Neural Computing and Applications
https://doi.org/10.1007/s00521-022-07376-1

S.I. :DEEP LEARNING MODELLING IN REAL LIFE: (ANOMALY DETECTION, BIOMEDICAL, CONCEPT ANALYSIS, FINANCE, IMAGE ANALYSIS, RECOMMENDATION)

Self organizing maps for cultural content delivery

Georgios Drakopoulos[1] · Ioanna Giannoukou[2] · Phivos Mylonas[1] · Spyros Sioutas[3]

Received: 3 January 2022 / Accepted: 27 April 2022

Abstract
Tailored analytics play a key role in the successful delivery of cultural content to huge and diverse groups. Primarily the latter depends on a number of information retrieval factors determining user experience quality, most prominently precision, recall, and timing. These imply that cultural analytics should be designed with strong predictive power. In turn, the latter relies heavily on the clustering of the system user base. A self organizing map is a neural network architecture trained in an unsupervised way through a modified Hebbian rule to couple distances between two distinct spaces such that a manifold in the high dimensional space is projected smoothly to the lower dimensional one. The twofold focus of this work is the development of a tensor user distance metric for SOMs as well as the inclusion of behavioral attributes therein, both aiming at additional descriptive power and clustering flexibility. As a concrete example, the proposed SOMs are applied to data taken from a cultural content delivery system. The proposed methodology is evaluated based on a scoring method assessing both complexity and clustering quality criteria, including the number of epochs, the average cluster distance, and the topological error, with encouraging results.

Keywords Self organizing maps · Dimensionality reduction · Behavioral attributes · Higher order analytics · Tensor distance metrics · Cultural content delivery

Mathematics Subject Classification 42A16 · 46E20 · 62H30 · 62P25 · 91C20 · 91C99 · 91D10 · 91D30 · 91D99

1 Introduction

At the dawn of the Internet era cultrual content creation thrives remarkably [27]. This trend has been attributed to combination of social, technological, and economic reasons [50]. Irrespective of the nature of the driving forces behind it, the need for sophisticated analytics for successful content delivery remains. The latter is ultimately assessed by a number of user experience (UX) quality criteria such as precision, recall, and timing. The first two relate to the relevance of the retrieved content to the user, whereas the last one dictates that the right material should be delivered at the right time. To satisfy them a wide array of advanced cultural analytics have been developed [52]. Although design perspectives differ widely, one common element is that some knowledge about the composition of the system user base is required [37]. However, using the intended analytics to cluster this base would defeat the purpose.

✉ Georgios Drakopoulos
c16drak@ionio.gr

Ioanna Giannoukou
igian@upatras.gr

Phivos Mylonas
fmylonas@ionio.gr

Spyros Sioutas
sioutas@ceid.upatras.gr

1 Department of Informatics, Ionian University, Plateia Tsirigoti 7, Kerkyra 49100, Hellas

2 Department of Management Science and Technology, University of Patras, Panepistimio Patron, Patra 26504, Hellas

3 Computer Engineering and Informatics Department, University of Patras, Panepistimio Patron, Patra 26504, Hellas

Published online: 24 May 2022

Unsupervised learning can be the algorithmic cornerstone of addressing this. In particular large user bases can be clustered with self organizing maps (SOMs) [28]. They are a class of unsupervised neural networks trained with a modified Hebbian rule to adaptively formulate the neurons as connections between two distinct vector spaces, the high dimensional *data space* and the low dimensional *coordinate space*. Their training objective is to efficiently project a manifold from the former to the latter while preserving the original topological attributes.

The primary research objective of this work is twofold. First, an extensible and flexible tensor distance metric for clustering the user base of a cultural content delivery system. Second, said metric incorporates behavioral attributes besides the ones typically encountered in recommender systems. This work differentiates itself from previous ones as it is one of the few contributions where a tensor distance metric is coupled with an unsupervised neural network architecture. Moreover, this partitioning extends the cultural analytics proposed in [12].

The remainder of this work is structured as follows. In Sect. 2 the recent scientific literature regarding SOMs and cultural analytics is briefly reviewed. The basic SOM functionality as well as intuition about it are given in Sect. 3, while the proposed methodology is described in Sect. 4. The results obtained from the benchmark dataset are explained in Sect. 5. Finally, in Sect. 6 possible research directions are given. Tensors and matrices are represented with capital boldface letters and vectors with capital small ones, whereas small letters are reserved for scalars. Vectors are always assumed to be columns, unless otherwise explicitly stated. The technical acronyms are explained the first time they are encountered in the work. In the various function definitions parameters follow arguments after a semicolon. Finally Table 1 summarizes the notation of this work.

2 Previous work

The seminal work [28] marked the introduction of SOM as a special class of unsupervised neural networks where distance metrics play a fundamental role [14]. The role of distance metrics in cluster separation is further investigated in [20]. Strong neurobiological evidence suggests that a form of biological SOM is present in the hippocampus region of the human brain [4]. In [47] the nature and the role of interconnected spaces in these SOMs are investigated. SOMs have been used to cluster the player base of a cultural game in [13], to identify the wear in manufacturing tools [5], to process video in a FPGA implementation [46], to discover clusters in graphics and computer vision [43], and support vector machine (SVM) hyperplane optimization for hyperspectral image clustering through an iterative process [23].

The design of cultural content delivery systems is the focus of [53]. Principles for effective cultural analytics design are given in [37]. The conditions under which cultural elements may be lost between generations are studied in [27]. The disposition of cultural goods is explored in [50]. The behavioral aspects of study and text understanding are listed and explored in [51]. Strategies for the coordinated massive content delivery over networks are discussed in [30]. Modern content delivery networks are analyzed in [61]. The implementation of cultural analytics over Neo4j with emphasis on content similarity metrics is the focus of [12]. A case study for the cultural content delivery system specifications tailored for museums is given in [7]. In [18] the music trends across generational cohorts and their effects on the optimization of musical recommender systems are studied.

Other neural network architectures include tensor stack netowkrs (TSNs) which consist of clusters of simpler neural networks, typically feedforward neural networks (FFNNs) [60]. FFNNs are trained by their own errors as well as by those of the other FFNNs in the same cluster [10]. Open topics in TSN research include optimal error diffusion sequences for efficient training [33] whereas applications include network speed in massive urban networks [59], graph connectivity assessment [16], sound classification [25], and cloud classification for assisting weather prediction [31]. Recurrent neural networks (RNN) are architectures where feedforward and feedback loops are allowed, giving them a strong non-linear behavior [44]. Open topics in the field are loop optimization [32], performance metrics [29], and power efficiency in FPGA implementations [17]. RNNs have been applied among others to document summarization [39], intrusion detection in software defined networks (SDNs) [49], and short-term energy predictions [15]. Convolutional neural networks (CNNs) are a class of regularized FFNNs based on a shared synaptic weight principle [1]. Although their primary applications are image processing [6, 57] and computer vision [8], CNNs are also used in vehicular communication [45], emotion recognition [26], and in the prediction of human spatial trajectories based on social properties [38]. Open problems in the field include downsampling [58] and CNN architecture evaluation metrics [48].

Tensor analytics have been recently established as the algorithmic cornerstone for designing new solutions to many problems, old and new alike. These include fMRI image clustering [11], tensor regression [34], tensor principal component analysis [55], discovering community structure based on structural and functional attributes in Twitter [9], and hyperspectral image restoration based on physics-inspired regularization models [35]. Tensor

Table 1 Notation of this article

Symbol	Meaning	First in
$\triangleq$	Definition of equality by definition	Eq. (1)
$\{s_1, \ldots, s_n\}$	Set with elements $s_1, \ldots, s_n$	Eq. (2)
$(t_1, \ldots, t_n)$	Tuple with elements $t_1, \ldots, t_n$	Eq. (23)
$\lvert S \rvert$	Set or tuple cardinality functional	Eq. (7)
$\mathrm{loc}\,(u)$	Grid location for neuron u	Eq. (1)
$\mathrm{invloc}\,(u)$	Set of data points assigned to neuron u	Eq. (4)
$\mathrm{weight}\,(u)$	Synaptic weights of neuron u	Eq. (1)
$\mathrm{bias}\,(u)$	Bias of neuron u	Eq. (7)
$\mathrm{neighbor?}(u_2)$	Indicator of neighboring neurons u_1 and u_2	Eq. (7)
$\mathrm{updated?}\,(u)$	Indicator of synaptic weight update u	Eq. (7)
$\mathrm{assigned?}(\mathbf{s})$	Neuron data vector u is assigned to	Eq. (7)
$\mathrm{neighb}(u)$	Neighborhood of neuron u	Eq. (2)
$\mathrm{prox}\,(u)$	Proximity space of neuron u	Eq. (3)
$\deg\,(i)$	Degree (number of friends) of i-th user	Eq. (27)
$\odot$	Hadamard tensor product	Eq. (18)
$\lVert\cdot\rVert$	Tensor, matrix, or vector norm	Eq. (25)
$\det\,(\mathbf{M})$	Determinant of matrix $\mathbf{M}$	Eq. (8)
$\mathrm{tr}\,(\mathbf{M})$	Trace of matrix $\mathbf{M}$	Eq. (25)
$\mathrm{diag}\,(\mathbf{D})$	Diagonal matrix $\mathbf{D}$	Eq. (18)
$\mathbf{I}_n$	$n \times n$ identity matrix	Eq. (18)
$\mathrm{prob}\,\{\Omega\}$	Probability of event Ω	Eq. (37)
$\langle p \vert\vert q \rangle$	Kullback-Leibler divergence of q from p	Eq. (35)
$\dim\,(S)$	Dimension of vector space S	Sub. 3.1
$S_1 \setminus S_2$	Asymmetric set difference S_1 minus S_2	Eq. (19)

completion strategies based on implicit information gain from transforms are explored in [21]. More recently, tensor subspace clustering is the focus of [54].

3 Cognitive maps

3.1 Function

The fundamental concepts of SOMs are described work to provide the necessary background. Intuituvely speaking, SOMs project a manifold in lower dimensions while preserving its primary topological features. Moreover, this projection comprises of smooth clusters which further allow visualization. Notice that the following analysis relies on a number of parameters whose actual values are given in Table 4.

SOMs learn how to best project a manifold D defined in a vector space $\mathscr{V}$ of dimension $\dim(\mathscr{V})$ to another vector space $\mathscr{C}$ of a much lower dimensionality $\dim(\mathscr{C})$. It is necessary that D is represented by a set of data points D_p with sufficiently variability to describe it. This is especially reflected in the selection of grid size as indicated by Eq. (5). Each neuron $u_{i,j}$ in a two-dimensional grid besides its fixed coordinate vector $\mathbf{c}_{i,j}$ or $\mathrm{loc}\,(u_{i,j}) \in \mathscr{C}$ denoting its location in the grid is also assigned an adjustable synaptic weight vector $\mathrm{weight}\,(u_{i,j}) \in \mathscr{V}$ as shown in Eq. (1):

$$\begin{aligned} \mathbf{c}_{i,j} &\triangleq \mathrm{loc}\,(u_{i,j}) = [i \quad j]^T \qquad \text{and} \\ \mathbf{w}_{i,j} &\triangleq \mathrm{weight}\,(u_{i,j}) \end{aligned} \tag{1}$$

Two key concepts ensuring the proper functionality of an SOM are that of the neighborhood and proximity of a given neuron u. The former is a property of the grid and it refers to the neurons immediately connected to u, whereas the latter depends heavily on the configurable parameters of the distance metric $g(\cdot,\cdot)$ and the threshold ξ_0. The neighborhood of u is formally defned as the set of Eq. (2).

$$\mathrm{neighb}(u) \triangleq \{v \,|\, \mathrm{neighbor?}(u) = \mathrm{true}\} \tag{2}$$

The indicator neighbor?$(\cdot)$ is true for neighboring neurons in $\mathscr{C}$. The three main cases where a neuron v (in gray) can be considered to be a neighbor of u (in black) are shown in Fig. 1. Similar considerations hold for the respective symmetric cases.

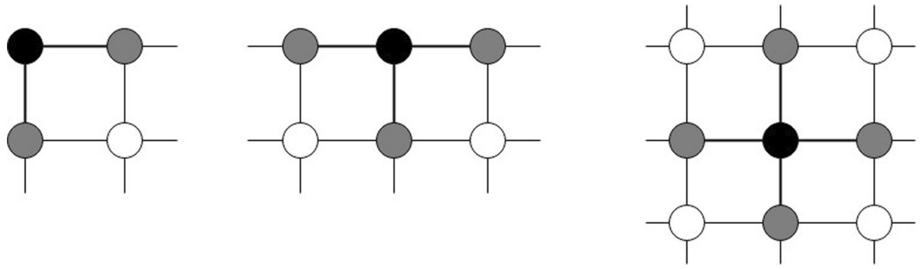

Fig. 1 The main cases of neuron neighborhood

The proximity $\text{prox}\,(u;\xi_0)$ is defined in terms of the proximity function $d(\cdot,\cdot)$ and a threshold ξ_0 as shown in (3). It is proper superset of $\text{neighb}(\cdot)\cdot$ and its role is to form the base of a cluster in $\mathscr{C}$. The neighborhood represents the center of a cluster, while the proximity the body of it. Therefore, the latter should be strictly bigger than the former with the resulting cluster size balancing smoothness and accuracy.

$$\text{prox}\,(u;\xi_0) \triangleq \left\{v \middle| g(u,v) \geq \xi_0\right\} \supset \text{neighb}(u) \tag{3}$$

Let $D_p \triangleq \{\mathbf{d}_k\} = n$ where $\mathbf{d}_k$ are the n data points describing D. Then the set of these points assigned to a neuron u is denoted by the set $\text{invloc}\,(u)$ as shown in (4):

$$\text{invloc}\,(u) \triangleq \left\{\mathbf{d}_k \in D_p \middle| \text{assigned?}(\mathbf{d}_k) = u_{i,j}\right\} \subseteq D_p \tag{4}$$

The number of available data points n determines the size of the $p_0 \times q_0$ square grid. A heuristic to determine p_0 and q_0 based on the entropy of D_p is shown in (5):

$$\begin{aligned} p_0 &= \alpha_p\left\lceil\log\left|D_p\right|\right\rceil = \alpha_p\lceil\log n\rceil \quad \text{and} \\ q_0 &= \alpha_q\left\lceil\log\left|D_p\right|\right\rceil = \alpha_q\lceil\log n\rceil \end{aligned} \tag{5}$$

Given Eq. (5), the density of the grid δ_0 is defined as the number of data points to the number of available neurons as shown in Eq. (6).

$$\delta_0 \triangleq \frac{\left|D_p\right|}{p_0 q_0} = \frac{n}{\alpha_p\alpha_q\lceil\log n\rceil^2} \tag{6}$$

Additionally $\left|D_p\right|$ determines the complexity of the SOM training procedure. Specifically, an iteration is defined as the set of operations necessary for using a single $\mathbf{d}_k \in D_p$ to train a single neuron and its proximity. Moreover, an epoch is defined as the set of operations needed to train the SOM with each $\mathbf{d}_k$ once. The data points need not be used in the same order. In fact, the following have been proposed:

- The $\mathbf{d}_k$ are driven to the SOM at a fixed order. This will be used here.
- The $\mathbf{d}_k$ are driven to the SOM at a random order.
- The $\mathbf{d}_k$ are driven to the SOM at an order causing the maximum cluster change.

From the above it follows that the order of which $\mathbf{d}_k$ are driven to the SOM depends on the data point selection policy and the current iteration. Thus, the notation $\mathbf{d}^{[t]}$ will be used to denote the data point being fed to the SOM at the current iteration.

In SOMs the distance metrics as well as the kernel or weight function play a central role. Mmetric $d(\cdot,\cdot)$ is defined over $\mathscr{V}$ and depends heavily on the nature of the manifold being approximated. For instance, for images it can be the difference between the respective DCT2 coefficients, for graphs the topological correlation or the angle between the respective primary eigenvectors, and for documents the distance on a concept tree. Moreover, $d(\cdot,\cdot)$ determines which neuron should have its synaptic weights updated. On the contrary the distance metric $g(\cdot,\cdot)$ and the kernel function $h(\cdot,\cdot)$ are tied to the grid geometry and operate on $\mathscr{C}$. The combination of $g(\cdot,\cdot)$ and $h(\cdot,\cdot)$ create the clusters of the final cognitive map. For instance, when both are semicircular with the same radius, then the clusters are comprised of semishperes. Possible options are shown in Table 2. It is important [28] that these regions are smooth so there are few discontinuities in the cognitive map.

The bias b or $\text{bias}\,(u)$ of neuron u in sharp contrast to its counterpart in an FFNN is not considered part of the synaptic weights of u and performs instead a much different role. Specicially, it prevents the occurence of dead neurons by monitoring either how many of the neighboring neurons have had their synaptic weights updated, as in this work, or the number of epochs since the last weight update of u.

$$\text{bias}\,(u) \triangleq \frac{|\{\text{neighbor?}(u) \cap \text{assigned?}(v)\}|}{|\{\text{neighbor?}(u)\}|} \tag{7}$$

If an unused neuron is found, then its synaptic weights are the average of those which have been updated. This allows the propagation of weights in unused regions.

Initializing $\text{weight}\,(u)$ is stochastic. For each of the $\dim\,(\mathscr{V})$ attributes stored in the n vectors of D_p the sample average $\mu[r]$ is computed for the r-th attribute and stored at the r-th position of mean value vector $\mathbf{m}$ as shown in Eq. (9). Similarly, the sample autocovariance matrix $\mathbf{G}$ is computed as in Eq. (10). This eventually results in the initialization condition of Eq. (8).

Table 2 Options for region shapes

$g(\cdot,\cdot)$	$h(\cdot,\cdot)$	Shape	$g(\cdot,\cdot)$	$h(\cdot,\cdot)$	Shape
Semicircular	Semicircular	Semispherical	Triangular	Triangular	Pyramid
Semicircular	Semicircular	Semiellipsoid	Triangular	Square	Pyramid
Semicircular	Triangular	Conical	Square	Square	Cubical
Semicircular	Square	Cylindrical	Gaussian	Gaussian	3d Gaussian

$$\text{weight}(u) \sim \frac{1}{\det(\mathbf{G})^{\frac{1}{2}}(2\pi)^{\frac{\dim(\mathcal{V})}{2}}} \exp\left((\text{weight}(u)-\mathbf{m})^T\mathbf{G}^{-1}(\text{weight}(u)-\mathbf{m})\right) \tag{8}$$

In Eq. (8) the mean value vector $\mathbf{m}$ contains the sample averages as in (9):

$$\mathbf{m} \triangleq \begin{bmatrix} \frac{1}{n}\sum_{k=1}^{n}\mathbf{d}_k[1] & \frac{1}{n}\sum_{k=1}^{n}\mathbf{d}_k[2] & \ldots & \frac{1}{n}\sum_{k=1}^{n}\mathbf{d}_k[n] \end{bmatrix}^T \tag{9}$$

Similarly the sample autocovariance matrix $\mathbf{G}$ is computed as in (10):

$$\mathbf{G} \triangleq \frac{1}{n}\sum_{k=1}^{n}(\mathbf{d}_k-\mathbf{m})(\mathbf{d}_k-\mathbf{m})^T \tag{10}$$

The reason for selecting the Gaussian distribution is it has the maximum differential entropy among all distributions of the same variance, meaning it can explain the maximum number of probabilistic scenaria [2, 19]. Also this choice ensures that the synaptic weights have initial values relevant to the data points, which also helps reducing the risk of unused neurons.

There is a wide array of training process termination criteria. A failsafe option is an estimation of the maximum number of epochs. Still, criteria which are indicative of the clustering performance are preferred. These include among others the percentage of data points assigned to different clusters in two successive epochs, the average change of the location of the cluster centroids, and the change of the topological error. In this article the SOM training process terminates if no cluster centroid moves away from the proximity of the respective centroid for three successive epochs. This criterion depends primarily on the the inherent structure of the dataset while it avoids arbitrary hyperparameters. The number of epochs ensures only clustering stability.

3.2 Training

During each iteration t in every epoch T a data point $\mathbf{d}^{[t]}$ is driven to the SOM. The winning neuron u^* for that iteration is the one with the synaptic weight vector closest to that data point. In case of a tie, one neuron is selected randomly. The function assigned?$(\cdot)$ is updated so that $\mathbf{d}^{[t]}$ is linked to u^* for the current epoch T. Moreover, the vector $\text{weight}(u^*)$ is updated such that it gets closer to $\mathbf{d}^{[t]}$ as in Eq. (11). It is a Hebbian learning rule where neurons compete for a reward in the form of the weight update but only one of them is successful in each iteration.

$$\text{weight}(u^*)^{[t]} = \text{weight}(u^*)^{[t-1]} + \mu[T]\left(\mathbf{d}^{[t]} - \text{weight}(u^*)^{[t]}\right) \tag{11}$$

Additionally, the synaptic weights of each neuron v in the $\text{prox}(u^*;\xi_0)$ are also updated as in (12) resulting in region in $\mathscr{C}$ being trained to respond to a single data point. If $h(\cdot,\cdot)$ has a proper decay rate, then this region obtains a smooth shape.

$$\text{weight}(v)^{[t]} = \text{weight}(v)^{[t-1]} + \mu[T]h(u^*,v)\left(\mathbf{d}^{[t]} - \text{weight}(v)^{[t]}\right) \tag{12}$$

The learning rate $\mu[T]$ in (12) ensures synaptic weight update stability. During the initial epochs it is high but gradually decays as clusters are forming. One option is the cosine learing rate of (13) where T_0 is an estimate of the number of epochs:

$$\mu_c[T;T_0] \triangleq \cos\left(\frac{\pi T}{2(T_0+1)}\right) \tag{13}$$

The Taylor expansion of the cosine function in (14) reveals that it drops quadratically. This is in accordance with the desired learning rate behavior described earlier.

$$\cos x = \sum_{k=0}^{+\infty}(-1)^k\frac{x^{2k}}{(2k!)} = +1-\frac{x^2}{2}+\frac{x^4}{4!}-\ldots \approx 1-\frac{x^2}{2} \tag{14}$$

An alternative to (13) is the linear decay learning rate μ_l defined as in Eq. (15). The parameter γ_0 controls the steepness of the learning rate.

$$\mu_l[T;\gamma_0] \triangleq \frac{1}{\gamma_0 T+1}, \qquad T \geq 0 \tag{15}$$

The training of an SOM is summarized in algorithm 1.

Algorithm 1 SOM training.

```
Require: The SOM parameters named below
Ensure: A cognitive map for manifold $D$
 1: for all neurons $u$ in the grid do
 2:     initialize $\mathrm{loc}(u)$ and $\mathrm{weight}(u)$ as in (1) and (8) respectively
 3: end for
 4: repeat
 5:     for all data points $\mathbf{d}^{[t]}$ do
 6:         find the winning neuron $u^*$ and update the assigned?$(\cdot)$ indicator
 7:         update $\mathrm{weight}(u^*)$ as in (11)
 8:         for all neurons $v \in \mathrm{prox}(u^*; \xi_0)$ do
 9:             update $\mathrm{weight}(v)$ as in (12)
10:         end for
11:         if neuron bias is enabled then
12:             update synaptic weights for neurons $u$ with assigned?$(u)$ = false
13:         end if
14:     end for
15: until the SOM is trained
16: return
```

Figure 2 shows how a 2×2 SOM with four neurons placed at the corners of a square grid projects a cylinder in $\mathscr{V}$ to a square in $\mathscr{C}$.

The distance function $g(\cdot,\cdot)$ used in the experiments is the Gaussian kernel of (16). Similarly the weight function $h(\cdot,\cdot)$ was the same kernel but with variance σ_h^2.

$$g\left(u, v; \sigma_g^2\right) \triangleq \frac{1}{\sigma_g\sqrt{2\pi}} \exp\left(-\frac{\| \mathrm{loc}(u) - \mathrm{loc}(v)\|_2^2}{2\sigma_g^2}\right) \tag{16}$$

The Gaussian kernel has been selected for $g(\cdot,\cdot)$ because the boundary between two Gaussians is a line, making thus cluster discovery easier. In the case of the SOM weight kernel $h(\cdot,\cdot)$, the Gaussian kernel has a high value around the cluster centroid but then decays sharply enough such that neurons both across the cluster and even in the periphery of bordering clusters receive a low weight. The latter allows a continuous transition between clusters. A necessary condition to help ensure the latter is $\sigma_h > \sigma_g$. In this way and because of the shape of the Gaussian kernel the reward from a successful neuron-data vector match is distributed within a cluster.

4 Proposed methodology

4.1 Overview

Partitioning the user base of any cultural content delivery system, or for that matter of any system relying on a connection- and similarity-oriented recommendation engine is by no means a trivial task. SOM architectures with tensor distance metrics $d(\cdot,\cdot)$, like the ones presented here, is one way to address this issue by fusing a multitude of base attributes to a single distance metric. Specifically in this work are presented metrics combining the features shown in Table 3. Said features pertain directly to the queries issused by the users, their preferences, and their peer interaction. In fact, as stated in Sect. 1 a major part of the novelty of this novelty of this article stems from the fact that behavioral attributes are used in addition to proposing a tensor user distance metric. In the dataset used the experiments of Sect. 5 there were no null, placeholder, or missing values. Additionally, there are no zero or near zero, namely close to machine precision, values which would have made some computations numerically hard. The dataset size in terms of data points is given in Table 4.

As is the case with most engineering applications, the development of the proposed user tensor distance metrics relies heavily on a number of assumptions. In their core lies the effort to express user engagement with respect to the cultural content delivery system in tangible terms, as explained in detail below.

Assumption 1 (User-to-user interaction tracking) The cultural portal is designed to allow users to befriend each other in two-way relationships. Given this ability predictions can be made in the same way as in a classical recommender system or even in alternative ways. This is reflected in features f_k of Table 3.

Assumption 2 (User personalization) Users personalize the delivery system indirectly as they filter cultural content through topic selection. Thus users can be clustered based on their pereference similarity as in conventional recommender systems. The most frequent topics for the k-th user are stored in the list s_k as shown in Table 3.

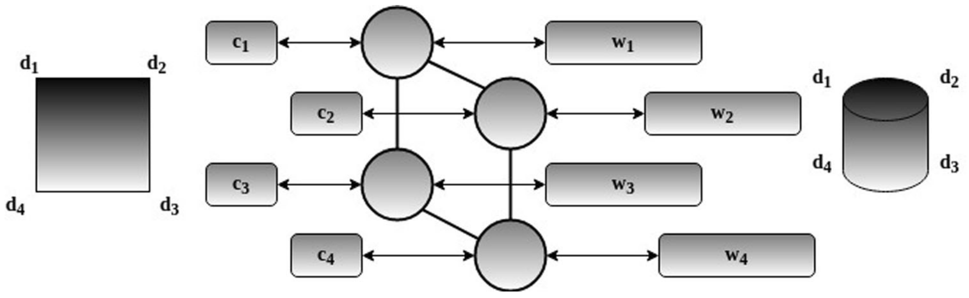

Fig. 2 A sample 2×2 SOM

Table 3 Dataset attributes for the k-th user

Attribute	Meaning	Type	Notes
$c_{1,k}$	Number of sessions per month	Behavioral	Scalar, min: 1, max: 52
$c_{2,k}$	Average session duration (hours)	Behavioral	Scalar, min: 0.83, max: 5.1
$c_{3,k}$	Average keywords per query	Behavioral	Scalar, min: 2, max: 7
$c_{4,k}$	Average queries per session	Behavioral	Scalar, min: 4, max: 21
f_k	List of friends	User	Set, cardinality min: 4, max: 68
s_k	List of cultural topics	Topic	Set, cardinality min: 5, max: 148

Table 4 Experimental setup

Parameter	Value	Parameter	Value
Number of training data points $\lvert D_p \rvert$	5148	Data driving policy	Fixed order
Scaling parameter α_p	10	Grid length p_0	121
Scaling parameter α_q	10	Grid height q_0	121
Distance metric $g(\cdot, \cdot)$	Gaussian	Kernel function $h(\cdot, \cdot)$	Gaussian
Variance σ_g^2	8	Variance σ_h^2	10
Threshold ξ_0	0.1	Bias policy	Updated neighbors
Local system load ρ	0.14	Repetitions R_0	1000
System memory	8 GB	System processor	Intel Core i5-3210M

Assumption 3 (Query format) User activity is primarily reflected in the query formulation. This behavior is also tied to the user predisposition towards the system with higher engagement denoting frequently a more positive stance. Both trends are captured by attributes $c_{1,k}$ to $c_{4,k}$ in Table 3.

In brief, the proposed metrics will be built on the three attribute matrices $\mathbf{M}_{F,i}^m$, $\mathbf{M}_{T,i}^m$, and $\mathbf{M}_{C,i}^m$ which respectively reflect friendship, topical, and behavioral patterns for the i-th user and are defined in Eqs. (17), (20), and (22). In order to compute the likeness between the i-th and j-th system user the following steps are taken:

- The attribute matrices of the j-th user are subtracted from the corresponding of the i-th to create a difference term. Each such term contains divergences in the respective patterns which in turn contribute to user distance. This emphasis on locality is intentional so as to build an as accurate as possible user profile.
- The hyperparameter m as shown in Eqs. (17), (20), and (22) contains by definition for the i-th user the number of users (besides the i-th one) necessary in order to determine the size of the attribute matrices. Essentially m represents a tradeoff between pattern locality degree and computational complexity.
- These terms are respectively weighted by matrices $\mathbf{W}_{F,i}^m$, $\mathbf{W}_{T,i}^m$, and $\mathbf{W}_{C,i}^m$ normalizing values so that they have a maximum of 1. This serves fairness as there is no a priori reason to place more importance in any attribute catrgory. Moreover, keeping the numerical values confined increases numerical stability.
- A third order tensor $\mathbf{T}_{i,j}$ is constructed by having as layers the results of the preceding step as shown in (24). In this way the results remain essentially uncoupled up to this point. This allows for a number of independent tensor functions to be applied to $\mathbf{T}_{i,j}$ simultaneously in order to yield pairwise user distances.
- The way the attribute and weight matrices referenced above, or any other such pairs that may be added in future extensions to $\mathbf{T}_{i,j}$ for that matter, are combined primarily depends on their form and properties. In this way any underlying domain semantics not only are obeyed but are also explicitly expressed.

In Fig. 3 an overview of the above process is shown. From the definition of the proposed tensor distance metric it can be seen that it has the following advantages:

- It can be directly extended to any number of heterogenous features by adding the appropriate attribute and weight matrices. In turn these will become additional tensor layers. This strategy keeps tensor dimensionality low while it remains capable of incorporating an arbitrary number of features.
- Each tensor layer is weighted independently so that a wide array of layer-specific semantics can be represented by any of the weight matrices $\mathbf{W}_F^m$, $\mathbf{W}_T^m$, and $\mathbf{W}_C^m$, as it happens in fact in this work. Therefore properly

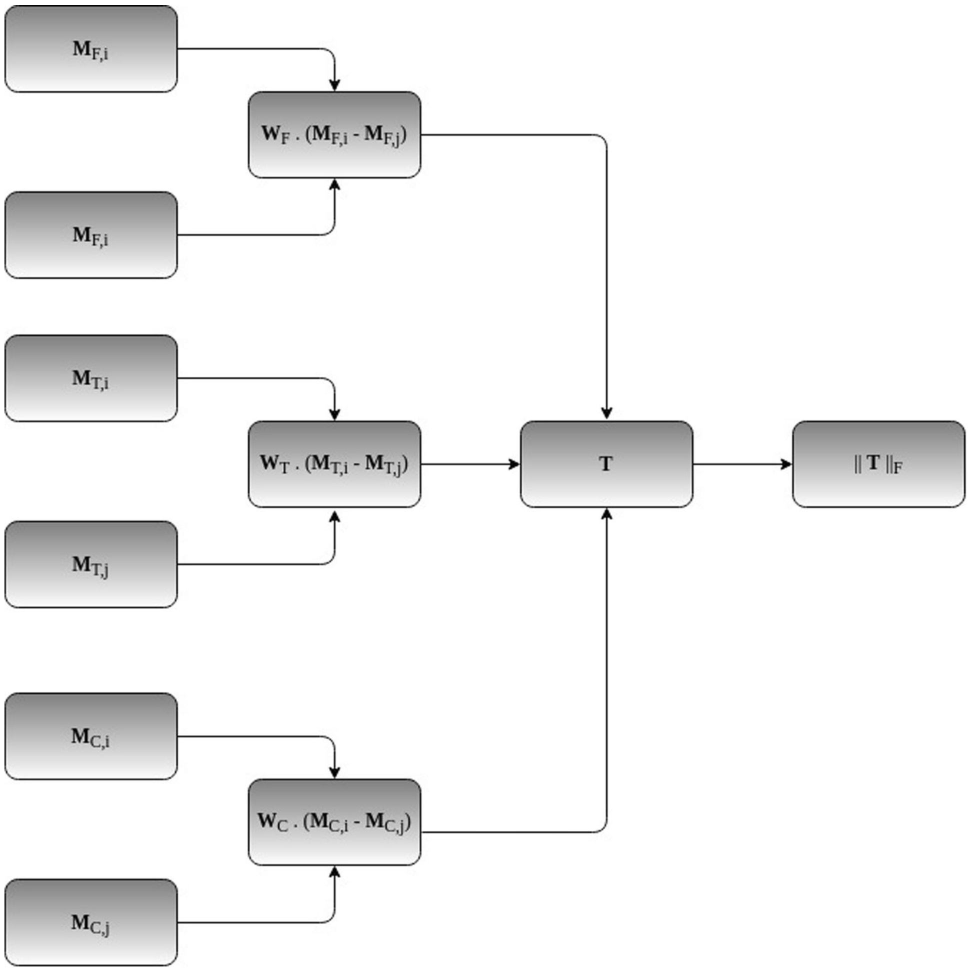
M_{F,i}
W_F . (M_{F,i} - M_{F,j})
M_{F,i}
M_{T,i}
W_T . (M_{T,i} - M_{T,j})
M_{T,j}
T
|| T ||_F
M_{C,i}
W_C . (M_{C,i} - M_{C,j})
M_{C,j}

Fig. 3 Tensor metric overview

encoded domain knowledge can play a role in determining pairwise user distances.

- Furthermore, it is possible for certain choices of weight matrices to determine or control events at the user level. This is useful for ad hoc tests such as assessing the effect of an individual user, for discovering outliers in the user base or to enforce system policy to inactive, blocked, or even suspended users.
- Moreover, as mentioned earlier, any tensor function can be applied to $\mathbf{T}_{i,j}$ in order to tailor the user distance metric to the needs of any underlying domain. To this end it is critical that layers remain uncoupled until tensor formation so that any combination thereof will be the explicit result of the applied function.

Therefore, in view of the above properties it can be argued that Fig. 3 essentially represents an entire tensor framework for generating tailored and multidimentional user distance metrics and not merely a single such metric.

4.2 Friendship attribute matrix

The first attribute matrix $\mathbf{M}_F^m$ is determined from the user friendship map. Specifically, the friends of the i-th user are ranked according to the number of the friendships between them and the m top ones are selected. This computes social coherency in the immediate vicinity of the user in question based on the number of local triangles, an essential community element as explained among others in [9]. Thus $\mathbf{M}_{F,i}^m$ is constructed form the connectivity patterns of the i-th user followed by those of the m top friends in descending order. Any occurring ties are resolved randomly. The entries are as shown in Eq. (17), therefore ensuring that major patterns are close to the diagonal.

$$\mathbf{M}_{F,i}^m[u,v] \triangleq \begin{cases} 1, & u = v \text{ or } u \text{ is a friend of } v \\ 0, & \text{otherwise} \end{cases} \in \{0,1\}^{(m+1)\times(m+1)} \tag{17}$$

It should also be highlighted that $\mathbf{M}_{F,i}^m$ is built to focus on the interaction of the local triangles irrespective of the system users involved. Therefore, this attribute matrix contains patterns which also act as a mask with any differences in this mask resulting in distance penalties. Taking into consideration the users would disrupt this as locality would disappear from users with dissimilar social environments. Moreover recall that triangles are unique in the sense that they are the elementary cliques where the size of vertices equals the size of edges. Despite their small size, triangles are crucial in maintaining structural coherency and modularity in large graphs.

The corresponding weight matrix $\mathbf{W}^m_{F,i}$ is shown in Eq. (18). It is defined as the Laplacian of $\mathbf{M}^m_{F,i}$, normalized such that its maximum element is 1. In this context $f_{j,i}$ is the number of friends the j-th user has in $\mathbf{M}^m_{F,i}$ and not the total number of friends it has in general as $\mathbf{W}^m_{F,i}$ is relative to the i-th user.

$$\begin{aligned}\mathbf{W}^m_{F,i} &\triangleq \left(\mathbf{I}_{m+1} - \mathbf{D}_i^{-1}\mathbf{M}^m_{F,i}\right) \odot 1/\max\left[\left|\left(\mathbf{I}_{m+1} - \mathbf{D}_i^{-1}\mathbf{M}^m_{F,i}\right)\right|\right] \\ \mathbf{D}_i &\triangleq \operatorname{diag}\left(1, f_{i_1,i}, \ldots, f_{i_m,1}\right)\end{aligned} \tag{18}$$

In Eq. (18) the indices i_k refer to the m top friends of the i-th user as specified earlier. Additionally, in contrast to the standard matrix multiplication the Hadamard product works between a matrix and a scalar, whereas the maximum is computed among all entries of the localized Laplacian of $\mathbf{M}^m_{F,i}$. The rationale behind selecting the latter as the weight of $\mathbf{M}^m_{F,i}$ is that a major property of graph Laplacians are inherently connected with the partitioning of the corresponding graph [22]. Since $\mathbf{M}^m_{F,i}$ is a local third order approximation of the overall graph adjacency matrix, an equally localized Laplacian is a suitable weight selection.

4.3 Topic attribute matrix

The attribute matrix $\mathbf{M}^m_{T,k}$ contains topical coherence patterns and its entries are computed as follows. As the corresponding list s_i of Table 3 is progressively populated with the cultural items retrieved from the system, the topic profile of this user becomes more accurate. The Tanimoto similarity coefficient is defined as shown in Eq. (19) and it is a symmetric measure of the coherence between any two sets V_1 and V_2 in terms of their respective elements.

$$\tau(V_1, V_2) \triangleq \frac{|V_1 \cap V_2|}{|V_1 \cup V_2|} = \frac{|V_1 \cap V_2|}{|V_1| + |V_2| - |V_1 \cap V_2|} \tag{19}$$

In (19) the second form follows directly from the Venn diagram and the properties of elementary set theoretic operations. For large sets cardinality estimators are used.

For the i-th user first the Tanimoto coefficient between the features s_i and s_j of Table 3 for any other connected user is computed. Along a similar line of reasoning with before the m top most similar users to the i-th one are selected and are placed in the same order as the attribute matrix $\mathbf{M}^m_{F,i}$ since the focus is on patterns and not on individual users. The elements of $\mathbf{M}^m_{T,i}$ have the values shown in Eq. (20).

$$\mathbf{M}^m_{T,i}[u,v] \triangleq \begin{cases} 1, & u = v \\ \dfrac{2}{\dfrac{1}{\tau(s_i, s_u)} + \dfrac{1}{\tau(s_u, s_v)}}, & u \neq v \end{cases} \in [0,1]^{(m+1)\times(m+1)} \tag{20}$$

The intuition behind Eq. (20) relies on a modified and smoothed version of the triangle inequality. The latter has the tendency to reinforce local patterns creating therefore stable areas around the u-th user. Moreover, the harmonic version smoothes out any outliers while it is tolerant to numerically small or even zero values. Since by definition the values of $\mathbf{M}^m_{T,i}$ already belong to the standard interval, there is no need to impose a weight on them. Therefore the elements of $\mathbf{W}^m_{T,i}$ are identically one.

4.4 Behavioral attribute matrix

Determining whether a specific user is inclined to frequently use the system is done implicitly through a combination of some of the features of Table 3. The fundamental underlying assumption is that user satisfaction degree is proportional to the system utilization frequency, as content users have a tendency to return as stated among others in [61]. In other words, it suffices that the user retention rate be estimated. Specifically, given the available features the latter can be reflected in the total query length, expressed as the total query terms, divided by the total session duration every month. The former is approximated by the product of the average keywords per query $c_{3,i}$ with the average number of queries per session $c_{4,i}$, whereas the latter by the product of average number of sessions per month $c_{1,i}$ with the average session duration $c_{2,i}$. The intentional inclusion of $c_{2,i}$ ensures that between users with comparable online behavior those with more idle system time are penalized. Given the above user similarity is then expressed as in Eq. (21):

$$\eta_{i,j} \triangleq \frac{2}{\dfrac{c_{1,i}c_{2,i}}{c_{3,i}c_{4,i}} + \dfrac{c_{1,j}c_{2,j}}{c_{3,j}c_{4,j}}} \tag{21}$$

In the preceding equation the harmonic mean is selected for the reasons stated in the formulation of $\mathbf{M}^m_{T,i}$. With these definitions, the elements of $\mathbf{M}^m_{C,i}$ are formed again based on the triangle inequality based on the m most similar users to the i-th one as determined by Eq. (21). Since the similarity between the users involved therein has already been determined by the use of harmonic mean, a simple arithmetic mean suffices as shown in Eq. (22) as using two nested harmonic means would defeat the original purpose. The order of the users, but not necessarily the users themselves as the focus is on patterns rather than on actual users, is the same with those of $\mathbf{M}^m_{F,i}$.

$$\mathbf{M}^m_{C,i}[u,v] \triangleq \begin{cases} \eta_{u,u}, & u = v \\ \frac{1}{2}\left(\eta_{i,u} + \eta_{u,v}\right), & u \neq v \end{cases} \in \mathbb{R}^{(m+1)\times(m+1)} \tag{22}$$

The diagonal entries of $\mathbf{M}^m_{C,i}$ in Eq. (22) are the quantities representing the respective behavioral user activity. Notice that unlike $\mathbf{M}^m_{T,i}$ the values of $\mathbf{M}^m_{C,i}$ do not belong to the standard interval and consequently a normalization scheme is necessary in order to maintain compatibility with the other attribute matrices. To this end the $\mathbf{W}^m_{C,i}$ is selected in a fashion similar to that of Eq. (18).

4.5 Proposed tensor metric

Given the preceding definitions of the three attribute matrices and their respective weights, the i-th user can be represented interally in the recommender system as the following ordered triplet of Eq. (23). Notice that none of the tuple elements is a distance matrix by itself but rather a pattern-based user encoding which can serve as the building block for one. A distance metric $d(\cdot,\cdot)$, including the proposed one, can determine the distance between any two users in the system user base.

$$a^m_i \triangleq \left(\mathbf{M}^m_{F,i}, \mathbf{M}^m_{T,i}, \mathbf{M}^m_{C,i}\right) \tag{23}$$

The proposed distance metrics have the parameterized form of Eq. (25) which in turn relies on the third order tensor $\mathbf{T}_{i,j}$ of Eq. (24). Said tensor $\mathbf{T}_{i,j}$ for any two points a_i and a_j is constructed by stacking levels consisting of the weighted differences of the respective attribute matrices $\mathbf{M}^m_{F,i}$, $\mathbf{M}^m_{T,i}$, and $\mathbf{M}^m_{C,i}$.

$$\begin{aligned} \mathbf{T}_{i,j}[:,:,1] \triangleq \mathbf{B}_F &= \mathbf{W}^m_{F,i} \odot \mathbf{M}^m_{F,i} - \mathbf{W}^m_{F,j} \odot \mathbf{M}^m_{F,j} \\ \mathbf{T}_{i,j}[:,:,2] \triangleq \mathbf{B}_T &= \mathbf{W}^m_{T,i} \odot \mathbf{M}^m_{T,i} - \mathbf{W}^m_{T,j} \odot \mathbf{M}^m_{T,j} \quad \in [0,1]^{(m+1)\times(m+1)\times 3} \\ \mathbf{T}_{i,j}[:,:,3] \triangleq \mathbf{B}_C &= \mathbf{W}^m_{C,i} \odot \mathbf{M}^m_{C,i} - \mathbf{W}^m_{C,j} \odot \mathbf{M}^m_{C,j} \end{aligned} \tag{24}$$

At this point it should be highlighted that the Hadamard products between the three attribute matrices and their corrsponding weights in (24) are not panacea. Instead these products rely heavily on the forms and semantics of their respective operands. For instance, if the weight matrices represented filtering or a linear transform, then ordinary matrix multiplication may well have been a better choice.

From its definition, it is clear that $\mathbf{T}_{i,j}$ codifies pattern differences between the i-th and j-th user. The distance between them is computed by the metric of (25):

$$\begin{aligned} d\left(a_i, a_j\right) &\triangleq \frac{\left\|\mathbf{T}_{i,j}\right\|_F}{3(m+1)^2} \\ &= \frac{1}{3(m+1)^2}\sqrt{\operatorname{tr}\left(\mathbf{B}_F^T\mathbf{B}_F\right) + \operatorname{tr}\left(\mathbf{B}_T^T\mathbf{B}_T\right) + \operatorname{tr}\left(\mathbf{B}_C^T\mathbf{B}_C\right)} \end{aligned} \tag{25}$$

Almost by construction the proposed metric of Eq. (25) satisfies the following three fundamental distance metric criteria as explained below:

- When the metric equals zero, then for all practical intents and purposes of the content delivery system users i and j exhibit the same digital behavior.
- The metric is symmetric by construction and therefore for any distinct user pair it has to be computed only once, reducing the total partitioning complexity.
- Finally and also by construction the tensor Frobenius norm satisfies the triangle inequality for tensors $\mathbf{T}_1$ and $\mathbf{T}_2$ and their sum $\mathbf{T}_1 + \mathbf{T}_2$ as shown in (26):

$$\left\|\mathbf{T}_1 + \mathbf{T}_2\right\|_F \leq \left\|\mathbf{T}_1\right\|_F + \left\|\mathbf{T}_2\right\|_F \tag{26}$$

The Frobenius norm is a measure of the root mean square of the tensor elements, which can be naturally interpreted as the energy fluctuation or the divergence of the local patterns of the i-th and j-th user. Also it is differentiable, which can readily lend itself to many gradient-based optimization algorithms. The only hard constraint in this case is that the final values should be mapped to the standard interval.

In order to assess the effect of the behavioral attributes a tensor $\mathbf{T}'_{i,j}$ consisting of only the first two layers of $\mathbf{T}_{i,j}$ was also used with identical parameters. In this case the SOM operates as a conventional recommender system since it relies exclusively on the social patterns of the system users and their respective preferences. This will be the building block of the distance metric d' defined in a similar manner with the one in Eq. (25). In turn d' will be one of the baselines used in Sect. 5.

4.6 Selecting hyperparameter *m*

As mentioned earlier the hyperparameter m balances the locality degree of the patterns stored in the three aforementioned attribute matrices with the computational complexity. Specifically, the lower m is the more prevalent local patterns become and the easier their computation are at the expense of the approximation accuracy. Especially since for the particular feature selection a third order approximation is employed, this should be balanced with a sufficient number of local triangles.

In this subsection a strategy based on the properties of user social interaction is used. Specifically since human social interaction tends to yield power law graphs [9], the average degree, namely the average number of friendship connections formed, is generally considered a global indicator describing graph robustness and expansion potential. Said degree is defined as in (27):

$$m^* \triangleq \left\lceil \frac{1}{|i|} \sum_i \deg(i) \right\rceil \tag{27}$$

In Eq. (27) the user indicator i ranges over the users registered in the system base. The unbiased version of the deterministic variance of m is given in (28) and it is used to naturally partition the scale of m. The smaller σ_m is, the more concentrated the graph degrees are around m. Power laws tend to have a relatively low average degree as there is a sizeable number of low degree vertices.

$$\sigma_m^* \triangleq \left\lceil \left(\frac{1}{|i|-1} \sum_i (\deg(i) - m^*)^2 \right)^{\frac{1}{2}} \right\rceil \tag{28}$$

As is the case with any engineering applications, the strategy presented here is by no means unique and can be replaced by any selection rule or heuristic for m. Nevertheless, the proposed strategy requires neither thresholds nor the values of other hyperparameters. Instead, it arises naturally from the data themselves, which in the general case is a desired approach in unsupervised learning contexts. An alternative would be that attribute matrix size $m+1$ be expressed as a percentage of the total system users. However, this may not be in accordance with the underlying user base.

5 Results

5.1 Experimental setup

The experimental setup for this work is shown in detail in Table 4. Therein the parameters used in the experiments and their associated values are listed. The definitions as well as the meaning of these parameters can be found in the preceding sections.

Each SOM configuration c contains every parameter necessary to create a working model and it is represented as a tuple with the structure of Eq. (29). For brevity, configurations will be referred to with the corresponding number of Table 5.

$$(\text{parameter } m, \text{distance metric}, \text{learning rate}) \tag{29}$$

In order to evaluate the performance of the proposed SOM architecture two similar architectures based on the same SOM grid structure but relying on different properties through their respective metric distances will serve as benchmarks as follows:

- The infinity norm indicated as ℓ_∞ in Table 5 similarly to d relies also on local patterns but focuses on the maximum divergence found among them.
- The cosine similarity metric, reflected to become a distance metric, indicated by ℓ_c in Table 5 relies on local angles between data points in the feature space.
- Additionally, configurations using the d' distance metric have exactly the same structure with the proposed one but exclude behavioral attributes.

Since each baseline architecture relies on an SOM grid of identical dimension, any major differences in clustering will be shown. Moreover, using algorithms from the SOM category resolves the problem of fine tuning any hyperparameters of the algorithms coming from other categories. Both baseline architecture use the same value for the hyperparameter m. Although this value may not be optimal for each baseline, it serves fairness in the sense that each clustering algorithm has the same resources in terms of local patterns to rely on.

5.2 Configuration scoring method

The performance of each SOM configuration c is evaluated based on a scoring method factoring in a multitude of individual performance metrics pertaining to either the computational complexity of each configuration or to its clustering performance.

Since the SOM training procedure is iterative, the total complexity can be evaluated by considering the total number of epochs. The latter excludes the cost attributes to moving data across the local memory hierarchy since there is sufficient memory to contain both the SOM and the training data. Moreover, the cost to compute the two distance metrics is comparable for the same value of the parameter m. The number of epochs were obtained by averaging the R_0 runs. Finally, the local system load ρ was very low and, therefore, there was no computational burden from other applications.

The scoring method J shown in Eq. (30) combines the n_e available metric of Table 6 by adding the absolute relative distance for the k-th such metric between the raw score e_k attained by c and the respective optimal one e_k^*. This distance of e_k^* from itself is zero. By ignoring the error sign in (30), this distance shows how far e_k lies from e_k^* regardless of whether the k-th metric is to be maximized or

Table 5 SOM configurations

Id	Configuration	Id	Configuration	Id	Configuration	Id	Configuration
1	$(m^* - 2\sigma_m^*, d, \cos)$	6	$(m^* - 2\sigma_m^*, d, \text{lin})$	11	$(m^* - 2\sigma_m^*, d', \cos)$	16	$(m^* - 2\sigma_m^*, d', \text{lin})$
2	$(m^* - \sigma_m^*, d, \cos)$	7	$(m^* - \sigma_m^*, d, \text{lin})$	12	$(m^* - \sigma_m^*, d', \cos)$	17	$(m^* - \sigma_m^*, d', \text{lin})$
3	$(m^*, d, \cos)$	8	(m^*, d, lin)	13	$(m^*, d', \cos)$	18	(m^*, d', lin)
4	$(m^* + \sigma_m^*, d, \cos)$	9	$(m^* + \sigma_m^*, d, \text{lin})$	14	$(m^* + \sigma_m^*, d', \cos)$	19	$(m^* + \sigma_m^*, d', \text{lin})$
5	$(m^* + 2\sigma_m^*, d, \cos)$	10	$(m^* + 2\sigma_m^*, d, \text{lin})$	15	$(m^* + 2\sigma_m^*, d', \cos)$	20	$(m^* + 2\sigma_m^*, d', \text{lin})$
21	$(m^* - 2\sigma_m^*, \ell_\infty, \cos)$	26	$(m^* - 2\sigma_m^*, \ell_\infty, \text{lin})$	31	$(m^* - 2\sigma_m^*, \ell_c, \cos)$	36	$(m^* - 2\sigma_m^*, \ell_c, \text{lin})$
22	$(m^* - \sigma_m^*, \ell_\infty, \cos)$	27	$(m^* - \sigma_m^*, \ell_\infty, \text{lin})$	32	$(m^* - \sigma_m^*, \ell_c, \cos)$	37	$(m^* - \sigma_m^*, \ell_c, \text{lin})$
23	$(m^*, \ell_\infty, \cos)$	28	$(m^*, \ell_\infty, \text{lin})$	33	$(m^*, \ell_c, \cos)$	38	$(m^*, \ell_c, \text{lin})$
24	$(m^* + \sigma_m^*, \ell_\infty, \cos)$	29	$(m^* + \sigma_m^*, \ell_\infty, \text{lin})$	34	$(m^* + \sigma_m^*, \ell_c, \cos)$	39	$(m^* + \sigma_m^*, \ell_c, \text{lin})$
25	$(m^* + 2\sigma_m^*, \ell_\infty, \cos)$	30	$(m^* + 2\sigma_m^*, \ell_\infty, \text{lin})$	35	$(m^* + 2\sigma_m^*, \ell_c, \cos)$	40	$(m^* + 2\sigma_m^*, \ell_c, \text{lin})$

minimized. By its very definition the higher J is, the worse a configuration is doing.

$$J \triangleq \frac{1}{n_e} \sum_{k=1}^{n_e} \left| \frac{e_k - e_k^*}{e_k^*} \right| \tag{30}$$

The individual metrics comprising the evaluation metric J as well as their respective meaning and whether their maximization or minimization is sought are shown in Table 6. These metrics are defined and explained in the remainder of this section.

This maximization allows for understanding how each SOM configuration c over- or even underperforms relative to others in a cumulative sense. Since the scoring method J for c comprises of a number of individual metrics, an overall assessment for c is obtained, whereas the components allow a finer granularity overview of c. This can serve as the building block of a broader SOM framework as follows:

- When a pool Q of SOM configurations is available, then a better configuration, in terms of approximating the underlying manifold D of Sect. 3, combining individual elements out of those already in Q can be generated.
- Alternatively the available configurations in Q can be adaptively selected as more information about D are obtained or even when D evolves. In this case the configurations themselves need not evolve or be otherwise modified.

Table 7 contains in descending order the best SOM configurations according to J. Besides the total score of the latter the results of the individual metrics comprising J are given in order to provide a more detailed view of the configuration performance. Plase note that the results are rounded to two decimal digits.

From the entries of Table 7 the following can be said:

- The configurations based on d frequently achieve fewer epochs in comparions to those based on d'. This implies that the inclusion of behavioral attributes adds more power to the clustering process which is exploited by distance metric d.

Table 6 Individual metrics

Metric	Meaning	Optimization	Definition
E_s	Difference in communities from the t-sne	Minimization	Eq. (36)
E_b	Number of epochs with bias	Minimization	Table 6
E_{nb}	Number of epochs with no bias	Minimization	Table 6
E_c	Difference in communities from the t-sne	Minimization	Eq. (38)
E_t	Topological error rate	Minimization	Eq. (39)
E_q	Quantization error	Minimization	Eq. (40)
E_d	Average correction decay	Maximization	Eq. (42)
E_h	Centroid change	Minimization	Eq. (44)
E_i	Average intercluster distance	Minimization	Eq. (45)
$n_e = 9$			

- The configurations based on the cosine decay rate also tend to require lower epochs. This can be attributed to its adaptive nature, since it has a relatively high value at the beginning but it drops with a quadratic rate. On the contrary, the inverse linear decay makes the initial cluster formation more difficult.
- Increasing the hyperparameter m results in the number epochs being lowered up to a point. After that, they increase again. This may indicate that extracting features from a small group of system users suffices for locality-based distance metrics. Moreover, this demonstrates the benefits of aligining parameters with the dataset.

5.3 Dataset exploration

Before describing the error metrics of Table 6, whether general ones or tailored for SOM architectures, the perfromance metric E_s will be analyzed. The latter is derived from the t-stochastic network embedding (t-SNE) data exploration and visualization technique [36]. t-SNE works in two steps as follows:

- For a set $\{x_k\}$ of n_p available p-dimensional data points an empirical reference dataset distribution is constructed through the softmax function as shown in (31):

$$p_{j|i} \triangleq \frac{\exp\left(-\frac{\left\|\mathbf{x}_i - \mathbf{x}_j\right\|_2^2}{2\sigma_i^2}\right)}{\sum_{k=1,k\neq i}^{n_p} \exp\left(-\frac{\left\|\mathbf{x}_i - \mathbf{x}_j\right\|_2^2}{2\sigma_i^2}\right)} \tag{31}$$

The above is proportional to the conditional probabilty that $\mathbf{x}_j$ is selected as a neighbor of $\mathbf{x}_i$ if the true dataset distribution were a multidimensional Gaussian distribution with vector mean $\mathbf{x}_i$ and covariance matrix $\sigma_p^2 \mathbf{I}_p$. The reference distribution is finally created by defining $p_{i,j}$ as in Eq. (32):

$$p_{i,j} \triangleq \frac{p_{j|i} + p_{i|j}}{2n_p} \tag{32}$$

- The t-SNE implicitly maps $\{\mathbf{x}_k\}$ to the lower dimension set of equal cardinality $\{\mathbf{y}_k\}$ by defininig the similarity metric shown in Eq. (33):

$$q_{i,j} \triangleq \frac{\left(1 + \left\|\mathbf{y}_i - \mathbf{y}_j\right\|_2\right)^{-1}}{\sum_{u=1}^{n_p} \sum_{v \neq u} \left(1 + \|\mathbf{y}_u - \mathbf{y}_v\|_2\right)^{-1}} \tag{33}$$

The above simialrity metric is a multidimensional Student t-distribution with one degree of freedom which is actually a multidimansional Cauchy distribution. The latter in a q-dimensional space eventually reduces to the form of Eq. (35):

$$f_X(x) \triangleq \frac{1}{\pi\left(1 + \|\mathbf{q}\|_2^2\right)}, \quad \mathbf{q} \in \mathbb{R}^q \tag{34}$$

The new low-dimensional vectors $\{\mathbf{y}_k\}$ are the solution of the non-linear problem of minimizing the Kullback-Leibler distance between the p-th dimensional and the q-th dimensional distributions as formulated in Eq. (35):

$$\langle p || q \rangle \triangleq \sum_{i \neq j} p_{i,j} \log\left(\frac{p_{i,j}}{q_{i,j}}\right) \tag{35}$$

Based on the above decription the t-SNE performs a non-linear dimensionality reduction. This can be used to define metric E_s as shown in Eq. (36):

$$E_s \triangleq \sum_{(C_k, C_k')} \left(1 - \tau(C_k, C_k')\right) + n_t \tag{36}$$

In Eq. (36) the similarity between the clustering $\{C_k\}$ obtained by any of the SOM configurations of Table 5 and the corresponding partitoning obtained by the t-SNE $\{C_k'\}$. First, the pairwise cluster similarity as determinined by the Tanimoto coefficient applied on the number of data points any two clusters C_k and C_k' have in common. Once the most similar clusters are found in both partitionings, the sum of their distances (and not of their similarities) plus the

Table 7 Top scoring configurations

Id	J	E_s	E_b	E_{nb}	E_c	E_t	E_q	E_y	E_h	E_i
3	0.01	0.00	0.00	0.08	0.00	0.00	0.00	0.00	0.00	0.00
2	0.16	0.11	0.16	0.13	0.17	0.13	0.21	0.18	0.19	0.17
8	0.25	0.22	0.19	0.25	0.33	0.22	0.26	0.21	0.28	0.28
4	0.34	0.29	0.28	0.33	0.37	0.31	0.36	0.25	0.37	0.44
7	0.39	0.42	0.30	0.31	0.42	0.34	0.47	0.35	0.39	0.47

number of any unmatched clusters n_t is computed. In this way Eq. (36) has a penalty not only for not properly matching clusters but also for the case when thw two partitioning schemes result in unblananced datapoint clustering. The inspiration for the penalty term was the respective terms of the Akaike information criterion (AIC) [24] and of the Bayes information criterion (BIC) [56].

5.4 Cluster probability distribution

There are many ways to evaluate the size of the clusters in the final cognitive map. One approach it is to evaluate the distribution of the cluster occurences. For a cluster with a centroid u its probability is approximated by Eq. (37).

$$\text{prob}\left\{C_u\right\} \triangleq \frac{\left|\mathbf{d}\right| \text{ invloc}\left(\mathbf{d}\right) \in \text{prox}\left(u\right)\right|}{n} \tag{37}$$

Once the empirical distribution $\tilde{f}$ of the cluster occurences is obtained at the end of the SOM training process, its quality is evaluated. The approach proposed here is to compare its Kullback-Leibler divergence from a reference distribution f^* reflecting the true user preferences. The uniform distribution will be used for f^* since it has the maximum entropy among all distributions with the same number of events. A robust partitioning of the system user base is expected to have lower entropy and, thus, to be further from f^*. The divergence of $\tilde{f}$ from f^* takes the special form of (38):

$$E_c \triangleq \left\langle \tilde{f} \| f^* \right\rangle = \sum_{C_u} \text{prob}\left\{C_u\right\} \log\left(n \, \text{prob}\left\{C_u\right\}\right) \tag{38}$$

From the entries of Table 7 the following can be said.

- The SOM configurations based on d systematically achieve largest divergences from f^*, meaning that they result in more robust distributions. Therefore, the inclusion of behavioral attributes improves partitioning quality.
- The learning rate has a minimal effect on E_c.
- The pattern for m reported in the preceding subsection also appears.

5.5 Topological error

The topological error E_t is perhaps the most important figure of merit for an SOM since it has been designed explicitly for this class of neural networks. Specifically, it is the percentage of the data points which have been assigned to a neuron in a cluster which is not its respective centroid and does not belong to its neighborhood. Thus, it quantifies the overall probability of a data point be assigned to a cluster periphery. It is defined as in Eq. (39), where the sum ranges over all centroids u.

$$E_t \triangleq \frac{1}{\left|\{u\}\right|} \sum_u \frac{\left|\left\{\mathbf{d}\right| \text{ invloc}\left(\mathbf{d}\right) = v \wedge v \in \text{ prox}\left(u; \xi_0\right) \setminus \text{neighb}(u)\right\}\right|}{\left|\left\{\mathbf{d}\right| \text{ invloc}\left(\mathbf{d}\right) = v \wedge v \in \text{ prox}\left(u; \xi_0\right)\right\}\right|} \tag{39}$$

From the entries of Table 7 it can be seen that the column of E_t has patterns similar to these of other columns. This seems to confirm the role of the behavioral attributes.

5.6 Quantization error

The quantization error E_q is similar to the topological error since it is also tied to the cluster geometry. However, it operates on $\mathscr{V}$ instead of $\mathscr{C}$ by computing the average distance of each data point from the synaptic weights of the respective centroid. In Eq. (40) the sum ranges over all cluster centroids u.

$$E_q \triangleq \frac{1}{\left|\{u\}\right|} \sum_u \frac{d\left(\mathbf{d}, \text{ weight}\left(u\right)\right)}{\left|\mathbf{d}\right| \text{ invloc}\left(\mathbf{d}\right) \in \text{ prox}\left(u; \xi_0\right)\right|} \tag{40}$$

The quantization error E_q evaluates the disperse of the data points within a cluster and how well the synaptic weights of the centroids are placed. From the entries of Table 7 it can be seen that the SOM configurations based on the tensor distance metric d outperform, even sometimes by a small margin, those based on d'.

5.7 Correction decay

Since during each epoch the n data points are used to update the synaptic weights of the cluster centroids, it makes sense to measure how much the average change evolves as the epochs progress. This is an indirect way to evaluate clustering, since a drop in the average correction of the synaptic weights indicates a stabilization of the partitioning process. Assuming there are N epochs in total ranging from T_1 to T_N, then the average change to the synaptic weights of the winning neurons u during epoch T_k can be computed by Eq. (41):

$$\sum_{u,t \in T_k} \left\| \text{ weight}\left(u\right) - \mathbf{d}^{[t]} \right\|_2 \tag{41}$$

In Eq. (41) the sum ranges over the iterations t of epoch T_k and over the respective winning neurons u. The Euclidean norm in a space of finite dimensionality is equivalent up to a constant to the Chebyshev and the infinity norms. Thus, the norm selection does not influence the order of magnitude of (41). The average correction decay E_d over the N epochs is defined as in Eq. (42):

$$E_d \triangleq \frac{\sum_{u,t\in T_1}\left\| \text{weight}\,(u) - \mathbf{d}^{[t]}\right\|_2 - \sum_{u',t'\in T_N}\left\| \text{weight}\,(u') - \mathbf{d}^{[t']}\right\|_2}{N} \tag{42}$$

From the entries of Table 7 it can be said that the SOM configurations based on the tensor distance metric d have a steeper descent than those based on d', resulting thus in higher values of E_d. Moreover, the cosine decay rate systematically results in quicker correction decay. These observations combined with the fewer number of epochs indicate a more efficient partitioning process. On the other hand, the configurations based on linear decay achieve a less sharper decay, revealing a slow cluster formulation. These findings are in accordance with those obtained from other metrics.

5.8 Centroid change

The error metric of centroid change E_h as its name suggests focuses on the centroid locations in order to evaluate clustering quality. Let Δ be the set of the cluster centroids along with their neighborhood as shown in Eq. (43).

$$\Delta \triangleq \bigcup_u \{u \cup \{v | \, v \in \text{neighb}(u)\}\} \tag{43}$$

Let $\{u'\}$ be the centroids obtained by the next epoch. Then E_h is the average probability that a centroid is moved as shown in Eq. (44). In this equation the first sum ranges over $N-1$ transitions between the N epochs. This specific form has the advantage it does not depend on hyperparameters.

$$E_h \triangleq \frac{1}{N-1}\sum_T \frac{|\{u' \setminus \Delta\}|}{|u'|} \tag{44}$$

From the entries of Table 7 it follows that SOM configurations based on d consistently outperform their counterparts based on d'. This is in accordance with the results obtained from the other metrics discussed in this work.

5.9 Average intercluster distance

The final performance metric presented here is the average intercluster distance E_i which approximates the radius of the polygon defined by cluster centroids by the distances between centroids in $\mathscr{C}$ as shown in Eq. (45). In there C_0 is the number of clusters, while the sum ranges over all possible distinct cluster pairs.

$$E_i \triangleq \frac{2}{C_0(C_0-1)}\sum_{(C_u,C_v)} g(u,v) \tag{45}$$

From the entries of Table 7 it can be seen that the SOM configurations based on d tend to yield a larger distance between clusters, implying that they are better separated. Alternatively, it can be interpreted as the respective polygons have a longer radius, utilizing thus more of the available neurons.

5.10 Recommendations

The experimental results presented earlier are in accordance, to the extent a comparison can be made, with the recent relevant scientific literature pertaining to behavioral recommender systems. Specifically, [40] argues in favor of including behavioral attributes to recommendation engines and presents certain architectures for doing so. Also [41] describes how behavioral features can be extracted with a crowdsourcing scheme in order to generate feedback based on them. High level strategies for including behavioral features tailored for recommendation engines are explored in [3]. The control aspects of distance metrics based on various behavioral attributes are given in [42]. From the above results, the following recommendations can be made:

- The inclusion of behavioral attributes appears to boost clustering performance. This may indicate that users with similar preferences tend to formulate similar queries. Alternatively, user may connect to others with similar interests. In this case, social activity reflects the preference similarity.
- Since behavioral attributes may be the key to partitioning, the content delivery system may well be designed to include many related keywords based on known emotional models. This can assist with clustering, since the distance metric component which takes into account keyword similarity can be more fine grained.
- Allowing users to connect in a content delivery system can allow its recommender engine to effectively mine its respective base. Along a similar line of reasoning, allowing multiple ways on interaction such as custom annotations, message boards, or chats can reveal even more information to be mined.

Tensor distance functions allow the design of flexible distance metrics which have the potential to outperform by far linear or even quadratic ones in problems where the data space has clusters of highly uneven shapes or areas. Moreover, tensor distances may be a viable alternative in scenaria where clusters are represented by multiple centroids. The preceding advantages may come with a cost. Besides the increased memory requirements and the high computational intensity, additional care should be taken to maintain numerical stability in long sequences of operations involving floating point numbers [35]. Moreover, because of the loss of many implicit symmetries when moving from two to more dimensions, many tensor algorithms differ considerably from their matrix counterparts. Perhaps the most representative case is the singular value decomposition (SVD), a key matrix factorization, which takes more than one distinct forms each maintaining some but not all of the properties of the original SVD [21, 54].

6 Conclusions and future work

This work focuses on the partitioning of user the base of cultural content delivery system. The proposed methodology relies heavily on the coupling of a tensor distance metric incorporating digital behavior attributes with self organizing maps. This combination, one of the first of its kind, yields considerable clustering flexibility as well as better clustering quality in terms of a broad spectrum of performance metrics including topological error, quantization error, correction decay, centroid change, and average intercluster distance. These metrics, applied on a benchmark dataset consisting of a variety of attributes, clearly indicate that among the tensor distances used the one relying on behavioral attributes adds a distinct boost to clustering quality. The latter strongly hints at the case that cultural consumers with similar digital behavior have also similar preferences when it comes to selecting cultural content. Although the cause of this potential link is yet to be determined, the aforementioned difference in error rates is considerable enough for most practical purposes. It should be also noted that the proposed methodology is generic enough to be applied to any content delivery system providing the required attributes.

This work can be extended in a number of ways. As an immediate first research step, the advantages as well as the disadvantages of the proposed clustering schemes should be tested against datasets of larger size and higher variability in attributes. Moreover, more error metrics can help in better understanding the potential and limits of tensor distances, especially the flexibility in cluster shapes. Reducing the discontinuities in the regions between clusters can lead to smoother cognitive maps.

Acknowledgements This research has been co-financed by the European Union and Greek national funds through the Competitiveness, Entrepreneurship and Innovation Operational Programme, under the Call “Research – Create – Innovate”, project title: “Development of technologies and methods for cultural inventory data interoperability”, project code: T1EDK-01728, MIS code: 5030954.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Albawi S, Mohammed TA, Al-Zawi S (2017) Understanding of a convolutional neural network. In: ICET, IEEE, pp 1–6
2. Anantharam V, Jog V, Nair C (2019) Unifying the Brascamp-Lieb inequality and the entropy power inequality. In: ISIT, IEEE, pp 1847–1851
3. Angulo C, Falomir Z, Anguita D, Agell N, Cambria E (2020) Bridging cognitive models and recommender systems. Cogn Comput 12(2):426–427
4. Behrens TE, Muller TH, Whittington JC, Mark S, Baram AB, Stachenfeld KL, Kurth-Nelson Z (2018) What is a cognitive map?Organizing knowledge for flexible behavior. Neuron 100(2):490–509
5. Brito LC, da Silva MB, Duarte MAV (2021) Identification of cutting tool wear condition in turning using self-organizing map trained with imbalanced data. J Intell Manuf 32(1):127–140
6. Cao X, Yao J, Xu Z, Meng D (2020) Hyperspectral image classification with convolutional neural network and active learning. IEEE Trans Geosci Remote Sens 58(7):4604–4616
7. Chanhom W, Anutariya C (2019) TOMS: a linked open data system for collaboration and distribution of cultural heritage artifact collections of national museums in Thailand. New Gener Comput 37(4):479–498
8. Dhillon A, Verma GK (2020) Convolutional neural network: a review of models, methodologies and applications to object detection. Prog Artif Intell 9(2):85–112
9. Drakopoulos D, Giotopoulos KC, Giannoukou I, Sioutas S (2020) Unsupervised discovery of semantically aware communities with tensor Kruskal decomposition: a case study in Twitter. In: SMAP, IEEE, https://doi.org/10.1109/SMAP49528.2020.9248469
10. Drakopoulos G, Mylonas P (2020) Evaluating graph resilience with tensor stack networks: a keras implementation. NCAA 32(9):4161–4176. https://doi.org/10.1007/s00521-020-04790-1
11. Drakopoulos G, Giannoukou I, Mylonas P, Sioutas S (2020) On tensor distances for self organizing maps: Clustering cognitive tasks. DEXA, Springer, Lecture Notes in Computer Science 12392:195–210. https://doi.org/10.1007/978-3-030-59051-2_13
12. Drakopoulos G, Giannoukou I, Mylonas P, Sioutas S (2020) The converging triangle of cultural content, cognitive science, and behavioral economics. MHDW, Springer, IFIP Advances in Information and Communication Technology 585:200–212. https://doi.org/10.1007/978-3-030-49190-1_18
13. Drakopoulos G, Voutos Y, Mylonas P (2020) Annotation-assisted clustering of player profiles in cultural games: a case for tensor analytics in Julia. BDCC 4(4):39. https://doi.org/10.3390/bdcc4040039
14. Faigl J, Hollinger GA (2017) Autonomous data collection using a self-organizing map. IEEE Trans Neural Netw Learn Syst 29(5):1703–1715
15. Fan C, Wang J, Gang W, Li S (2019) Assessment of deep recurrent neural network-based strategies for short-term building energy predictions. Appl Energy 236:700–710
16. Frusque G, Jung J, Borgnat P, Gonçalves P (2020) Multiplex network inference with sparse tensor decomposition for functional connectivity. IEEE Trans Signal Inf Process Over Netw 6:316–328
17. Gao C, Neil D, Ceolini E, Liu SC, Delbruck T (2018) DeltaRNN: A power-efficient recurrent neural network accelerator. In: International Symposium on field-programmable gate arrays, ACM, pp 21-30
18. Glevarec H, Nowak R, Mahut D (2020) Tastes of our time: analysing age cohort effects in the contemporary distribution of music tastes. Cult Trends 29(3):182–198

19. Goldfeld Z, Greenewald K, Weed J, Polyanskiy Y (2019) Optimality of the plug-in estimator for differential entropy estimation under Gaussian convolutions. In: ISIT, IEEE, pp 892-896
20. Gu F, Cheung YM (2017) Self-organizing map-based weight design for decomposition-based many-objective evolutionary algorithm. IEEE Trans Evolut Comput 22(2):211–225
21. Hou J, Zhang F, Wang J (2021) One-bit tensor completion via transformed tensor singular value decomposition. Appl Math Model 95:760–782
22. Hu Z, Nie F, Chang W, Hao S, Wang R, Li X (2020) Multi-view spectral clustering via sparse graph learning. Neurocomputing 384:1–10
23. Jain DK, Dubey SB, Choubey RK, Sinhal A, Arjaria SK, Jain A, Wang H (2018) An approach for hyperspectral image classification by optimizing SVM using self organizing map. J Comput Sci 25:252–259
24. Khalid A, Sarwat AI (2021) Unified univariate-neural network models for lithium-ion battery state-of-charge forecasting using minimized akaike information criterion algorithm. IEEE Access 9:39154–39170
25. Khamparia A, Gupta D, Nguyen NG, Khanna A, Pandey B, Tiwari P (2019) Sound classification using convolutional neural network and tensor deep stacking network. IEEE Access 7:7717–7727
26. Khare SK, Bajaj V (2020) Time-frequency representation and convolutional neural network-based emotion recognition. IEEE Trans Neural Netw Learn Syst 32(7):2901–2909
27. Kobayashi Y, Kurokawa S, Ishii T, Wakano JY (2021) Time to extinction of a cultural trait in an overlapping generation model. Theor Popul Biol 137:32–45
28. Kohonen T (1990) The self-organizing map. Proc IEEE 78(9):1464–1480
29. Kong W, Dong ZY, Jia Y, Hill DJ, Xu Y, Zhang Y (2017) Short-term residential load forecasting based on LSTM recurrent neural network. IEEE Trans Smart Grid 10(1):841–851
30. Leyva-Mayorga I, Torre R, Pla V, Pandi S, Nguyen GT, Martinez-Bauset J, Fitzek FH (2020) Network-coded cooperation and multi-connectivity for massive content delivery. IEEE Access 8:15656–15672
31. Li M, Liu S, Zhang Z (2020) Deep tensor fusion network for multimodal ground-based cloud classification in weather station networks. AdHoc Netw 96:101991
32. Li S, Li W, Cook C, Zhu C, Gao Y (2018) Independently recurrent neural network (indrnn): Building a longer and deeper RNN. In: CVPR, IEEE, pp 5457–5466
33. Liang L, Xu J, Deng L, Yan M, Hu X, Zhang Z, Li G, Xie Y (2021) Fast search of the optimal contraction sequence in tensor networks. IEEE J Sel Top Signal Process 15(3):574–586
34. Liu Y, Liu J, Zhu C (2020) Low-rank tensor train coefficient array estimation for tensor-on-tensor regression. IEEE Trans Neural Netw Learn Syst 31(12):5402–5411
35. Liu YY, Zhao XL, Zheng YB, Ma TH, Zhang H (2021) Hyperspectral image restoration by tensor fibered rank constrained optimization and plug-and-play regularization. IEEE Trans Geosci Remote Sens 60:1–17
36. Van der Maaten L, Hinton G (2008) Visualizing data using t-SNE. J Mach Learn Res 9(11):2579–2605
37. Manovich L (2020) Cultural analytics. MIT Press
38. Mohamed A, Qian K, Elhoseiny M, Claudel C (2020) Social-stgcnn: a social spatio-temporal graph convolutional neural network for human trajectory prediction. In: CVPR, pp 14424-14432
39. Nallapati R, Zhai F, Zhou B (2017) Summarunner: a recurrent neural network based sequence model for extractive summarization of documents. In: AAAI, vol 31
40. Nápoles G, Grau I, Salgueiro Y (2020) Recommender system using long-term cognitive networks. Knowl Based Syst 206:106372
41. Nguyen LV, Jung JJ (2020) Crowdsourcing platform for collecting cognitive feedbacks from users: a case study on movie recommender system. In: reliability and statistical computing, Springer, pp 139-150
42. Ramakrishnan P, Balasingam B, Biondi F (2021) Cognitive load estimation for adaptive human-machine system automation. In: Learning Control, Elsevier, pp 35-58
43. Sacha D, Kraus M, Bernard J, Behrisch M, Schreck T, Asano Y, Keim DA (2017) Somflow: guided exploratory cluster analysis with self-organizing maps and analytic provenance. IEEE Trans Vis Comput Graphics 24(1):120–130
44. Sherstinsky A (2020) Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. Phys D Nonlinear Phenom 404:132306132306132306-132306
45. Song HM, Woo J, Kim HK (2020) In-vehicle network intrusion detection using deep convolutional neural network. Veh Commun 21:100198
46. de Sousa MAdA, Pires R, Del-Moral-Hernandez E (2020) SOMprocessor: a high throughput FPGA-based architecture for implementing self-organizing maps and its application to video processing. Neural Netw 125:349–362
47. Spiers HJ (2020) The hippocampal cognitive map: One space or many? Trends Cog Sci 24(3):168–170
48. Sultana F, Sufian A, Dutta P (2020) Evolution of image segmentation using deep convolutional neural network: a survey. Knowl Based Syst 201:106062106062106062
49. Tang TA, Mhamdi L, McLernon D, Zaidi SAR, Ghogho M (2018) Deep recurrent neural network for intrusion detection in SDB-based networks. In: NetSoft, IEEE, pp 202-206
50. Tyrowicz J, Krawczyk M, Hardy W (2020) Friends or foes? A meta-analysis of the relationship between “online piracy” and the sales of cultural goods. Inf Econ Policy 53:100879
51. Vaessen J, Strasen S (2021) A cognitive and cultural reader response theory of character construction. Style Read Response Minds Media Methods 36:81
52. Wilkinson N, Klaes M (2017) An introduction to behavioral economics. Macmillan International Higher Education
53. Wu X, Li Q, Li X, Leung VC, Ching P (2020) Joint long-term cache updating and short-term content delivery in cloud-based small cell networks. IEEE Trans Commun 68(5):3173–3186
54. Xia W, Zhang X, Gao Q, Shu X, Han J, Gao X (2021) Multiview subspace clustering by an enhanced tensor nuclear norm. IEEE Transactions on cybernetics
55. Yang JH, Zhao XL, Ji TY, Ma TH, Huang TZ (2020) Low-rank tensor train for tensor robust principal component analysis. Appl Math Comput 367:124783
56. Yu M, Li Y, Podlubny I, Gong F, Sun Y, Zhang Q, Shang Y, Duan B, Zhang C (2020) Fractional-order modeling of lithium-ion batteries using additive noise assisted modeling and correlative information criterion. J Adv Res 25:49–56
57. Zhang Y, Liu Y, Sun P, Yan H, Zhao X, Zhang L (2020) IFCNN: a general image fusion framework based on convolutional neural network. Inf Fusion 54:99–118
58. Zhou DX (2020) Theory of deep convolutional neural networks: downsampling. Neural Netw 124:319–327

59. Zhou L, Zhang S, Yu J, Chen X (2019) Spatial-temporal deep tensor neural networks for large-scale urban network speed prediction. IEEE Trans Intell Transp Syst 21(9):3718–3729
60. Zhou M, Liu Y, Long Z, Chen L, Zhu C (2019) Tensor rank learning in CP decomposition via convolutional neural network. Signal Process Image Commun 73:12–21
61. Zolfaghari B, Srivastava G, Roy S, Nemati HR, Afghah F, Koshiba T, Razi A, Bibak K, Mitra P, Rai BK (2020) Content delivery networks: state of the art, trends, and future roadmap. CSUR 53(2):1–34