DIVISION OF COMPUTER SCIENCE

The Use of Metrics in Connectionist Psychological Models

Caroline Lyon and Neil Davey

Technical Report No.157

October 1992

The use of metrics in connectionist psychological models

Caroline Lyon and Neil Davey

Jan 1993

Abstract

Models of psychological and cognitive phenomena that are based on connectionist processing have recently been described. These include Norris's back propagation model [1], Schyns's model based on Kohonen nets [2] and Hinton and Shallice's model that adds a recurrent layer to an MLP type network[3] This paper looks at some of the ways data are represented and at the metrics that are employed in these models. It investigates the techniques that are appropriate for different processing tasks in connectionist networks. The term "connectionist network" is an alternative to "neural network", which is the name more often used in the computer science literature.

1 Introduction

During the operation of a neural network it is necessary to use measures of similarity, to examine how close one set of data is to another. This feature is common to most neural network models, appearing in various guises. For a typical multi-layer perceptron (MLP), or back propagation model, the actual output from the net will be compared to the desired output during training. Then connection weights can be adjusted to minimise the difference between actual and desired values. In self-organised Kohonen-type nets one set of data will be compared to another, so that similar sets can be grouped together. With reinforcement learning actual actions or states are evaluated against targets. Now, the appropriate measures of similarity vary with the type of data being assessed. This paper looks at implicit assumptions about the nature of the data being processed with the use of different metrics. Whether these assumptions are realistic can then be explored.

The input and output of neural nets are sets of elements that can be represented as linear arrays or vectors. The weights on connections to a neuron within a neural net can also be represented in vector form, where a set of vectors make up the weight matrix. Thus the measures of similarity which we shall

examine are typically measures of how "close" one vector is to another. In the psychological models examined here the comparison of vectors is an integral part of their operation.

This paper will first look at some commonly used metrics. It will give an overview of the three models and their use of these metrics (Section 3). It will then look more closely at the use of these metrics and give an assessment of appropriate tools for different tasks (Section 4).

2 Commonly used metrics

2.1 The Euclidean distance measure

This is a measure of the difference between two vectors of the same length. It is the sum of the differences between each corresponding pair of elements. Consider two vectors with n elements each:

$$x_1, x_2, \ldots, x_n \qquad \text{and} \qquad y_1, y_2, \ldots, y_n$$

Then the Euclidean distance D between them is given by:

$$D^2 = \sum_{i=1}^{n} (x_i - y_i)^2$$

If $n = 2$ then this is just an expression of the theorem of Pythagoras.

The Root Mean Square (RMS) measure is related to the Euclidean distance. As the root of the mean of the squared distance between each pair of elements

$$RMS^2 = \frac{1}{n} D^2$$

2.2 Hamming distance

In the case of binary vectors this is conceptually similar to Euclidean distance. It is the number of corresponding elements that differ. Suppose two binary vectors have D elements that differ. Then

$$\begin{aligned} \text{Hamming distance} &= D \\ \text{Euclidean distance} &= \sqrt{D} \end{aligned}$$

This metric can also be used to measure distance between any two vectors with ordered discrete valued elements

2.3 Generalized form of Euclidean distance

The Euclidean distance is a special case of the more general Minkowski metric. Consider again two vectors with n elements each:

$$x_1, x_2, \ldots, x_n \qquad \text{and} \qquad y_1, y_2, \ldots, y_n$$

Then the Minkowski distance D_M between them is given by:

$$D_M^\lambda = \sum_{i=1}^{n} (x_i - y_i)^\lambda$$

where λ is a real number ≥ 1.

2.4 The cosine measure

Another measure of similarity is the cosine between two vectors. Whereas the Euclidean distance measured the difference between vectors, this measures their similarity, a reciprocal concept. Intuitively we are saying that if you take two vectors *of the same length* and find they go in the same direction, then they are close to each other. As the angle between two vectors declines the cosine approaches 1. If the cosine is 0 then the vectors are orthogonal, or at right angles in 2 or 3 dimensions. Now, the formula for the cosine θ between two vectors $\mathbf{v}$ and $\mathbf{w}$ is given by

$$cos\theta = \frac{\mathbf{v}.\mathbf{w}}{||\mathbf{v}||\,||\mathbf{w}||}$$

where $\mathbf{v}.\mathbf{w}$ is the inner product of vectors $\mathbf{v}$ and $\mathbf{w}$.

In other words if vectors $\mathbf{v}$ and $\mathbf{w}$ have n elements, or are in n-dimensional space, then

$$\mathbf{v}.\mathbf{w} = v_1 w_1 + v_2 w_2 + \ldots v_n w_n$$

and $||\mathbf{v}||$, the norm or length of $\mathbf{v}$ is given by

$$||\mathbf{v}|| = \sqrt{v_1^2 + v_2^2 + \ldots + v_n^2}$$

The derivation of this formula is found in the original PDP book [4, chap 9] and in many books on linear algebra.

When this measure is used it can be necessary to normalize vectors so that they are the same unit length. This operation adjusts the length of the vector to $||\mathbf{1}||$, while preserving the relative size of each element. Then the normalised vector $\mathbf{v}'$ is

$$\mathbf{v}' = \frac{v_1}{||\mathbf{v}||} + \frac{v_2}{||\mathbf{v}||} + \ldots + \frac{v_n}{||\mathbf{v}||}$$

and the cosine measure is just the inner product of the two vectors:

$$cos\theta = \mathbf{v}'.\mathbf{w}'$$

If vectors are normalised the results of the cosine measure can be interpreted so that it is equivalent to the reciprocal Euclidean measure, since

$$||\mathbf{v}' - \mathbf{w}'||^2 = \mathbf{1} + \mathbf{1} - \mathbf{2}(\mathbf{v}'.\mathbf{w}')$$

2.5 Log probability or cross-entropy measure

This interprets real valued elements of a vector as probabilities that the individual components of a binary vector have the value 1. A set of such vectors therefore defines a probability distribution for each element. If the desired output of a net is the set of vectors:

$$\{\mathbf{d^i}\}$$

and the actual output is

$$\{\mathbf{a^i}\}$$

then the cross entropy is:

$$C = -\sum_{i,j} d_j^i log_2(a_j^i) + (1 - d_j^i) log_2(1 - a_j^i)$$

When C is minimised the probability of producing the correct output is maximised.

Treating the vectors as probability distributions is likely to be appropriate when the training data is probabilistic, or fuzzy. An example is the association between symptoms and diseases in medical diagnosis. It is typically used to give a likelihood that a neuron is going to fire and relates the desired output probability to the actual probability. It is described in [5, p. 207] and in [3, p.80].

3 Connectionist models

3.1 The idiot(savant)

Norris's paper [1] describes the development of a connectionist model of an "idiot savant" date calculator using an MLP. The Euclidean distance measure is used to compare desired with actual output. This network should calculate the day of the week on which any given date between 1950 and 2000 falls. The model is based on the original back-propagation architecture described in *Parallel Distributed Programming* by Rumelhart et al. [4]. There are seven

outputs, one for each day. The input was initially a binary vector with 58 units that coded the input date in the following way. There were 31 units for the day field, 12 for the month field, and 15 for the year field. The year had 5 units to code the decade, 10 to code the last digit of the year. The net was trained on 1000 presentations of 3650 random dates. Performance on unseen dates was scarcely better than chance.

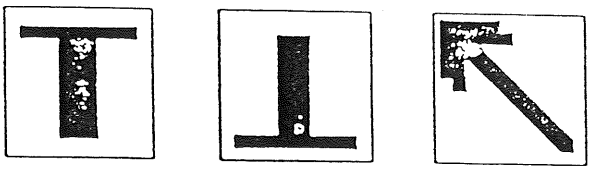
A later version that divided the task into three modules with a human co-ordinator was much more successful. In this case the first module did a simple mapping of the days of the month of January 1950 onto days of the week. The second module mapped any date in 1950 onto a weekday. It took as input the weekday output from the first module and coupled this with the month. The final module mapped any date from 1950 to 2000 onto a weekday. It took as input the weekday output from the previous module together with the month and the year. Though overall performance improved, the errors related to leap years remained high.

In fact, this sort of model is never likely to perform very well. A neural net can of course always learn a mapping from one particular pattern to another, given an appropriate representation. However, to be useful it must learn many mappings and be able to generalize to unseen data. This requires that patterns that are near each other in input space should typically be near each other in output space. The date calculator does not meet this requirement. Changing a single element in the input vector can result in any of the 7 output units firing.

Another example of this limitation on the use of connectionist methods is found in the MLP network designed to divide a given number by 3 in [6].

3.2 The concept acquisition model

Schyns [2] implements two stages of neural net processing in order to model concept acquisition. In the first stage he uses a Kohonen net to process visual symbols that represent prototypes of the concepts *Dog, Cat* and *Bird*, as in the figures below. .



Dog. *Cat.* *Bird.*

. Prototypes of the categories *dog, cat,* and *bird.*

Noisy exemplars of the prototypes are fed to the net, which organises them

in unsupervised mode. The cosine similarity measure is used. Specific regions of the output map will respond to exemplars of a given category. Schyns finds that the simulation agrees with the prototype theory of concept acquisition. The noisy exemplars presented to the network are locally organised into groups, depending on their proximity to the prototypes.

The second stage in the concept acquisition process is to name each exemplar, each representative region of the first stage output map. This is done using a "Brain State in a Box" neural net - a variation on the Hopfield architecture. The input to this stage is a vector combining data from the previous output with a binary ascii representation of the name. This method successfully allocates a name to each of the three categories here, since the problem is sufficiently small.

In general this sort of approach is likely to meet with difficulties, as Hopfield nets can only store a limited number of states. The ascii representation needs further thought, since the conceptual similarities and differences apparent to humans will not be reflected in the binary vector.

3.3 Modelling acquired dyslexia

In "Investigations of acquired dyslexia" [3] Hinton and Shallice use an MLP with an added recurrent layer to show that typewritten words can be mapped onto semantic concepts. Their network initially output semantic feature vectors when presented with letter strings. They then damaged the network and showed it exhibited characteristics that resembled those found in deep dyslexia. The input is a word composed from a restricted domain of letters, using $[b,c,d,g,h,l,m,n,p,r,t]$ in the first position,$[a,e,i,o,u]$ in the second, $[b,c,d,g,k,m,p,r,t,w]$ in the third and $[e,k]$ in the last. There are 28 possible words or grapheme units for the input. These words describe items in 5 categories: indoor or outdoor objects, body parts, animals and foods.

The output from the net is a binary vector which represents the meaning of a word using 68 semantic feature elements. Each of these "sememe units" belongs to one of the semantic feature subsets, denoting size, colour, location, structure, purpose etc. If a particular sememe is deemed present it would be flagged to 1, else it would be 0.

The net has an MLP type architecture with one hidden layer. To this is added a layer through which the output recirculates. This feature was introduced to "clean up" the output and to encourage the output layer to build basins of attraction for words in similar semantic categories.

In training the net the desired output is compared to the actual output using the cross-entropy measure. Once the net has been trained and then damaged to simulate acquired deep dyslexia the output vector between the pre- and post-damaged states is compared. The cosine metric is used to do this.

The paper concludes that the damaged network exhibited characteristics that resembled several of the phenomena found in deep dyslexia and semantic

access dyslexia. A striking example is the case of the patient shown the word "peach" on a card and asked to read it says "apricot".

4 Tools for the job

This section examines the limitations and advantages of different metrics. It looks at the way in which they are used in the three models and how they might best be employed.

4.1 Scope of Euclidean distance measure

While all the elements of the vector are of the same sort this is an appropriate measure. However, suppose that element j is a different sort to element k. Then $\mid x_j - y_j \mid$ could have a different significance to $\mid x_k - y_k \mid$ but the Euclidean distance measure would obscure this. As an example take 2 vectors whose elements are the digits of a binary number. The Euclidean distance D between the pair of vectors:

$$[0,0,0] \qquad [0,0,1]$$

is the same as that between

$$[0,0,0] \qquad [1,0,0]$$

In both cases $D = 1$. But the numerical difference is 1 in the first case, 4 in the second. In fact the measure can be even more contrary. Consider the following pairs of binary numbers:

$$[0,0,0,0] \qquad [1,0,0,0,]$$

and

$$[0,1,1,1,] \qquad [1,0,0,0]$$

Numerically the first pair of numbers differ by 8 while the Euclidean distance is 1. The second pair of numbers differ only by 1, but the Euclidean distance is the maximum for 4-element binary vectors at $\sqrt{4}$.

Similar problems can arise when letters are represented in binary Ascii code. The difference between two letter vectors is an arbitrary measure.

This confusion can be avoided by using a less compact, unit basis, representation. Suppose, for the moment, that an integer is represented by an element of a vector, such that each element represents a different number. Then

$[0,0,0,0]$ represents .. 0
$[0,0,0,1]$ 1
$[0,0,1,0]$ 2
$[0,1,0,0]$ 3

Using this notation the Euclidean distance between pairs of numbers would not give a contrary result: that a close pair had a larger distance than a pair further apart. However, it would merely give the same distance measure between any two different numbers. In order to capture the relative difference another type of representation is needed, such as cumulative flagging. Then

$[0,0,0,0]$ represents .. 0
$[0,0,0,1]$ 1
$[0,0,1,1]$ 2
$[0,1,1,1]$ 3

The general point is that when the output vectors have an interpretation in which a metric is applicable, it is desirable that items that are close in the representation space are also close in the interpretation space. Technically, suppose that X and Y are topological spaces, with X being the representation space, or input, while Y is the interpretation space, or output, onto which these vectors are mapped by a semantic function σ. If

$$\sigma : X \rightarrow Y$$

then we simply require σ to be a topology preserving, homeomorphism from X to Y.

This point is brought out in the results from a project that converts text to phonemes using an MLP [7]. The output vector represents phonemes, and errors in output are measured by Euclidean distance. Now, phonemes can be represented so that they display topological relationships [8], but can also be coded as symbols. This project compares performance using different coding schemes. Codes that preserve information about the phonemes' relationships give better results than compact coding that obscures this.

4.1.1 The Euclidean metric in Norris's model

One of the reasons why the net in this model met difficulties was the nature of the similarity measure between actual and desired output. If the actual output was wrong by one day or wrong by 3 days the error measure between the two 7 element vectors would be the same. The situation can be more confused if there is no gating mechanism to prevent more than one output neuron from firing. If the right output neuron fires together with several wrong ones there can be a larger error than if the right neuron fails to fire at all, while a single wrong one does.

4.2 Scope of the cosine measure

First it must be emphasised that similarity depends on length as well the angle between vectors. The cosine between two vectors is not a measure of their similarity if they are of different length.

Secondly, there are underlying constraints similar to those that apply to the Euclidean distance metric.

This cosine measure is often used in Kohonen nets (though Kohonen himself originally used the Euclidean distance metric [9, p 33]) so consider first the limitations on the sort of data which these nets can process. The basic processing idea is that similar input vectors will organise themselves into a cluster, so that the input is divided into separate classes. Central to to this self-organising procedure is the repeated comparison of vectors. In his original "Self-Organisation and Associative Memory" Kohonen stresses that only certain types of data can be processed in this way:

> "One must emphasize that all types of input information cannot be self-organized....it is necessary that the universe of the input data has some kind of *metric* defined all over it. Such a metric indeed exists for the usual *primary sensory signals*...At higher hierarchical levels, on the other hand, the existence of ordered maps is still unclear" [9, p. 266].

To process feature vectors with discrete symbolic elements in this way is an uncertain exercise.

An example can be found in a field far removed from psychological models, but relevant just the same. In a British Airways research project different neural network methods are investigated for monitoring engine condition [10]. One of the methods explored is using Kohonen's method to classify vectors, whose elements are various parameters derived from measurements of temperature, pressure, speed etc. There is no metric defined over the whole input field. Of this method the paper says:

> " There is no guarantee that the coding which the network produces is going to be entirely relevant to the problem in hand; the variable which most affects the 'distance' between the records of the input data is not necessarily the one which is the best predictor of the target quantity! "

Two input vectors could appear close even when some very significant factors differed, so long as a number of much less significant factors matched.

4.2.1 The cosine measure in Schyns's model

Schyns uses a Kohonen net in an appropriate way by using a visual symbol to represent a prototype of a concept. The operation is all in the domain of image

processing. In essence input vectors are compared, using the cosine measure, to the weight vectors afferent to each output node. A response is triggered where the similarity is greatest. Since each input vector is an array of 100 elements, each +1 (white) or -1 (black), all vectors have the same length and do not have to be normalised.

4.3 Choosing appropriate metrics

4.3.1 Examples from Hinton and Shallice's model

Consider the binary output vector of this network in its 68-dimensional sememe space. The elements of this semantic feature vector fall into 2 groups of subsets: those which are fully dependent and those which are not. For example, all objects in this domain of physical objects must fall into one, and only one, of the size categories. This is a set of dependent sememes. However in other subsets objects may flag up either, both or neither of two sememes, for example "wild" and "fierce". These sememes are connected but not dependent. Furthermore, the subsets themselves are not fully independent. Certain sememes from different subsets are commonly associated.

The authors address this situation in two ways. First, there are constraints in the form of strong negative lateral correlations to inhibit more than one of several competing units from firing. Secondly, the output recirculates through the "clean up" layer which encourages positive lateral correlations.

It could be possible, however, to extract more information from the output vector. Taking this model as an example let us first look at a case where the Euclidean measure is preferable to the cosine. In the semantic feature vector consider a subset of dependent sememes. Those representing size are presented as 3 categories:

[under 1ft]	[1ft to 2yds]	[over 2yds]
[small]	[medium]	[large]

The value of the activity of each will be binary [3, p 83], to be interpreted as a decision on whether the word being processed represents an object that falls in that class. A limitation in this approach is that there is no measure of relative differences: if a small object is flagged "medium" or "large" it will appear equally wrong.

Another way of representing the data that makes more information explicit is to use a cumulative representation, similar to that described above 4.1. The units would then fire in the following way:

$[0, 0, 1]$ for length.. $>$ 0.....(1)
$[0, 1, 1]$ ..... $>$ size 1.....(2)
$[1, 1, 1]$ .....$>$ size 2.....(3)

Using this representation the relative difference between (1) and (2) compared to (1) and (3) is reflected in the Euclidean measure.

But now note how the relative difference measured by the Euclidean distance (section 2.1) and the cosine metric (section 2.4) and the Euclidean distance vary:

vectors compared	cosine of angle between them	Euclidean distance
(1) and (2)	.707	1.0
(2) and (3)	.817	1.0
(1) and (3)	.577	1.414

The Euclidean distance is a better metric to determine which of the 3 dependent classes determining size is selected. The difference between (1) and (2) is the same as that between (2) and (3), but less than that between (1) and (3).

Using the cosine measure, in which the numerator is the inner product of the vectors, matching '1's give a stronger correlation than matching '0's. This gives the result, less desirable in this case, that vectors (2) and (3) are closer than (1) and (2).

The Hamming distance is a simple measure, conceptually similar to Euclidean distance. If other simplifying assumptions about the representation of data have already been made this measure may be appropriate. In measuring the distance between binary vectors nothing is lost by using the Hamming instead of the Euclidean measure. However, the Hamming measure cannot easily be slotted into a back propagation training algorithm, since this requires a differentiable error function.

However, now consider those cases where the cosine measure is preferable. Take another subset of sememes that are not fully dependent, such as "mammal", "wild", "fierce" etc. Here the positive correlation between semantic features that are flagged is, for classification purposes, more useful than negative matches. Hinton and Shallice chose the cosine measure for comparing output vectors because:

> "By comparison with a Euclidean distance measure, proximity is more sensitive to changes toward other possible stored-meaning vectors ...".

In this case the cosine metric is a more appropriate measure from one point of view.

This analysis indicates that it might be worth investigating a different way of measuring similarity between 68- dimensional sememe vectors. It could be more profitable to divide the vector into its constituent subsets and measure the similarity between these subsets, using the appropriate metrics in each case.

However, the use of the cosine metric with *binary* vectors should also be assessed from another point of view. Implicit in its use is the assumption that

each sememe is of equal importance. If "is a mammal" is a more important discriminating factor than "is brown" this measure will not detect it. Words that represent objects that differ on significant factors may appear close if they match on less significant ones.

4.4 Avoiding inappropriate metrics

Diversionary tactics can be used to convert a problem to a domain in which available metrics are suitable. Schyns's method that converts symbolic concepts into a visual image is an example of this.

Another approach is to present the processing task in such a way that the use of similarity metrics are minimised. If the output from a feed forward network, such as an MLP, has only two nodes the problems of comparing the target output vector to the actual are sidestepped. In this case the data has to be represented and the processing task formulated in a way that the net merely classifies input into one of two classes. A language processing task, for instance, might present strings of symbols that were classified as grammatical or ungrammatical [11].

If this sounds a limited capability consider how a processing task might be decomposed into constituent parts. A neural network does not have to be a monolithic structure: processing symbolic data can be effectively done with hierarchies of networks [12].

Whether this is psychologically plausible I do not know.

5 Conclusion

Neural networks have largely been developed for practical use in fields where elements of the data being processed have some topological relationship. This includes the many applications for different signal processing tasks, as well as the processing of primary sensory data.

When methods developed for these domains are transferred to processing more symbolic data certain assumptions can be smuggled in. It is important that these assumptions should be made explicit so that any problems they present are addressed.

The performance of connectionist systems depend on a number of factors: the architecture, the learning algorithm, the pattern of connectivity. Among these factors must be included the representation of data and the selection of metrics. The choice of representation and of appropriate metrics is one of the critical tasks in developing a connectionist system.

References

[1] D Norris. How to build a connectionist idiot(savant). *Cognition*, 1990.

[2] P G Schyns. A modular neural network model of concept acquisition. *Cognitive Science*, 1991.

[3] G E Hinton and T Shallice. Lesioning an attractor network:investigations of acquired dyslexia. *Psychological Review*, 1991.

[4] D Rumelhart and J McClelland. *Parallel Distributed Processing.* MIT, 1986.

[5] G E Hinton. Connectionist learning procedures. *Artificial Intelligence*, 1989.

[6] Eberhart and Dobbins. *Neural Network PC Tools.* Academic Press, 1990.

[7] R Linggard, D J Myers, and C Nightingale, editors. *Neural Networks for Vision, Speech and Natural Language.* Chapman and Hall, 1992.

[8] H. Lucke and F. Fallside. Application of the compositional representation to lexical access using neural networks. In *ICSLP , Kobe*, 1990.

[9] T Kohonen. *Self-Organisation and Associative Memory.* Springer-Verlag, 3rd edition, 1989.

[10] S.Cumming. Neural networks for monitoring of engine condition data. *Neural Computing and Applications*, 1993.

[11] C Lyon and R Frank. Improving a speech interface with a neural net. In R Beale and J Finlay, editors, *Neural Networks and Pattern Recognition in Human Computer Interaction.* Ellis Horwood, 1992.

[12] C Lyon and R Frank. Detecting structures in natural language using a neural net with rules. In *ICANN*, 1992.