0 and the least sweetest has a sweetness of 0.4, then the system has to store these values for apple, forgetting all values lying in between. Using such a database for the explanation, it can always be decided whether the value of the test object lies within the range of values the system has learned or not which can be quite helpful information. This method has also quite a cognitive plausibility. This plausibility derives from the fact that one remembers extreme examples of features of certain objects rather well. Where does awareness to extreme features derive? I want to state that this is a phenomenon of salience. A new feature "super-sour" becomes the new boundary of the category apple and therefore plays a special role as marker. If I take this "theory of the extreme" for granted what I need now is a device that carries out a comparism of the value of feature from the test object with the corresponding minimum and maximum value to determine whether it is in the range of a certain acceptability.

So I can give for every feature a degree of typicality explaining the overall hypothesis step by step.Using this method the system can decide for every feature of the test object whether this feature is:

- a typically high (and how much higher)
- a typically low (and how much lower)

in the typical range, adding information about where it lies in the range.

A final example. If I now combine the indirect search method with the ability to remember extreme values of learning features, the system can give very good information of the following kind:

a. how far the test object differs from the learned object being closest to it,

b. whether or not features of the test object are within or outside the range of values for this feature that the system has learned,

c. which tendency a feature in a range has.

Of course, not only the best hypothesis can be explained, but all others too. We could, for example, provide the system with two kinds of thresholds, one for the typicality and one for the relative certainty that decides which hypothesis should be explained and which not. It would be good if the system explained besides the best hypothesis also the second best, if the distance between both is very small, i.e., the relative certainty for the best hypothesis over the second best is very low.

Here is at last an example produced by the system as it has been described:

Presented was the following data vector (see table 2):

$$\begin{pmatrix} 0.6 & 0.8 & 0.2 & 0.7 & 0 & 0.3 & 1 & 0 & 0.3 & 0.2 \end{pmatrix}$$

Results Generated hypothesis: apple

Best alternative: lemon

Evaluation:

typicality for apple: = 0.74
typicality for lemon: = 0.39
relative certainty: 0.35

note: the feature sour is 0.2 higher than the highest learned for apple.

feature	tested value	learned.ex	max	min
sweet	0.6	0.9	0.9	0.4
sour	0.8	0.6	0.6	0.1
green	0.2	0.0	0.8	0.0
yellow	0.7	0.9	0.8	0.0
orange	0.0	0.0	0.0	0.0
red	0.3	0.0	0.8	0.0
seeds	1.0	1.0	1.0	1.0
pit	0.0	0.0	0.0	0.0
small	0.3	0.0	0.4	0.0
elongated	0.2	0.1	0.3	0.0

Table 2: The features of the presented object are compared with the corresponding features of the most similar learned example that has been regenerated and with the maximum and minimum values that have ever been seen for this object during learning process

4 Discussion

We could show that the system I introduced worked quite well in many respects. It is able to form hypotheses about the membership of a test sample to a certain kind of fruit, on the experience the neural net gained when it was trained on a set of examples that did not have to include the test example. We could speak of this process as a kind of categorization the system learned from presentation.

The system gives a lot of information about its hypotheses and their evaluation. In particular the evaluation of the typicality for a test sample as to its distance from the learned examples of the kind of fruit is a starting point for using the method of hypotheses forming in real applications. The same holds for the ability of the system to measure relative distances between the certainty values of hypotheses.

There are two things I would like to mention that were not of concern here, but they are of some importance when this method is applied to larger and more complex applications. The first is regarding the abductive process itself. The problem is as follows:

Imagine a system that can generate the same hypotheses as this one,and in addition for example can hypothesize whether a test sample can be grown in a certain climate. So one gets more than one hypothesis at the time on different levels. It may be a matter of context which hypothesis is really needed. Even a better example is the following which is chosen for illustration:

Think of a house burning down. Now one could hypothesize how it came to that. Here are some hypotheses that might be true but not relevant in a context:

a. The house burned down because there was a leaking gas pipe.

b. The house burned down because there was fire attacking it.

c. The house burned down because it had been built.

and even

d. The house burned down because the house burned down.

With these examples I want to show that there should be a kind of bias for making a certain kind of hypothesis if there are more than one possibilities.

The second problem I want to address is that there is no 100% guarantee that the net finds really the closest learned example when it does the search in the input space, although this method should be quite reliable. The reason for that is that because I use non-linear functions in the propagation during the learning process, there is no one-to-one mapping from both sides possible.

The two problems just outlined are of different kind due to their handling: the first one is solved by taking care of it by building a mechanism which selects according to the actual relevance of a hypothesis within the context and has nothing to do with the system itself, being only a system extention. The second is regarding the explanation process performed by the system, it can be minimized by a highest possible number of learned examples or by explicit comparison of the input vector of the test sample and the closest learned input vector.

Finally, the following question could arise. Why is what I did here not just pattern identification? In the technical sense of what the net actually does it is pattern identification, although I prefer to say pattern evaluation as will be explained below. closest patterns. There are two important reasons:

1. The "pattern identification" is only a part of abductive reasoning, the part of the hypothesis generation. The other two parts are:

- the data selection ,here simplified by providing a frame giving information about which features are to be selected as input for the hypothesis generation
- the evaluation/explanation gives an interpretation of the "identity value" of the "pattern".

What is important is that the explanation component gives reasons for a certain degree of identity : "the apple is untypical *because* it is smaller than any other I have seen before" When I say "pattern identification" I say something about a part of what is going on technically, as I could say that modus ponens is a pattern completion or that inclusion is pattern partitioning.

To give one more example why it is at least a special type of pattern identification: Do you say the "xor" problem is pattern identification? Again, I could say yes, for this is what such a net does, but I am not precise enough because the net learns something interpretable as a rule from truth-values into a truth-value. In a similar

way I see abduction as a special kind of a process involving components that could be viewed as pattern identification.

2. The output of the process is not just information about whether the data of a test sample are identical to those the system was trained on. Rather what is done here could be called pattern proximum evaluation because if patterns are presented the system cannot identify with any it has learned, It gives as output approximation values that say how "close" the pattern given is to those it has learned.

5 Conclusion

As I showed, the process of abduction which plays an important role in various domains of cognition and theory forming can be performed by a neural network using the back-propagation algorithm. In particular, the following features make such a model powerful, interesting and attractive:

1. The model can learn a set of right hypotheses by being presented only inputs (data) and outputs (hypotheses), without the specification of explicit rules that have to be implemented.

2. The model performs well in recognizing learned data-hypothesis-pairs as well as finding hypotheses for not learned input data.

3. The model is capable to deal with fuzzy input data.

4. Because rules have not to be explicitly formulated the model can easily be extended by only presenting new data-hypothesis-pairs, enhancing thereby its knowledge and precision.

5. It is attractive and easy to implement the model on a parallel computer for high speed.

6. By applying this model of abductive reasoning one gets information about the degree of prototypicality of the hypothesis generated,i.e.,the distance between the center of a category, represented by the learned data-hypothesis-pair,and the pair consisting of the data given and the hypothesis generated by the model.

7. The system can produce a scale of hypotheses that gives the possibility of determining the relative certainty of the best hypothesis by computing the distance between it and the closest alternative.

References

(Freeman, Skapura, 1991) James A. Freeman / David M. Skapura, Neural Networks: Algorithms, Applications, and Programming Techniques, Edisson-Wesley Publishing Company, 1991

(Harman 1965) G. Harman, The inference to the best explanation, Philosophical Review, vol. lxxivJanuary 1965

(Josephson, 1993) John Josephson, Abductive Inference: Computation, Philosophy, Technology Edited by John R. Josephson and Susan G. Josephson 1993 Cambridge Cambridge University Press 1993 (unpublished)

(Lakoff, 1987) George Lakoff, Women, Fire and Dangerous Things: What Categories Reveal about the Mind, University of Chicago Press 1987.

(Langacker, 1987) Ronald Langacker, Foundation of Cognitive Grammar I: Theoretical Prerequisites, Stanford University Press 1987.

(McDermott, 1985) Drew McDermott and Eugene Charniak, Introduction to artificial intelligence, Addison-Wesley, c1985.

(Minsky, 1981) Marwin Minsky, A Framework for Representing Knowledge in Mind Design, edited by J. Haugeland, Cambridge, MA MIT Press, 1981, pages 95-128.

(Minsky, 1969) Marvin Minsky and Seymour Papert, Perceptrons : an introduction to computational geometry Cambridge, MA : MIT Press, 1969.

(Parker, 1985) D. B. Parker, Learning Logic, Technical Report TR-47, Center for Computational Research in Economics and Management Science, MIT, Cambridge, MA, April 1985.

(Rosch, 1973) Eleanor Rosch, On the Internal Structure of Perceptual and Semantic Categories in Timothy E. Moore, editor, Cognitive Development and the acquisition of Language, pages 111-144, New York, 1973, Academic Press.

(Rosch, 1977) Eleanor Rosch, Human Categorization in Neil Warren, editor, Studies in Cross-cultural Psychology, volume 1, pages 1-49, London, 1977, Academic Press.

(Rosch, 1978) Eleanor Rosch, Principles of Categorization in Eleanor Rosch and Barbara B. Lloyd, editors, Cognition and Categorization, pages 27-47, Hillsdale, NJ, 1978, Erlbaum.

(Rumelhart et al.,1986) David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams, Learning internal Representations by Error Propagation in James L. McClelland and David E. Rumelhart, editors, Parallel Distributed Processing: Explorations in the Microstructure of Cognition, pages 318-362, MIT Press, 1986.

(Talmy, 1983) Leonard Talmy, How Language Structures Space Technical Report 4, Institute of Cognitive Studies, University of California at Berkeley, January 1983.

(Werbos, 1974) P. Werbos, Beyond Regression: New Tools for Prediction and Analysis in the Behavioural Science, PhD thesis, Harvard, Cambridge, MA, August 1974.

(Zadeh, 1992) L. A. Zadeh, An Introduction to fuzzy logic applications in intelligent systems edited by Ronald R. Yager, Lotfi A. Zadeh. ¡1992¿ Boston : Kluwer Academic, c1992.