■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:... We briefly consider the case of multi-class classification. In this scenario, class-0 materials are not dangerous, while class-1, ..., class-C materials are dangerous. Here, we assume that class-1 materials represent minimal danger and class-C materials represent maximal danger. Adopting our previous convention, we consider one TN possibility; C true classifications $TP_1, \ldots, TP_C$; C false classifications $FN_1, \ldots, FN_C$; C false classifications $FP_1, \ldots, FP_C$; and mis-classifications of dangerous materials, $MC_{i,j}$, $i, j \in \{1, \ldots, C\}$, $i \neq j$ (where dangerous material of class i is wrongly classified as material of class j).

$i \backslash j$	0	1	2	3
0	TN	FP_1	FP_2	FP_3
1	FN_1	TP_1	$MC_{1,2}$	$MC_{1,3}$
2	FN_2	$MC_{2,1}$	TP_2	$MC_{2,3}$
3	FN_3	$MC_{3,1}$	$MC_{3,2}$	TP_3

$i \backslash j$	0	1	2	3
0	0	1	2	4
1	10	0	8	6
2	20	19	0	16
3	40	39	38	0

$i \backslash j$	0	1	2	3
0	1.	5.	6.	7.
1	10.	1.	9.	8.
2	13.	12.	1.	11.
3	16.	15.	14.	1.

These three tables show the classification types (left) where class i is classified as class j; an expert-specified table of the danger (cost) for each type (middle); the resulting order for the 16 types (right).

•••

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:...

i \ j	0	1	2	3
0	1	39/40	38/40	36/40
1	30/40	1	32/40	34/40
2	20/40	21/40	1	24/40
3	0/40	1/40	2/40	1

=

i \ j	0	1	2	3
0	$w_{0,0}$	$w_{0,1}$	$w_{0,2}$	$w_{0,3}$
1	$w_{1,0}$	$w_{1,1}$	$w_{1,2}$	$w_{1,3}$
2	$w_{2,0}$	$w_{2,1}$	$w_{2,2}$	$w_{2,3}$
3	$w_{3,0}$	$w_{3,1}$	$w_{3,2}$	$w_{3,3}$

Weight table $\langle w_{i,j} \rangle$. Values are normalized to the interval $[0,1]$. Weight $w_{i,j}$ is the weight associated with classification result type $C_{i,j}$, classifying a material of class i as a material of class j.

Thus, whenever a word / stream of individual classification result types (TN, TP, MC, FP, FN) is produced when classifying a stream of materials, the value of $w_{i,j}$ is assigned to the result type $C_{i,j}$. Based on the sequence of $w_{i,j}$ values, one can define and compute the performance value P when a set of materials has been classified.

The middle table presented on the previous page — showing "degree of danger"/estimated "cost of a classification" — should be used to generate "weights" $w_{i,j}$, with $w_{i,j} \in [0,1]$, where the minimal danger value (0) is mapped to 1 and maximal danger (40) is mapped to 0. Here, we consider a classification performance function yielding 1 for best-possible and 0 for worst-possible classification. Considering the example used on the previous page, the "danger" values $v_{i,j}$ of the middle table are mapped to $w_{i,j} = 1 - v_{i,j}/40$. The resulting weight table is shown on this page (top).

...

■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:... We use the weight table from the previous page for the calculation of the performance function value, where class-0, ..., class-3 materials are involved. For example, a stream of eight materials might have the TRUE class 8-tuple

$$(0,1,1,3,0,0,1,3).$$

The classification system might generate the following 8-tuple as classification result:

$$(0,0,2,2,1,0,1,0).$$

Thus, the 8-tuple of result types is

$$(TN, FN_1, MC_{1,2}, MC_{3,2}, FP_1, TN, TP_1, FN_3).$$

The corresponding 8-tuple of weight values is

$$(40,30,32,2,39,40,40,0)/40.$$

If one used the average of the component values of this 8-tuple as overall performance function value, one would obtain the function value

$$\underline{\underline{P}} = (40/40 + 30/40 + \ldots + 0/40)/8$$
$$= (223/40)/8 = 223/320$$
$$\approx \underline{\underline{0.697}}.$$

• **Note.** This approach is based on a pre-defined weight table $\langle w_{i,j} \rangle$ used to assign a classification performance value in $[0,1]$ to each individual material classification result type $C_{i,j}$, $i,j = 0,\ldots,C$. Why is this done?

...

OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:... In the case of binary classification, only $2 \times 2 = 4$ classification result types are possible (TN, TP, FP, FN); when performing multi-class classification, the number of result types is $(C+1) \times (C+1)$. Thus, the number of possible result-type tuples of length L is $((C+1)^2)^L$ — which is 4^L in the case of binary classification where $C=1$. On page 16 (1/16/2023), the normalized values of $\overline{n}_0$, $\overline{n}_1$, $\overline{n}_2$ and $\overline{n}_3$ are used to define the relative occurrences of FN, FP, TP and TN result types in a length-L result-type tuple — e.g., (TN, TN, TN, TN, TP, TP, FP, FP) leading to $\mathbb{n} = (\overline{n}_0, \overline{n}_1, \overline{n}_2, \overline{n}_3) = (0/8, 2/8, 2/8, 4/8)$. Here, the $\mathbb{n}$-space is "only" 4-dimensional, and it is a viable approach to have an expert define specific performance values $p(\mathbb{n})$ for certain $\mathbb{n}$-tuple values; and interpolation can be employed to estimate p-values for any $\mathbb{n}$-tuple value in the domain where $\sum \overline{n}_i = 1$ and $\overline{n}_i \geq 0$. Such an interpolation approach can no longer be used when the $\mathbb{n}$-tuple dimension grows with increasing number of classes, i.e., $C = 2, 3, 4, \ldots$ Therefore, it is generally necessary to pre-define a weight table $\langle w_{i,j} \rangle$ for all possible $(C+1)^2$ result types.

...

Bernd Hamann

1/30/2023
Stratovan

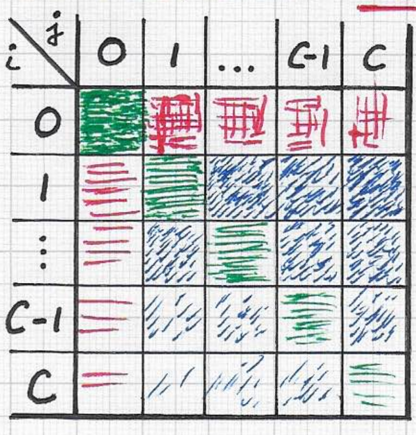
■ OBJECT AND MATERIAL EIGENFUNCTIONS – Cont'd.

• Laplacian eigenfunctions and neural networks:...

$i \backslash j$	0	1	...	C
0	$n_{0,0}$	$n_{0,1}$	...	$n_{0,C}$
1	$n_{1,0}$	$n_{1,1}$	...	$n_{1,C}$
⋮	⋮	⋮	⋱	⋮
C	$n_{C,0}$	$n_{C,1}$	...	$n_{C,C}$

Counter table of numbers of occurrences of classification result types $C_{0,0}, \dots, C_{C,C}$.

Of course, one can generate a counter table (left) as part of the classification system's performance evaluation. For example, if one used a "very long" material stream (of known materials) as input, the system would determine a material class for each classified material as output – and a classification result type would be generated, subsequently increasing the corresponding counter $n_{i,j}$ by 1. If one is interested in knowing the normalized relative occurrences of the result types, one will simply divide the integer-value counters by the number of all classified materials, i.e.: $\overline{n}_{i,j} = n_{i,j}/n$, where $n = \sum_{i,j=0}^{C} n_{i,j}$.



Shaded normalized counter table, considering types and values.

One can think of the "performance tuple" $(\overline{n}_{0,0}, \overline{n}_{0,1}, \dots, \overline{n}_{0,C}, \overline{n}_{1,0}, \dots, \overline{n}_{1,C}, \dots, \overline{n}_{C,0}, \dots, \overline{n}_{C,C})$ as a point in $(C+1)^2$-dimensional space. "THE SYSTEM WILL CONVERGE TO A LIMIT PERFORMANCE." The convergence process could be visualized via a colored and shaded counter table (left).

...