9

Self-Organization and Aggregation of Knowledge

Koichiro Ishikawa[1,2], Yoshihisa Shinozawa[2] and Akito Sakurai[2]
[1]*Aoyama Gakuin University*
[2]*Keio University*
[1,2]*Japan*

1. Introduction

Self-organization refers to formation processes of complex structures at the global level, which emerges solely from interactions among lower-level components without top-down controls or orders (Camazine et al. (2003)). Such processes are observed in many research areas (Johnson (2002); Buchanan (2007)), such as chemistry (e.g., Zhabotinsky (1991)), physics (e.g., Bak et al. (1988)), biology (e.g., Kauffman (1995)), brain and neuroscience (e.g., Haken (1996)), and economics (e.g., Krugman (1996)), etc.

On the other hand, phenomena, so-called "wisdom of crowds" (Surowiecki (2004)) or "collective intelligence" (Page (2007)) are also seen on the Internet, especially within the Web 2.0 (O'Reilly (2005)) environment (Tapscott & Williams (2006)). The mechanism for producing these phenomena has not yet been fully elucidated. But some of these phenomena have the same characteristics as self-organization, because individual user's diverse and spontaneous behavior or opinion aggregates to meaningful knowledge from the bottom up (Bonabeau et al. (1991)).

In the research area of computer science, a method called self-organizing map (SOM) has been developed and utilized (Kohonen (2001)). SOM is one of the neural network systems, which models the column structure of visual cortex (V1) in animal brains (Van Hulle (2000)) and learns based on an unsupervised machine learning algorithm. SOMs have been applied to problems in many fields such as data visualization, dimensional reduction, clustering, etc.

In this chapter, we focus on the multi-media content broadcasted on the Internet with the streaming technique. To serve most users by utilizing the bandwidth of the Internet, it is ideal to broadcast only important portions of the content that viewers really want to watch. If this is realized, viewers can avoid wasting their time by watching the portions that they do not desire to watch. Moreover, by not broadcasting less important portions unnecessary network traffic is decreased.

With the spread of broadband networks, it has become very popular to broadcast so-called, "rich" multi-media content including movies (e.g., *YouTube* (n.d.)). Therefore it is very urgent and meaningful to develop technology for sorting out important portions of the content.

In view of the current technological status, though, automatic recognition of importance is difficult to realize, since it is closely connected with meaning of the content. The difficulty is also obvious from the fact that almost all the search engines of Web pages (e.g., *Google* (n.d.)) provide the results based mainly on strings of characters and not their meaning. The reason why this method works is because we use words that reflect our intentions and the words are easy to recognize automatically. However, such methods cannot be applied to multi-media

content because in their expression, the meaning is implicit to the objects in them which are difficult to recognize.
The technique we propose in this chapter is to provide summary of viewing history of users who watched a particular multimedia content to users who are going to watch the content (Ishikawa et al. (2007)). To provide the summary of viewing history, the proposed technique aggregates users' behavior of content viewing. Our claim is that the resulted aggregation is in fact "wisdom of crowds" or "collective intelligence" is proven by experiments where new users have shortened their time to determine the important portions of the content. For the aggregation, based on similarity of learning mechanism of SOM and the aggregation process, we adopted two kinds of neural networks that are derived from SOM. We also reported on the experiments that were successful and that the implemented SOM worked efficiently in time and space.
We propose in this chapter two SOM-like algorithms that accept online, as input, the start and end of viewing of a multimedia content by many users; a neural network is finally self-organized, providing an approximation of the density distribution showing how many users see a part of a multimedia content. In this way "viewing behavior of crowds" information is accumulated as experience accumulates, summarized into one SOM-like network as useful knowledge for viewing is extracted, and is presented to new users as the knowledge is transmitted (See section 3.1.1 more detail).
Accumulation of multimedia content on the Internet increases the need for time-efficient viewing of the content and the possibility of compiling information on many users' viewing experiences. Under these circumstances, some methods have been proposed (e.g., Yu et al. (2003); Ishikawa et al. (2007)) that presents, in the Internet environment, a kind of summary of viewing records of many viewers of a certain multimedia content. The summary is expected to show that some part is seen by many users, but some other parts are rarely seen. The function is similar to websites utilizing "wisdom of crowds" and is facilitated by our proposed algorithm.
Our proposed methods are automatically and adaptively detect relatively important parts in multimedia content. The important parts of content should characterize the entire content and also be regarded as its summary. By applying our method, users can capture important screenshots of the content and also extract a set of important parts that contain characteristic scenes.
This chapter is organized as follows; section 2 introduces SOM. In section 3, each of the two methods proposed in this chapter are explained, the experiments performed for evaluating the effectiveness of the proposed methods, and the results are reported. Finally section 4 concludes this paper.

2. Self-organizing map

This section explains a self-organizing map (SOM) which is regarded as the most common process and concept which is referred to as self-organization. SOM is a method of unsupervised machine learning proposed by Kohonen, which is one of the neural networks that carries out competitive learning. SOM has been utilized for various purposes, such as function approximation, data clustering, and dimensional reduction of high-dimensional data.
The process of one-dimensional SOM learning is as follows. First, n neurons are allocated on a one-dimensional line. Without loss of generality, a line segment $[0, M]$ is considered, which is refered to as a line hereafter for simplicity. The position of each neuron, $y_i(t), 1 \leq i \leq n$ is

referred to as a feature vector. At an initial time $t = 0$, there is basically no difference in locating the neurons randomly or regularly (however it has been reported that, if many neurons are at the same initial positions, delay of convergence and deterioration in quality may occur). It has also been reported that the approximation accuracy obtained depends on the number of neurons.
Next, the network input $x(t)$ is presented to the SOM network. The neuron $y_{\hat{i}}(t)$ with the shortest distance to $x(t)$ becomes the winner. Euclidian distance is commonly used for measuring the distance. The winner is then output.
After outputting, the network learning is done based on the input $x(t)$. First, neighborhood neurons with distance to the winning neuron shorter than the predefined threshold are selected as target neurons for learning where the distance is usually the same as used for selecting the winner. The distance may be determined differently from the winning neuron. Furthermore, the threshold used to judge the neighborhood neuron has often been decreased gradually according to the number of inputs presented to the network (i.e., the range of neighborhood neurons is narrowed down gradually over time). The feature vectors $y_i(t)$ for the winning and neighborhood neurons are updated by the following formula based on the network-input $x(t)$;

$$y_i(t+1) = y_i(t) + \alpha\big(x(t) - y_i(t)\big). \tag{1}$$

Here, α is the learning coefficient. It is also common to change this value in accordance with the presented number of network-inputs.
As is seen in Equation 1, as the result of all the above mentioned processes, neurons are re-located on the line according to the observation frequency of network input; namely, there is a dense re-location of neurons for parts of the content where a high frequency of input was observed, while there was a sparse re-location of neurons for parts of the content where a low frequency of input was observed. Therefore, the network approximates the probability distribution of inputs with the granularity that corresponds to the number of the network neurons (section 3.3.1 gives a detail analysis on the functions of the SOM algorithm).

3. Methods and results

3.1 Problem settings

3.1.1 Overview

Users usually view multimedia content such as video sequentially. Let us call the content viewed sequentially as time-series digital content hereafter. Sequential content viewing requires exactly the same time as the recording time. Since users need to spend long periods of time to view time-series digital content, they would be pleased if a method would make it possible for them to view only the important parts in a short amount of time. Here, we presume that there are parts that the users truly desire to view coexisting with parts that are insignificant, especially when the digital content has a long recording time. Therefore, the purpose mentioned above can be achieved if the importance of each part of the content could be estimated in some way.
A possible solution would be to use still images (called snapshots or thumbnails) to represent parts of time-series digital content. In fact many programs do this. One of the drawbacks of this method is that the intervals of taking snapshots are constant. In this case, the intervals between the shots are relatively spaced out in important parts whereas they are relatively dense in the insignificant parts.
Another possible solution, that in fact we adopted, is to turn to "wisdom of crowds" (Surowiecki (2004)) or "collective intelligence" which has been attracting much attention

(Ohmukai (2006)). Although "wisdom of crowds" has not yet been clearly defined, common features would be to record users' input, to show a kind of summary of the users' input, and to implicitly assume that the more users carry out a particular behavior, the more the behavior becomes valuable.
A "wisdom of crowds" website is considered to having the function of promoting knowledge sharing and developing by hopefully realizing phases of socialization and externalization of the SECI model proposed by Nonaka & Takeuchi (1995). It accepts many users' input, accumulates them, and presents their summaries. By looking at the summaries, the users might add new input. If the system works as desired, it helps knowledge to accumulate, to be shared, and to develop.
To realize the idea of "wisdom of crowds" in effective content viewing, we need a way to accumulate and summarize time-series content. Since the semantics of time-series content is hard to analyze, some new idea is necessary. We focus on the viewing frequency of each part of content. A system that records other users' viewing history and provides frequency about what other users view enables a new user to see how majority behaves, as with the above mentioned "wisdom of crowds" websites.
An actual system requires the function to reflect viewing operations of online users as well as to receive and deliver content on the Internet. The function should reflect promptly the input and operation carried out by each user, then such reflected information is used to facilitate user operation.
To handle vast quantities of data online, Ishikawa et al. (2007) proposed to record data on a relational database (RDB). If viewing behavior data of all users is recorded completely in a database, an accurate histogram can be obtained. However, considering the total quantity of multimedia content on the Internet, it is more costly than necessity to record all the viewing behavior data. If only smaller amount of data is recorded, the accuracy of the histogram degrades.
To solve the conflicts, we propose to use two SOM-like neural networks that successively approximate input data, and memorize the up-to-date results. Required data storage capacity is on the order of a fixed length of a real vector (array) of the neurons of the SOM in both of the two networks. Conventional SOM cannot be applied to the subject of this study. This is because the viewing history data does not represent viewing parts directly, but rather only the start and end points of the viewing. If representation of viewing parts could be obtained directly it would be possible to learn approximate viewing frequency with SOM, however, in fact only the starting and ending points of viewing is available. Consequently, we have developed two new algorithms and adopted them.

3.1.2 Sharing viewing history

This section explains summarization of multimedia digital contents through sharing viewing history. Ishikawa et al. (2007) proposed a system where online viewers of multimedia content transmit descriptions of where they viewed and skipped, as viewing history. The viewing history is recorded and processed on a server on the network (refer to Fig. 1). And then the server provides the processed summary to new viewers of the content (refer to Fig. 2).
It is likely that the parts that many viewers actually viewed indicate the important parts in the content. Viewers are able to reflect the summary of viewing history provided by the system on their viewing behavior, namely, they can utilize this information in order to view only the important parts of the content skipping other parts they deem unnecessary. In Ishikawa et al.

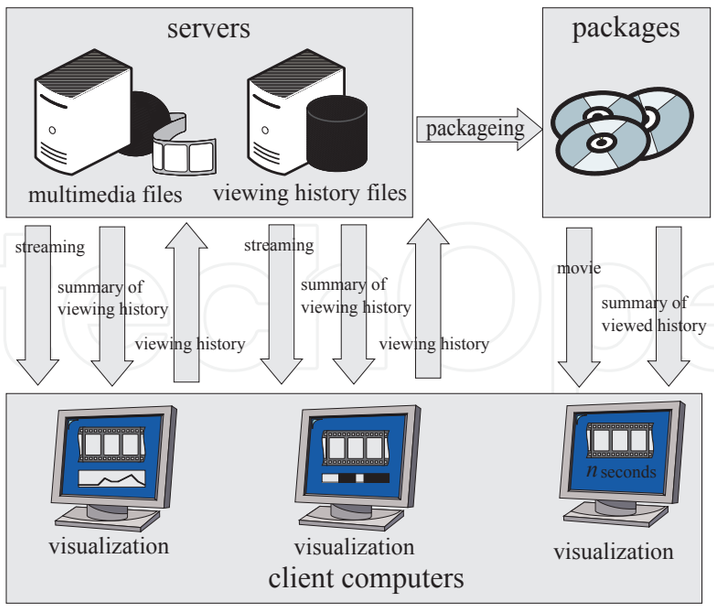


Fig. 1. Environment proposed in Ishikawa et al. (2007).

(2007), the data actually transmitted is a pair

$$R(t) = \langle position(t), operation(t) \rangle, \tag{2}$$

or, more precisely, either of the following:

$$R_{start}(t) = \langle position(t), 1 \rangle, \tag{3}$$

$$R_{stop}(t) = \langle position(t), -1 \rangle, \tag{4}$$

where t is the time when an event $R(t)$ occurs supposing that the time is so discretized that the two events do not occur at the same time, $position(t)$ is the position in a multimedia content where the viewing event starts or ends, operation is 1 (or -1) when viewing starts (or ends, respectively).
A position in time-series digital content is specified by the relative distance from its start in terms of replay time length, e.g., if the content is 10 minute length and the position is 2 minutes from its start, the position is said to be 0.2. $R_{start}(t)$ is sent when a play button is pressed, fast forward button is released, or slide bar is released after sliding; $R_{stop}(t)$ is sent when a stop button is pressed, fast forward button is pressed, or slide bar is moved first. The server stores $R_{start}(t)$ and $R_{stop}(t)$ in a RDB , according to the method proposed in Ishikawa et al. (2007). Furthermore, the server provides the following information to new viewers of the content.

$$C(p) = \sum_{t \in \{t | position(t) \leq p\}} operation(t). \tag{5}$$

This process is realized by executing the following SQL commands on the RDB of the content distribution server.

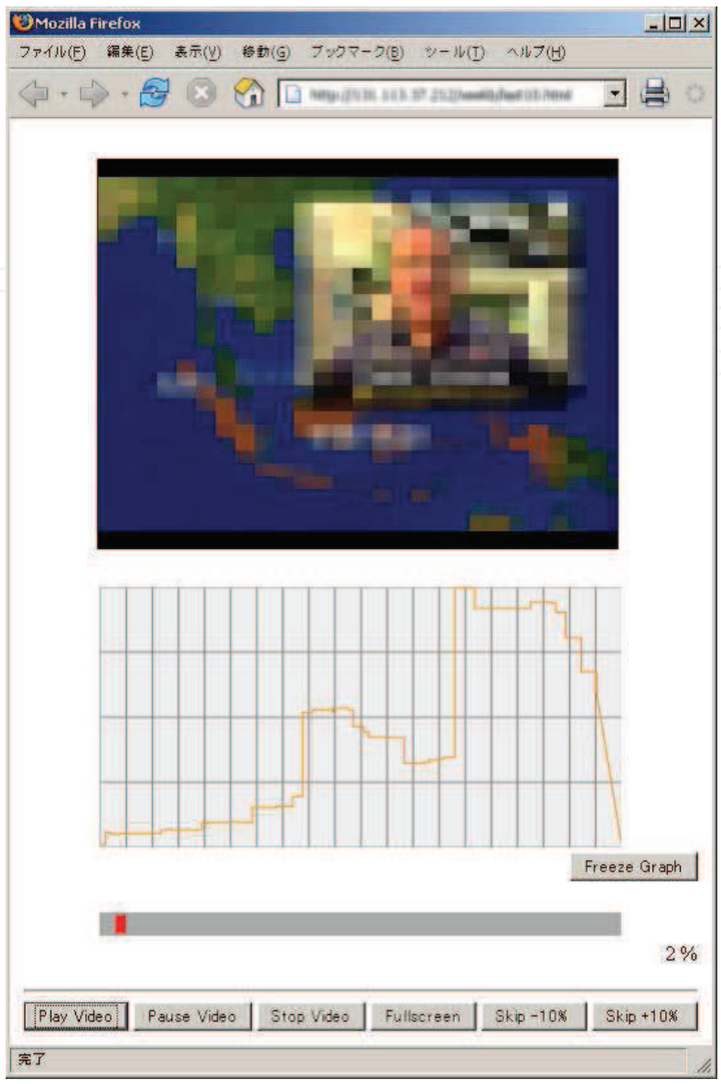


Fig. 2. A scrrenshot on a client computer proposed in Ishikawa et al. (2007).

```
CREATE VIEW tmpview AS
 SELECT position,
        sum(operation) AS count
   FROM history GROUP BY ROUND(position);
SET @i=0;
SELECT position, @i:=@i + count AS result
   FROM tmpview ORDER BY position;
```

The client computer that receives $C(p)$ draws a graph of $C(p)$ on a 2D plane (Fig. 2). From the graph, users are able to learn how frequently each part of the video content they are viewing was viewed, and to reflect it in their own viewing behavior, by skipping some parts of the content.

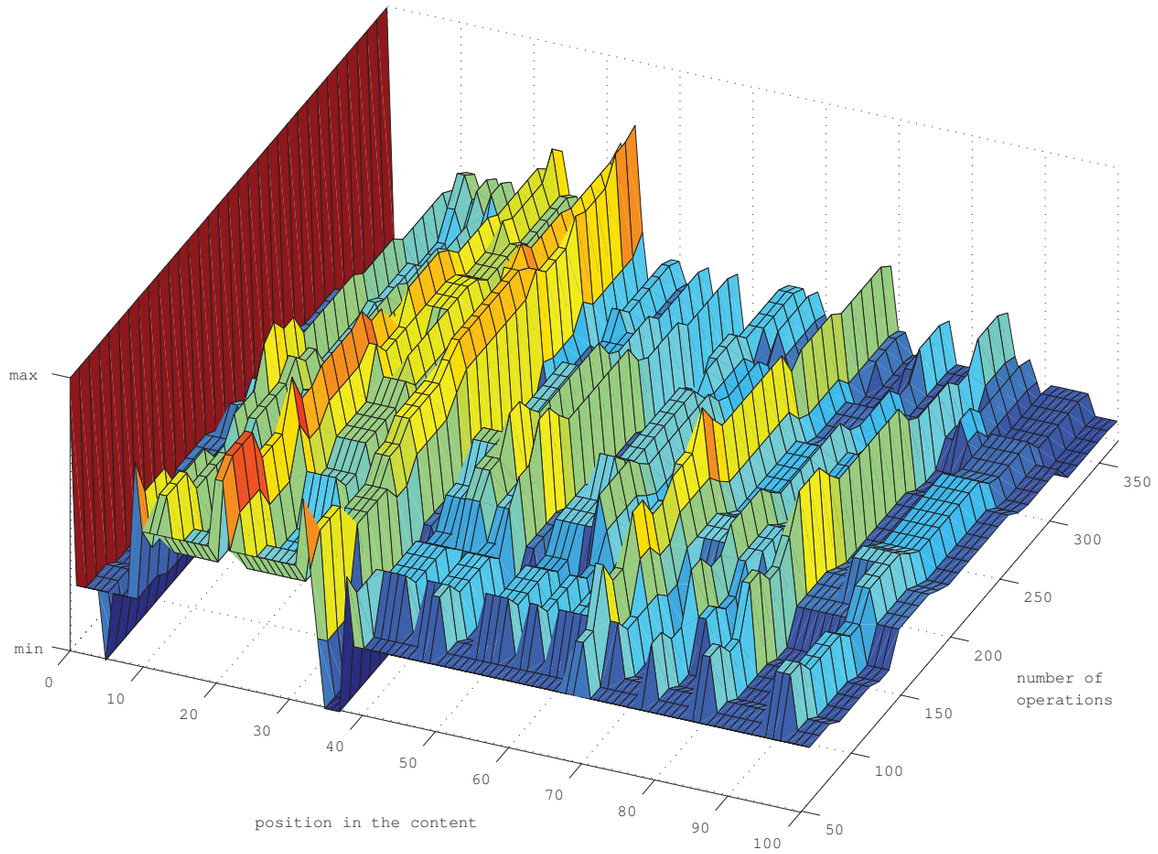


Fig. 3. Experimental result in Ishikawa et al. (2007). X axis shows position in a content. Z axis shows the viewed frequency at each X, i.e., X-Z graph is histogram of viewed frequency of the content at a moment. Y axis shows the number of operations. As the number of operations increases, the peak of histogram (X value of about 34) become distinctive (section 3.3.2.3 gives experimental details).

3.1.3 Online identification of summarizing points

This section proposes a method to identify content parts that characterize the entire content. Note that time-series digital content is in general considered to be composed of multiple parts. A part in this case, may be a still image of one scene, or time-series movie.

The method we considered is (1) to utilize viewing history described in section 3.1.2 and (2) to adopt SOM to summarize the viewing history. The reason why viewing history should be utilized is that the parts only a few user view will not characterize the whole content whereas the parts that many users view may characterize the whole. The reason why SOM should be adopted is that SOM learns frequency distribution of inputs as feature vectors of neurons gather to high frequency area of inputs and disperse from low frequency area of inputs. The reason why we combine them is that we could know important parts from where neurons gather and less important parts from where neurons dispers if SOM learns viewing history.

One additional merit of using SOM is that its space complexity is far less than that of RDB, since SOM is represented by feature vectors, i.e., an array of real numbers or integers. There are two disadvantages of using conventional SOM. One is that the conventional SOM accepts points as input, i.e., a scalar data, whereas since the viewing history is a list of range data, that

is, pairs of staring and ending of viewing, SOM must accepts range data as inputs. The other is that the starting and ending points of viewing do not necessarily come in a pair to a server computer as pointed out in Ishikawa et al. (2007).
To make matters worse, starting data and ending data from the same client computer do not necessarily have the same identification code, so that it is difficult to match the starting data and ending data from the same client computer. If we could assign identification code to the client computer, we could solve the problem. One possible ID is the IP address of the client computer, but IP addresses of many computers are assigned dynamically so that they may change between the starting of viewing and the end of viewing. The other possible ID is a cookie that would be set in a multimedia player and referred to by a server computer, but cookies by multimedia players are not popular and not standardized yet.
Since the advantages of SOM are indispensable, we devised two new methods, both of them consist of networks and their learning algorithm based on the conventional SOM. The proposed method described in section 3.2 has an SOM-like network that accepts starting points and ending points independently, that is, in any order without identifying its counterpart and learns viewing frequency distribution. The one in section 3.3 has two SOM-like networks, each of which accept one of starting and ending points and learn independently and one more SOM-like network that learns viewing frequency distribution from the former networks.

3.2 Method 1: with a single neural network

3.2.1 Overview

Our purpose is to recover frequency distribution of viewing events from their start and end events. In this section, we focus on equal density partition $x_0 < x_1 < \cdots < x_n$ of frequency distribution $p(x)$ such that $\int_{x_i}^{x_{i+1}} p(x)\,dx$ is a constant independent of i.
The proposed algorithm is shown in Fig. 4. Corresponding to the type of $position(t)$ and $operation(t)$ of network input (see Equation 2), the values of neurons, i.e., positions are updated (lines 10–31, 40). Since an update step α is a constant, a neuron might move past a neuron next to it. To prevent this, a neuron should maintain a certain distance (ε) from the neuron next to it (lines 32–39). Derivation of update formulae is as follows: Consider one-dimensional SOM-like network $\mathcal{X}$

$$\mathcal{X} = \langle x_1, \ldots, x_n \rangle,\ x_1 < x_2 < \cdots < x_n \in \Re^1.$$

If $\mathcal{X}$ is arranged such that for some c

$$\int_{x_i}^{x_{i+1}} p(x)\,dx = c,$$

then clearly $\mathcal{X}$ reflects the density function $p(x)$ in such a way that

$$\frac{c}{x_{i+1} - x_i} \approx p(x),\ \text{for } x \in [x_i,\, x_{i+1}].$$

Suppose that $p(x)$ is sufficiently smooth and for simplicity the sign of $\partial p\,/\,\partial x$ does not change in (x_i, x_{i+1}). Then, by putting $y = p(x)$,

initialize network $\mathcal{Y} \leftarrow \mathcal{Y}^0 = \langle y_1^0, \ldots, y_{|\mathcal{Y}|}^0 \rangle, y_1^0 < y_2^0 < \cdots < y_{|\mathcal{Y}|}^0$
$t \leftarrow 0$
repeat forever
  $t \leftarrow t + 1$
  receive operation information $R(t) = \langle x(t), op(t) \rangle$
  $B = \langle b_0 \leftarrow 0, b_1 \leftarrow y_1, \ldots, b_{|\mathcal{Y}|} \leftarrow y_{|\mathcal{Y}|}, b_{|\mathcal{Y}|+1} \leftarrow \sup(\mathcal{Y}) \rangle$
  for $i \in \{1, 2, \ldots, |\mathcal{Y}|\}$
    $\Delta_i \leftarrow 0$
  end for
  if $op(t) = 1$ **then**
    for $i \in \{2, 3, \ldots, |\mathcal{Y}| - 1\}$
      if $x(t) < b_i$ **then**
        $\Delta_i \leftarrow 2y_i - b_{i+2} - b_i$
      elseif $x(t) < y_i$ **then**
        $\Delta_i \leftarrow 2y_i - b_{i+2} - x(t)$
      elseif $x(t) < y_i$ **then**
        $\Delta_i \leftarrow b_{i+2} - x(t)$
      end if
    end for
  else
    for $i \in \{2, 3, \ldots, |\mathcal{Y}| - 1\}$
      if $x(t) < b_i$ **then**
        $\Delta_i \leftarrow -(2y_i - b_{i+2} - b_i)$
      elseif $x(t) < y_i$ **then**
        $\Delta_i \leftarrow -\big(2y_i - b_{i+2} - x(t)\big)$
      elseif $x(t) < y_i$ **then**
        $\Delta_i \leftarrow -\big(b_{i+2} - x(t)\big)$
      end if
    end for
  end if
  $\mathcal{Z} = \langle z_1, \ldots, z_{|\mathcal{Y}|} \rangle \leftarrow \mathcal{Y} + \alpha\Delta$
  for $i \in \{2, 3, \ldots, |\mathcal{Y}| - 1\}$
    if $z_i \leq y_{i-1}$ **then**
      $z_i \leftarrow y_{i-1} + \varepsilon$
    end if
    if $z_i \geq y_{i+1}$ **then**
      $z_i \leftarrow y_{i+1} - \varepsilon$
    end if
  end for
  $\mathcal{Y} \leftarrow \mathcal{Z}$
end repeat

Fig. 4. Procedures of the proposed in section 3.2

$$
\begin{aligned}
& \int_{x_0}^{x_n} p(x)\,dx \\
= & \int_{p(x_0)}^{p(x_n)} y\,\frac{\partial p^{-1}}{\partial y}\,dy \\
= & \; p^{-1}(y)\,y\Big|_{p(x_0)}^{p(x_n)} - \int_{p(x_0)}^{p(x_n)} p^{-1}(y)\,dy \\
= & \; \big[x_n\,p(x_n) - x_0\,p(x_0)\big] \; - \sum_{i=0}^{n-1}\int_{p(x_i)}^{p(x_{i+1})} p^{-1}(y)\,dy \\
= & \; \big[x_n\,p(x_n) - x_0\,p(x_0)\big] \; - \sum_{i=0,\,x_i\le\exists x\le x_{i+1}}^{n-1} \big[p(x_{i+1}) - p(x_i)\big]\,x \\
= & \; (x_n - x_0)\,p(x_0) \; + \sum_{i=0,\,x_i\le\exists x\le x_{i+1}}^{n-1} \big[p(x_{i+1}) - p(x_i)\big]\,(x_n - x).
\end{aligned}
$$

Hereafter $p(x_0) = 0$ is assumed. If increment or decrement events, as Equation 3 or 4, $\Delta p(x)$ occur such that

$$\frac{\partial p}{\partial x}\,\Delta x \approx E[\Delta p(x)], \text{ for } x \in [x_0, x_n],$$

then

$$\int_{x_0}^{x_n} p(x)\,dx \approx E[\Delta p(x)\,(x_n - x)].$$

Therefore if we could arrange $\mathcal{X}$ such that for $x \in [x_i, x_{i+1})$

$$E[\Delta p(x)\,(x_n - x)]$$

is a constant independent of i, $\mathcal{X}$ is the one we want. From this we get the following update formulae for x_i

$$x_i \leftarrow x_i + \alpha\,\Delta x_i,$$

where

$$\Delta x_i = \begin{cases} \Delta p\,\big[(x_{i+1} - x_i) - (x_i - x_{i-1})\big], & (x < x_{i-1}) \\ \Delta p\,\big[(x_{i+1} - x_i) - (x_i - x)\big], & \big(x \in [x_{i-1}, x_i)\big) \\ \Delta p\,(x_{i+1} - x), & \big(x \in [x_i, x_{i+1})\big) \\ 0, & (x \ge x_{i+1}), \end{cases}$$

and $\Delta p(x) = \Delta p$ for any x.

3.2.2 Experiments

We describe the results of experiments conducted to verify the proposed algorithm. The parameters were set for the experiments as follows;

- The number of neurons in the network $\mathcal{Y}$ (See line 1 in Fig. 5) is 41, and the neuron are initially positioned equally spaced between 0 and 100.
- The learning parameter α is fixed at 0.1.
- The parameter ε , the minimum separation between the neurons, is fixed at 0.01.

3.2.2.1 Single peak with straight slopes

We experimented on a single-peaked frequency distribution, which is a relatively simple example, as a viewing frequency distribution. The result is shown in Fig. 5.
To simulate such viewing history, the network input was given to the network with the following conditions:

– Viewing starts at the position p selected randomly from the range of positions 40 through 50 of content with 50% probability.
– Viewing ends at the position p selected randomly from the range of positions 75 through 85 of content with 50% probability.

The frequency of viewing operations is indicated by the solid line on the upper pane of Fig. 5. The horizontal axis is the relative position from the start of the content. The vertical axis indicates the sum of viewing operations, where the starting operation is 1 and the ending operation is -1 up to the position, thus $C(p)$ in Equation 5. The lower pane of Fig. 5 shows how the neuron positions in the network $\mathcal{Y}$ change as inputs are presented to the network. The change up to 10,000-th input is shown in the figure.
It shows that neurons gathered to the frequently-viewed part before 1,000 network-inputs. After that the neurons on x such that $p(x) = 0$ continued to be absorbed into the area $p(x) > 0$. The position of each neuron at 10,000-th inputs is plotted with circles overlapping on the upper pane of Fig. 5 where the relative vertical positions are not relevant. The plateau in this figure corresponds to high frequency of viewing, and neurons are located on these parts with gradual condensation and dilution on each side.

3.2.2.2 Double peaks with jagged slopes

Fig. 6 shows the result for a frequency distribution with double peaks with jagged slopes, which is more similar to practical cases. The axes in the figure are the same as Fig.5. In this experiment, neurons gathered at around two peaks, not around valleies after about 4,000-th inputs.

3.2.3 Discussion

In section 3.2 we focused on the utilization of a kind of "wisdom of crowds" based on observed frequency of viewing operations. "kizasi.jp" is an example of a site which utilizes "wisdom of crowds" based on word occurrence or co-occurrence frequencies which are observed in blog postings. Here words play the role of knowledge elements that construct knowledge.
Multimedia content has different characteristics than blogs, which causes difficulties. It is not constructed from meaningful elements. Even a state of the art technique would not recognize the meaningful elements in multimedia content.
A simple way to circumvent the difficulty is to utilize occurrences or frequency of viewing events for the content (Ishikawa et al. (2007)). But, since multimedia content is continuous, direct collection and transmission of viewing events are very costly. Since a viewing event consists of a start and an end point, we can instead use these and recover the viewing event.
In this section, we considered a new SOM-like algorithm which directly approximates the density distribution of viewing events based on their start and end points. We have developed a method based on SOM because SOM has an online algorithm, and the distribution of obtained neurons reflects the distribution of occurrence density of given data.
A clustering algorithm can also serve as a base algorithm for the problem. However, the problem that we want to solve is not to get clusters in viewing frequency but to present the overall tendency of viewing frequency to users.

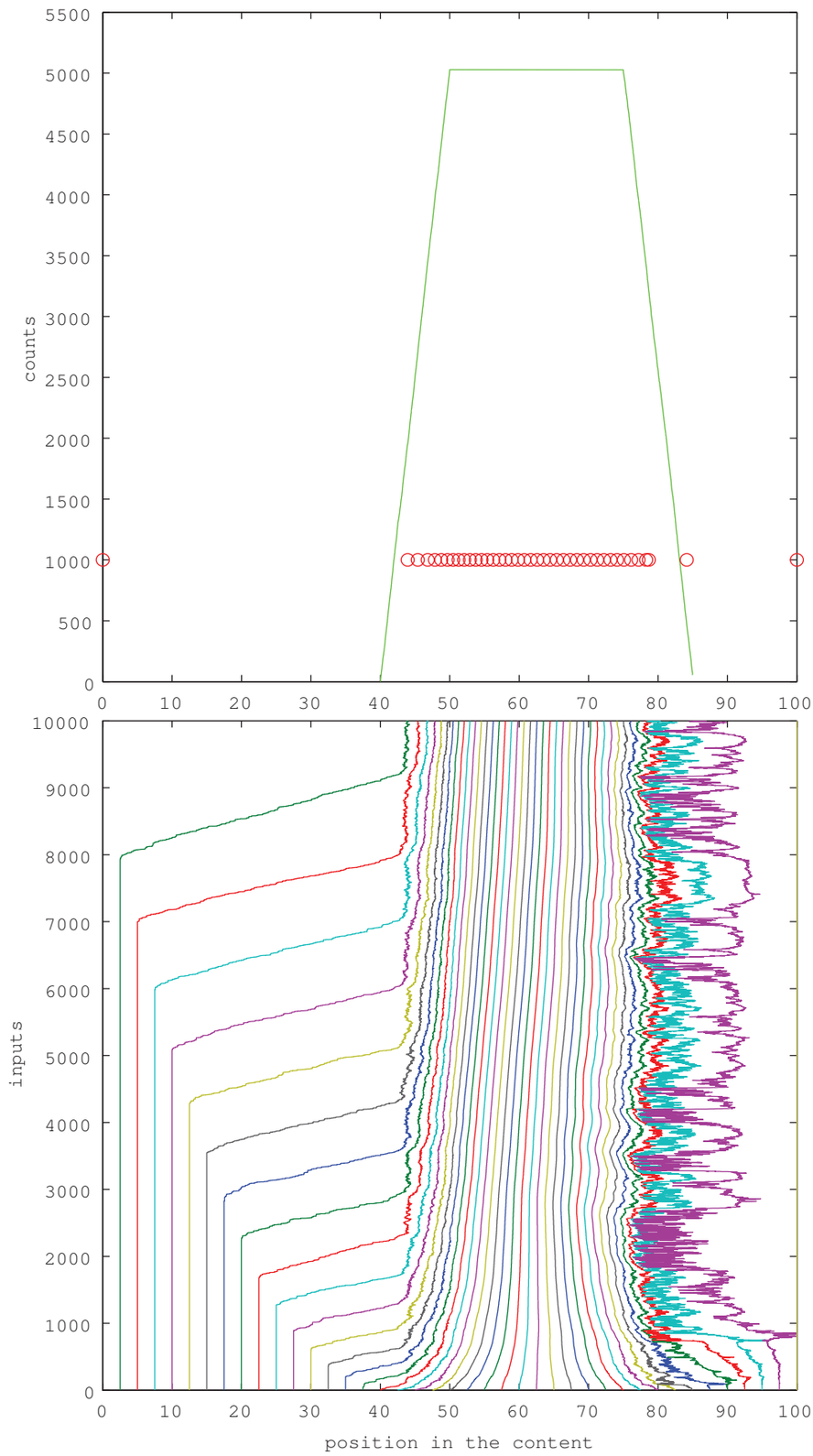


Fig. 5. Result of an experiment in section 3.2.2.1.

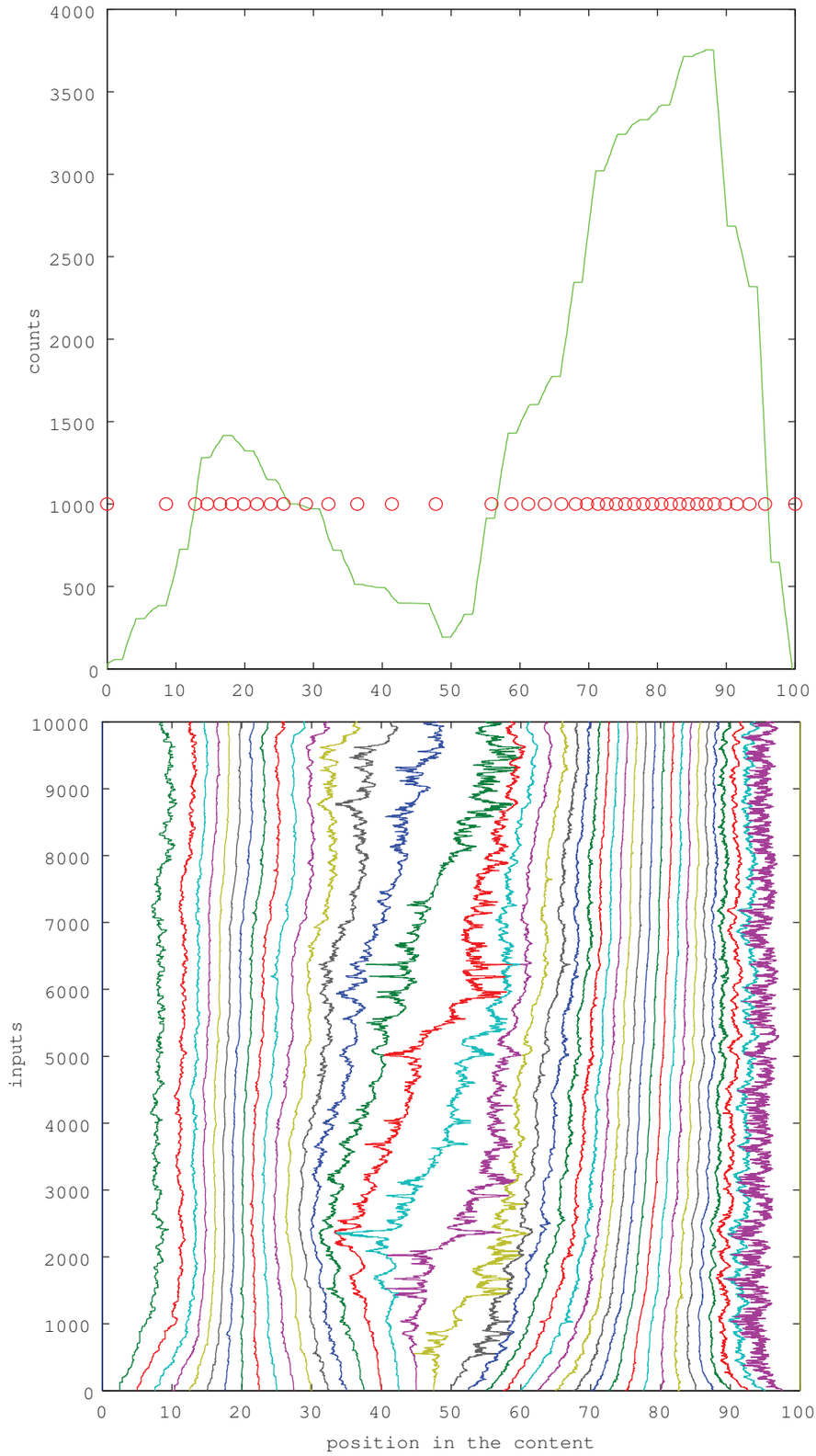


Fig. 6. Result of an experiment in section 3.2.2.2.

By applying the proposed algorithm, the computational complexity of time and space can be reduced substantially, compared with, for example, a method of recording all the viewing history data using RDB. Time complexity of an algorithm using RDB is as follows, where n is the number of histogram bins and corresponds to the number of neurons in our proposed algorithm.

1. $R(t) = \langle p, m \rangle$ is inserted into sorted array A which stores all the information (start and end points) received from users (see Ishikawa et al. (2007)):

   $\mathbf{\Omega}(\log t) + \mathbf{\Omega}(\log t)$

2. B, an array obtained from A by quantizing the time scale is calculated:

   $\mathbf{\Omega}(\log t)$

3. From $b_i = i \times \frac{\left(\max(B) - \min(B)\right)}{n} \in B$, $1 \leq i \leq n$, b_i is calculated:

   $\mathbf{\Omega}(\log t)$

On the other hand, the process of the algorithm proposed in this section does not require sorting (above in 1) and deciding the insertion location of data (above in 2), but requires the network learning process for each input observed data. Time complexity is calculated as follows;

4. $\mathrm{argmin}_i \left(\| p - y_i \| \right)$, in the network are calculated.

   $\mathbf{O}(n)$

5. The feature vectors of the winning neuron and the neurons to be updated are updated.

   $\mathbf{O}(n)$

Hence, time complexity does not depend on t. Space complexity is also only $\mathbf{O}(n)$.
To see how the algorithm converges, we kept on learning up to 50,000 inputs from the same distribution as in Fig. 5, decreasing the learning parameter (α) linearly from 0.1 to 0.001 after the 10,000-th network-input. Fig. 7 shows the frequency distribution $\left(1 \,/\, (y_{i+1} - y_i)\right)$ calculated using the neurons' final position y_i, plotted as "+", compared to that obtained directly from the input data by Equation 5, plotted as a solid line. The horizontal axis of Fig. 5 indicates the relative position in the content and the vertical axis indicates observation frequency normalized as divided by their maximal values. The result shows that the neurons converged well to approximate the true frequency.

3.3 Method 2: with multiple neural networks

3.3.1 Overview

3.3.1.1 Estimation of viewing section

For the reasons described at the end of section 3.1.3, in this section, we divide the proposed method into two phases, namely, the phase to estimate a viewing part by obtaining start/end points, and the other phase to estimate the characteristic parts through the estimation of

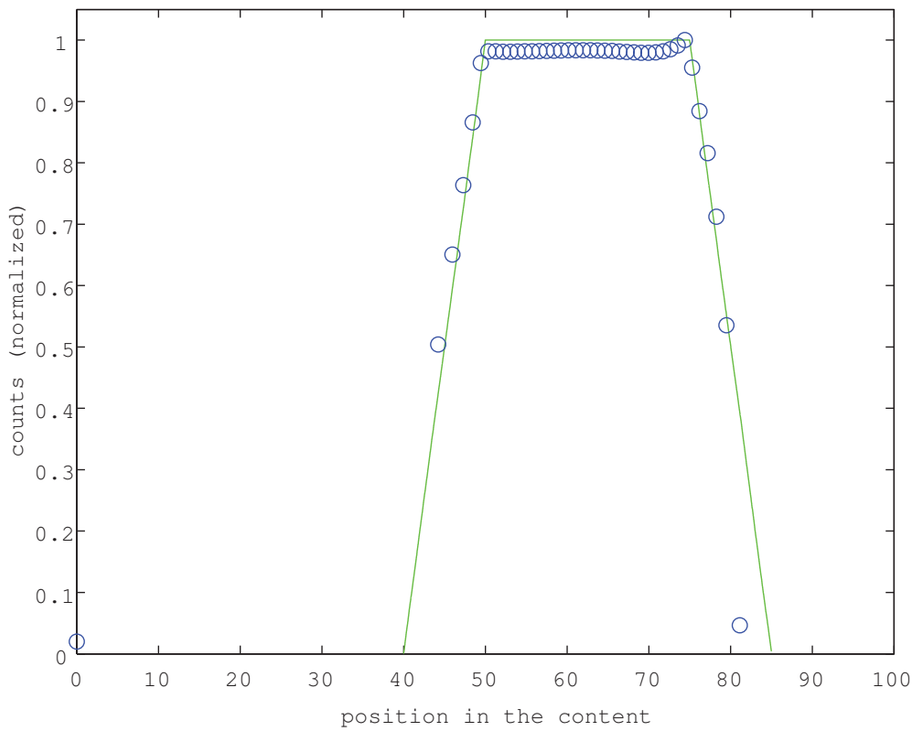


Fig. 7. True frequency distribution (solid line) versus approximated one (circle) for the same input distribution as in section 3.2.2.1.

viewing parts. SOM is utilized in each phase so that we are able to achieve the identification of characteristic parts in a content.

We want to clarify what the neurons will approximate with the SOM algorithm before describing the method proposed in this section. In the SOM or LVQ algorithm, when it has converged, neurons are re-located according to the frequency of network-inputs. Under the problem settings stated in section 3.1, the one-dimensional line is scalar quantitized by each neuron after learning (Van Hulle (2000)).

In other words, the intermediate point of two neurons corresponds to the boundary of the Voronoi cell (Yin & Allinson (1995)). The input frequency on a line segment, L_i, separated by this intermediate point is expected to be $1 \,/\, \|L_i\|^2$. This is because, when LVQ or SOM has converged, the expected value of squared quantitization error E is as follows;

$$E = \int \|x - y_i\|^2 \, p(x) \, dx$$

where $p(x)$ is the probability density function of x and i is the function of x and all $y_\bullet$ (See section 2 for x_i and y_i). This density can be derived as follows (Kohonen (2001));

$$\nabla_{y_i} E \quad = \quad -2 \int (x - y_i) \, p(x) \, dx.$$

It is shown, intuitively, by

$$(y^2 - 2xy)\Big|_{y_i}^{\frac{y_{i+1}+y_i}{2}} \approx (y^2 - 2xy)\Big|_{\frac{y_i+y_{i-1}}{2}}^{y_i}$$
$$y^2\Big|_{y_i}^{\frac{y_{i+1}+y_i}{2}} \approx y^2\Big|_{\frac{y_i+y_{i-1}}{2}}^{y_i}.$$

Refer to section 2 about x and y_i. Therefore, the input frequency z_i on the line segment L_i is expressed as follows;

$$\begin{aligned} z_i &\approx \|L_i\| \,/\, \|L_i\|^2 \\ &= \left(\frac{y_{i+1} - y_i}{2} + \frac{y_i - y_{i-1}}{2}\right)^{-1} \\ &= \frac{2}{y_{i+1} - y_{i-1}}, \end{aligned} \tag{6}$$

where $y_0 = 0$ and $y_{n+1} = M$. Moreover, the piecewise linear function in the two-dimensional plane that connects

$$\left(y_i, \textstyle\sum_{j=1}^{i} z_j \Big/ \sum_j z_j\right) \tag{7}$$

approximates the cumulative distribution function of $x(t)$. Section 3.3.2.1 experimentally confirms this point (also refer to Fig. 7 in section 3.2.3).

Fig. 8 shows the procedure of the proposed method overall. In this section, we explain from line 1 through 32 only that shows how to estimate frequency of viewing parts by using the above mentioned function of SOM (From line 33 through 43 will be explained in the next section).

We prepared the network (a set of neurons) $\mathcal{S}$ that learns the start point of viewing and the network $\mathcal{E}$ that learns the end point of viewing. Here, the number of neurons of each SOM is set at the same number for simplicity. The learned result of the networks $\mathcal{S}$ and $\mathcal{E}$ will be used for approximating the viewing frequency by network $\mathcal{F}$ (the next section goes into detail) .

According to the network input type, R_{start} or R_{stop} in Equation 2 (lines 5–10), either network $\mathcal{S}$ or $\mathcal{E}$ is selected in order to have it learn the winning as well as neighborhood neurons by applying the original SOM updating formula of Equation 1 (lines 11–18). As stated above, the frequency of inputs in each Voronoi cell, whose boundary is based on the position of each neuron after the learning, is obtained in both networks $\mathcal{S}$ and $\mathcal{E}$.

Based on the above considerations, we propose the following process. When the input $x(t)$ is the start point of viewing (line 19) [1] , the frequency z_i is calculated as below, based on Equation 6.

$$z_i = \begin{cases} \frac{2}{y_{i+1} - y_{i-1}}, & \text{if } y_i \in \mathcal{S} \\ \frac{-2}{y_{i+1} - y_{i-1}}, & \text{if } y_i \in \mathcal{E} \end{cases} \tag{8}$$

[1] When the input $x(t)$ is the end point of viewing, we could reverse the process of Fig. 9 to calculate the cumulative sum until the sum becomes 0 or negative, coming back, in the relative position in a content, from the end point of viewing. And it is expected that applying this process could double the learning speed. However, this paper focuses on only the start point of viewing for simplicity.

neuron $y \leftarrow$ initial value, $\forall y \in$ SOM $\mathcal{S}, \mathcal{E}, \mathcal{F}$
$t \leftarrow 0$
repeat forever
  $t \leftarrow t + 1$
  receive operation data $R(t) = \langle x(t), op(t) \rangle$
  if $op(t) = 1$ **then**
    let $\{y_i \mid 1 \leq i \leq |\mathcal{S}|, y_i \leq y_{i+1}\} \leftarrow \mathcal{S}$ and $i_{max} \leftarrow |\mathcal{S}|$
  else
    let $\{y_i \mid 1 \leq i \leq |\mathcal{E}|, y_i \leq y_{i+1}\} \leftarrow \mathcal{E}$ and $i_{max} \leftarrow |\mathcal{E}|$
  end if
  $\hat{\imath} \leftarrow \mathrm{argmin}_i \left(\|x(t) - y_i\| \right)$
  $y_{\hat{\imath}} \leftarrow y_{\hat{\imath}} + \alpha \left(x(t) - y_{\hat{\imath}} \right)$
  if $\hat{\imath} > 1$ **then**
    $y_{\hat{\imath}-1} \leftarrow y_{\hat{\imath}-1} + \alpha \left(x(t) - y_{\hat{\imath}-1} \right)$
  end if
  if $\hat{\imath} < i_{max}$ **then**
    $y_{\hat{\imath}+1} \leftarrow y_{\hat{\imath}+1} + \alpha \left(x(t) - y_{\hat{\imath}+1} \right)$
  end if
  if $op(t) = 1$ **then**
    ordered set $S' \leftarrow \{i \mid s_i \in \mathcal{S},\ x(t) < s_i\}$
    ordered set $E' \leftarrow \{j \mid e_j \in \mathcal{E},\ x(t) < e_j\}$
    $Z \leftarrow s_{\min S'}$
    remove $\min S'$ from S'
    while $Z > 0$ **do**
      $w \leftarrow \mathrm{argmin}(s_{\min S'}, e_{\min E'})$
      remove w from S' or E'
      if w comes from S' **then**
        $z \leftarrow \frac{2}{s_{w+1} - s_{w-1}}$
      else
        $z \leftarrow \frac{-2}{e_{w+1} - e_{w-1}}$
      end if
      $Z \leftarrow Z + z$
    end while
    if $w > 0$ **then**
      $v \leftarrow x(t) + rand \times (e_w - x(t))$
      $\hat{\imath} \leftarrow \mathrm{argmin}_i \left(\|v - f_i\| \right),\ f_i \in \mathcal{F},\ 1 \leq i \leq |\mathcal{F}|$
      $f_{\hat{\imath}} = f_{\hat{\imath}} + \alpha \left(v - f_{\hat{\imath}} \right)$
      if $\hat{\imath} > 1$ **then**
        $f_{\hat{\imath}-1} \leftarrow f_{\hat{\imath}-1} + \alpha \left(v - f_{\hat{\imath}-1} \right)$
      end if
      if $\hat{\imath} < |\mathcal{F}|$ **then**
        $f_{\hat{\imath}+1} \leftarrow f_{\hat{\imath}+1} + \alpha \left(v - f_{\hat{\imath}+1} \right)$
      end if
    end if
  end if
end repeat

Fig. 8. Procedure of the method proposed in section 3.3.

Here, the cumulative sum Z of frequency z_i is calculated for the neurons $y_i (y_i \geq y_{\hat{\imath}}, y_i \in \mathcal{S}, \mathcal{E})$ which are right to the winning neuron $y_{\hat{\imath}}$ (lines 24–33). The neuron $e_w(t)(\in \mathcal{E})$ is identified at which Z becomes 0 or negative first from the right of the winning neuron. Thus, the cumulative frequency of the beginning/ending operations of viewing is accumulated from the winner neuron, as starting point, through the point of $e_w(t)$ at which the cumulative frequency becomes 0 or negative value for the first time, after decreasing (or increasing temporally) from a positive value through 0, that means the cumulative sum of positive values is equivalent to, or become smaller than, the cumulative sum of the negative values. Namely, $e_w(t)$ corresponds to obtaining the opposite side of the peak of the probability density function (refer to Fig. 9).
Hence, the interval is obtained, as mentioned above, that is between observed $x(t)$, as the beginning point of viewing, and the point $e_w(t)$ on the opposite side of the peak on the smoothed line of histogram. We could fairly say that this interval $[x(t), e_w(t)]$ is the most probable viewed part when the viewing starts at $x(t)$.

3.3.1.2 Adaptive estimation of partial content position

Based on the estimated most probable viewing parts obtained as the result of the process described in the previous section, the following process concentrates neurons to a content part viewed frequently. The third SOM $\mathcal{F}$ is prepared to learn the frequency based on the estimation of the above mentioned most probable viewing parts, namely, the targeted content parts.
From the most probable viewing parts, $[x(t), e_w(t)]$, estimated by applying the process of Fig. 9, a point v is selected randomly with uniform probability and presented to network $\mathcal{F}$ as the network-input (lines 35–36). Then network $\mathcal{F}$ learns by applying the original SOM procedure (lines 37–43). In other words, the winning and neighborhood neurons are determined for point v that was selected above, and then the feature vectors are updated based on Equation 1.
As the result of this learning, the neurons of network $\mathcal{F}$ are densely located in parts viewed frequently. In short, in the process of the method proposed above, the estimated most probable viewing part ($[x(t), e_w(t)]$ above) is presented to SOM network $\mathcal{F}$ as network-input in order to learn the viewing frequency of the targeted part. The neurons (feature vectors) of the SOM network $\mathcal{F}$ is re-located as their density reflects the content viewing frequency. Based on this property, we can extract the predetermined number (the number of the network $\mathcal{F}$ neurons) of frequently viewed content parts by choosing the parts corresponding to the point the neurons are located. The above mentioned process is incremental; therefore, the processing speed will not be affected significantly by the amount of stored viewing history data.
The proposed method can serve, for example, to obtain locations in the content to extract still images as thumbnails. We can extract not only still images at the position of neurons, but also content parts having specified length of time, the center of which is the positions of neurons, if we specify the time length within the content.

3.3.2 Experiments

The following sections describe the results of the experiments performed in order to evaluate the algorithm proposed in the previous section. The following conditions were adopted in the experiments performed.

– The number of neurons in either of the network $\mathcal{S}$, $\mathcal{E}$, or $\mathcal{F}$ is 11; the neurons are initially positioned equally spaced between 0 and 100.

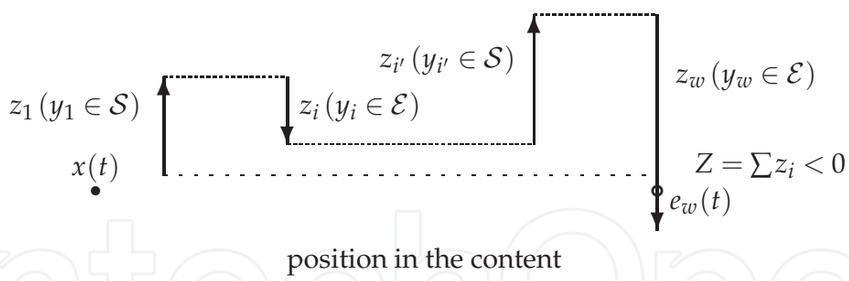


Fig. 9. Visualizing example of cumulative sum of z_i.

- The learning coefficient of each SOM is fixed at 0.01 from the beginning until the end of the experiment.
- Euclidian distance is adopted as the distance to determine the winning neuron.
- A neuron that locates adjacent to the winning neuron (topologically) is the neighborhood neuron (the Euclidian distance between the winning neuron and the neighborhood neuron is not considered).

3.3.2.1 Confirmation of proposed algorithm behavior (single peak)

This section uses a comparatively simple example which is a single-peak histogram in order to describe the proposed algorithm operation in detail. This example is borrowed from the case where the position at approximately 70% in the content is frequently viewed. To simulate such viewing frequency, the network input was given to the network under the following conditions.

- $R = \langle p, 1 \rangle$ (corresponds to starting viewing) is observed at the position p randomly selected in a uniform manner from the entire content with 10% probability
- $R = \langle p, -1 \rangle$ (corresponds to ending viewing) is observed at the position p randomly selected in a uniform manner from the entire content with 10% probability
- $R = \langle p, 1 \rangle$ is observed at the position p randomly selected in a uniform manner from position 55 through 65 in the entire content with 40% probability
- $R = \langle p, -1 \rangle$ is observed at the position p randomly selected in a uniform manner from position 75 through 85 in the entire content with 40% probability

The actual input probability density is shown in lines on the upper pane of Fig. 10. The horizontal axis indicates a position in the content indicated as a percentage of the content from the starting position of the entire content. The vertical axis indicates sum of user operations, where start operation is 1 and end operation is -1. Namely, it plots $C(p)$ in Equation 5 for p as the horizontal axis.

The lower pane of Fig. 10 shows how the neuron positions of the network $\mathcal{F}$ changed as inputs are presented to the network by applying the proposed method until 50,000-th input, where the vertical axis indicates the number of inputs t, and the horizontal axis indicates $y(t)$.

It seems that neurons converged after approximately 30,000 network inputs. The final position of each neuron is plotted with circles overlapped on the upper pane of Fig. 10 (the vertical position is meaningless). The plateau in Fig. 10 corresponds to high frequency of viewing; neurons are located centering on this portion.

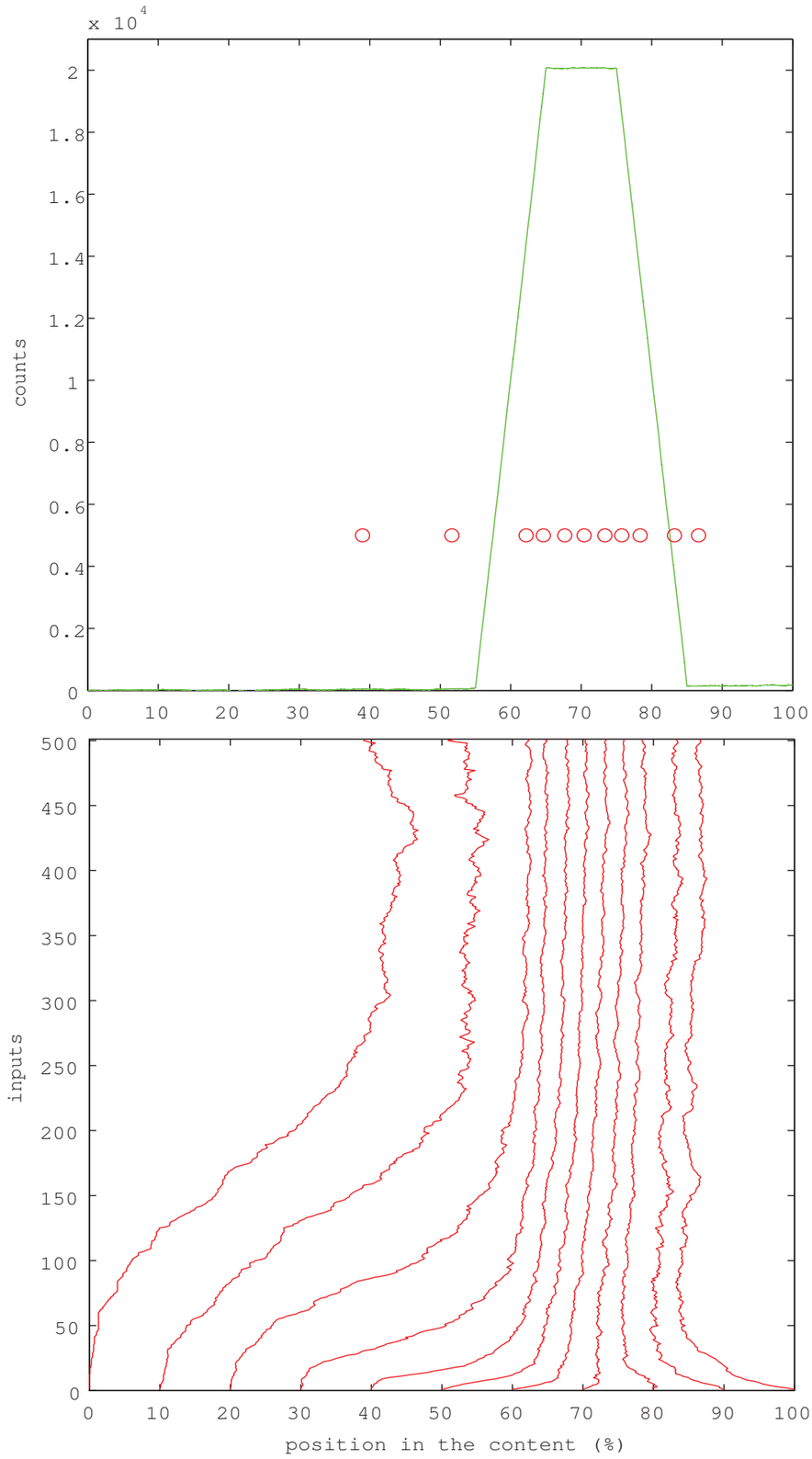


Fig. 10. Result of experiment in section 3.3.2.1.

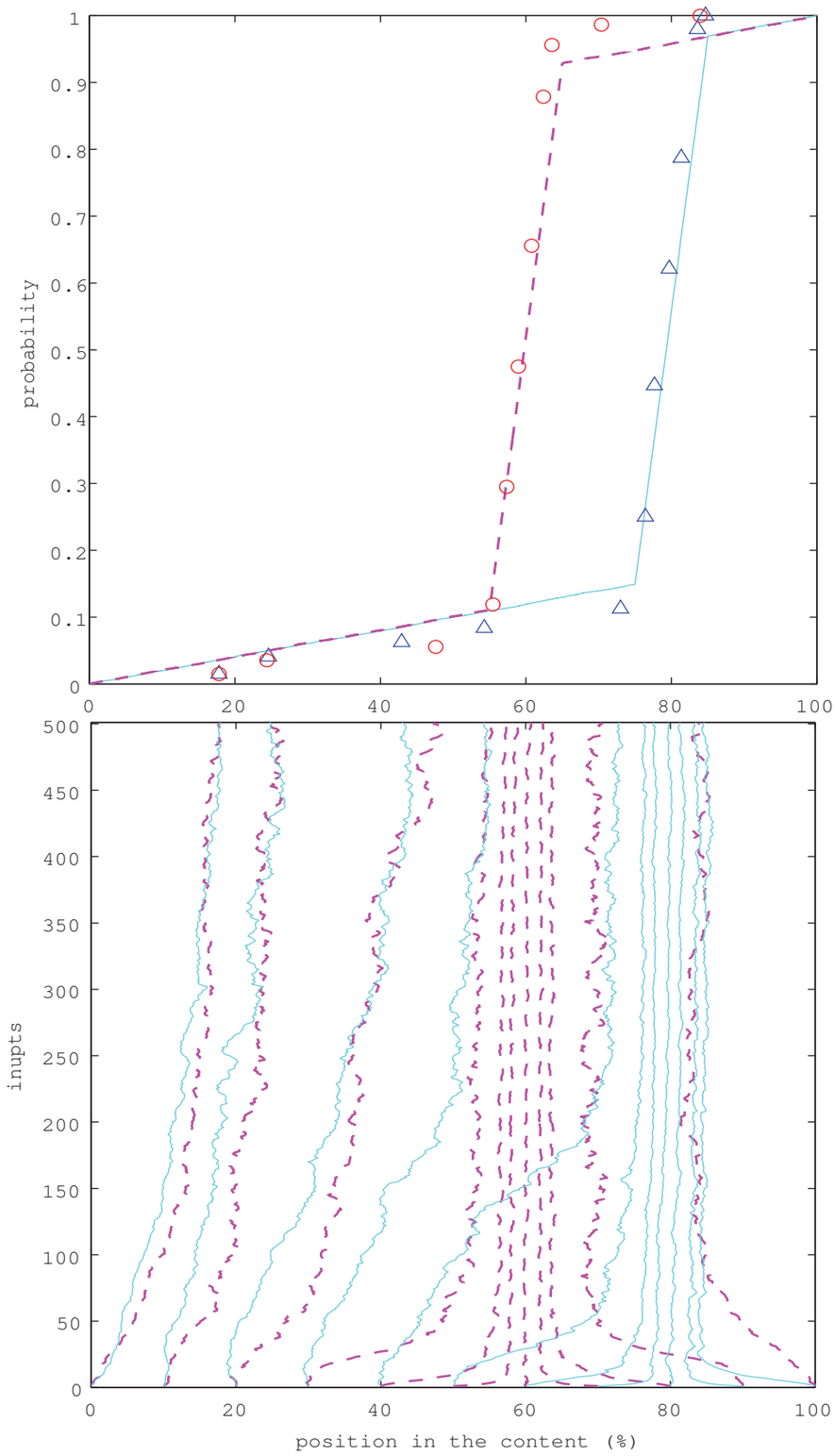


Fig. 11. The final positions of neurons and cumulative frequency of inputs (the upper pane) and the change of positions of neurons during the learning (the lower pane) of SOM $\mathcal{S}$ and $\mathcal{E}$.

The lower pane of Fig. 11 shows the change of positions of neurons during the learning of networks $\mathcal{S}$ and $\mathcal{E}$. The horizontal axis indicates a relative position in the content, while the vertical axis indicates the number of inputs. The dashed line indicates each neuron of network $\mathcal{S}$. It is likely that the neurons converged at a very early stage to the part with high frequency of input.
Here, as described in section 3.3.1.1, each neuron corresponds to scalar quantized bin of the viewing frequency. Accumulated number of neurons from the left approximates the cumulative distribution function.
The upper pane of Fig. 11 shows the cumulative frequency of the network inputs that designates the start points of viewing (Equation 3), in dashed line, and the cumulative frequency of the network inputs that designates the end points of viewing (Equation 4), in solid line (each of them is normalized to be treated as probability).
Moreover, in both of $\mathcal{S}$ and $\mathcal{E}$, the neurons (feature vectors) after learning were plotted with circles by Equation 7 in the upper pane of Fig. 11. Each point approximates the cumulative distribution function with adequate accuracy.

3.3.2.2 Confirmation of proposed algorithm behavior (double peaks)

Fig. 12 shows result of the proposed method applied to a more complicated case when the input frequency distribution has double peaks. The axes are the same as Fig. 10. In this experiment too, neurons converged after around 40,000 network-inputs. We see that neurons are located not around the valley, but around the peaks.

3.3.2.3 Application example of proposed method

The section shows a subject experiment that used an actual viewing history of multimedia content. We had 14 university students as the experimental subjects and used data that was used in Ishikawa et al. (2007).
The content used in the experiment was a documentary of athletes *Mao Asada One Thousand Days* (n.d.), with a length of approximately 5.5 min. We gave the experimental subjects an assignment to identify a part (about 1 sec.), in which a skater falls to the ground only one time in the content, within the content and allowed them to skip any part while viewing the content. Other than the part to be identified (the solution part), the content has the parts related to the solution (the possible solution part), e.g., skating scenes, and the parts not related to the solution, e.g., interview scenes. The percentage of the latter two parts in the whole content length was almost 50%.
The total operations of the experimental subjects were approximately 500, which was overwhelmingly small in number when compared to the number of inputs necessary for learning by the proposed method. For this reason, we prepared the data set whose viewing frequency of each part is squared value as that of obtained from the experiment originally. To circumvent the problem we presented data repeatedly to the network in a random manner until the total of network inputs reached 20,000.
Table 1 shows the experimental results. The far right column shows the distance to the solution part in the number of frames (1 sec. corresponds to 30 frames). We see that neuron 3 came close within approximately 1 sec. of the solution part. Neurons 1 through 3 gathered close to the solution part.
Furthermore, the column named goodness is filled in with circles when each neuron was on the possible solution part; otherwise, with the shortest distance to the possible solution part. Although the number of the circles did not change before and after learning, the average

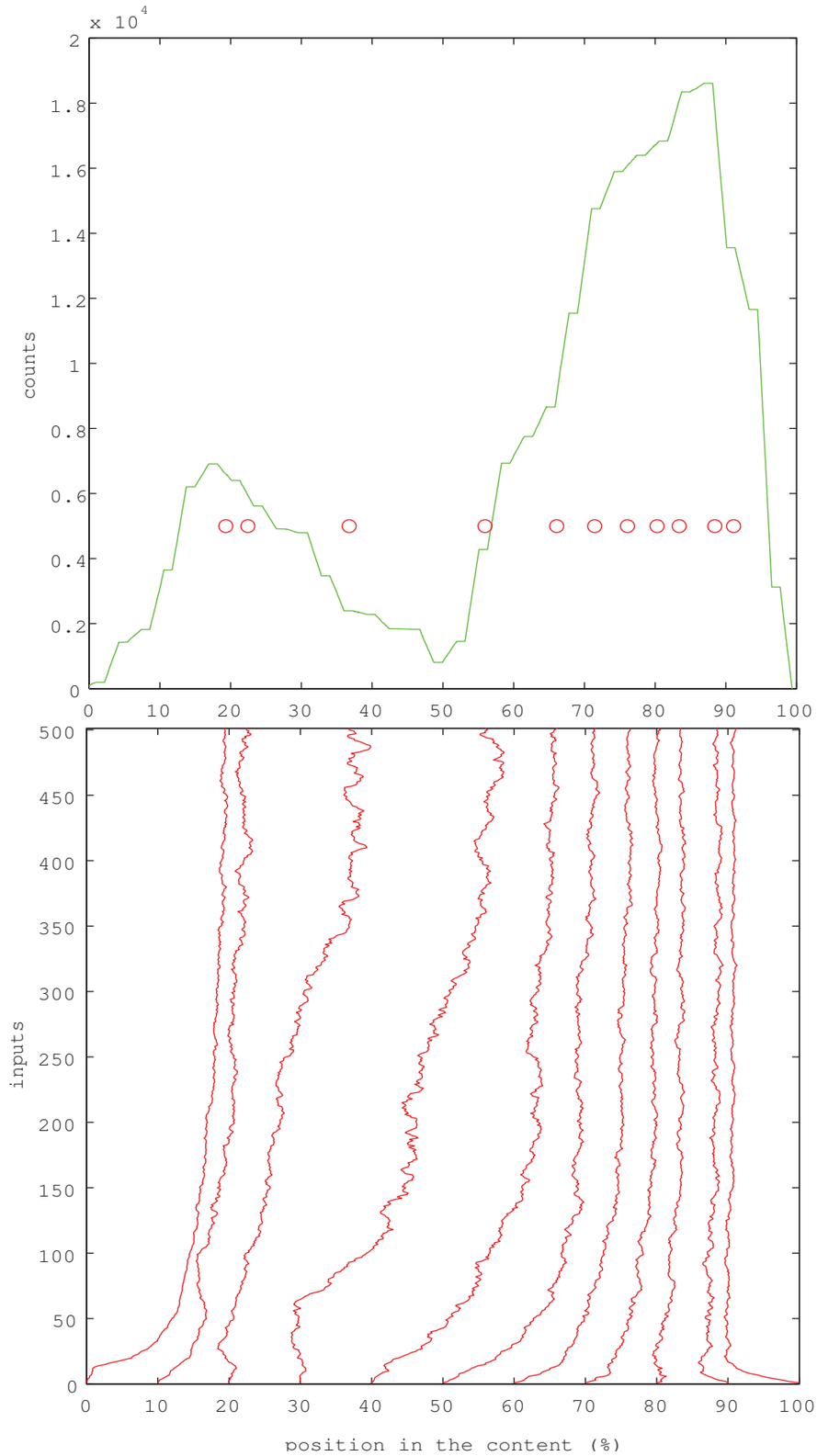


Fig. 12. Result of experiment in section 3.3.2.2.

neuron number	initial positions	converged positions		residuals
	goodness	positions (%)	goodness	(absolute values)
1	248	29.98	433	507
2	130	31.95	231	299
3	10	34.45	◯	35
4	435	41.70	166	704
5	346	50.76	337	—
6	413	56.91	◯	—
7	◯	66.22	13	—
8	◯	70.79	◯	—
9	◯	75.02	◯	—
10	◯	83.95	218	—
11	296	85.85	17	—
average	268.3		202.1	

Table 1. Result of experiment in section 3.3.2.3.

distance to the possible solution part was shortened, and neurons moved to the appropriate parts. Regarding neuron 1, the reason the distance to the possible solution part increased may be that neurons are concentrated on the solution part and it was pushed out. If so, when an adequate amount of data is given, neurons are expected to be located on the solution and possible solution parts.

Fig. 13 shows an example of extracted still images from the part that corresponds to each neuron position.

The proposed method gave a result, i.e., neuron or still image 3, close to the solution as is seen in Table 1 but is not a solution as is seen in Fig. 13. The reason is as follows; (1) as reported in Ishikawa et al. (2007), the existence of experimental subjects that preferred to do their own search and ignored the histogram intentionally could lead to lowering accuracy, (2) the total length of the content time could vary a few seconds because the content was distributed via networks (this was already confirmed under the experimental environment), (3) user operations were mainly done by using the content player's slider; there were fluctuations too significant to realize the accuracy on the second time scale level, and (4) as stated above, the amount of data for learning was not necessarily adequate causing the estimation to be significantly varied. We consider these things to be unavoidable errors.

However, the part identified by using the proposed method is about one frame before the part the skater started her jump. The time point to which this skater's jump lands on is the actual solution part, and this identified part is just 35 frames away from the solution part. Therefore, although the solution part is not extracted as a still image by proposed method, if we extract a content parts starting 2 seconds before the neuron positions and ending 2 second after of them, the solution part is included. Under such condition, the content was summarized to 44 seconds (which is total viewing times, $(2+2)$ sec. $\times 11$ neurons, for a set of partial content parts) from 5.5 minutes; in addition, the solution part was also included. For this reason, we claim that this proposed method is effective from the standpoint of shortening the viewing time.

3.3.3 Discussion

The method proposed in section 3.3 identifies a part that is important in the content automatically and adaptively, based on the accumulated viewing history. As explained

Fig. 13. Still images extracted at neurons' positions.

for example in Ishikawa et al. (2007), mainly in video content, multiple still images (also referred to as thumbnails) captured from a video file could be utilized as a summary of the video. Usually, capture tasks are conducted automatically at a constant time interval, or done manually by users while viewing the video content. The method proposed in this section can also be applied to capturing still images, and it is expected that this proposed method can make it possible to capture those still images in the parts with high frequency of viewing with less task burden.

As introduced in section 2, neurons are located according to the frequency of network-inputs by using the SOM method. However, it is impossible for conventional SOM to learn such data having a range. For this reason, we have formulated a new method that takes advantage of self-organizing features.

The method proposed in section 3.3.1 is a general method to extend SOM to approach the issue where the data to be learned has a range and only the starting or ending points of the range are given as input. We think this method will extend application range of SOM. The

experiments performed confirmed that, based on the data of start/end of viewing obtained by using the method in Ishikawa et al. (2007). This method has made a success to determine most appropriate time for extracting still images that reflect the viewing frequency of the content adaptively, incrementally, and automatically.
A process that corresponds to building a histogram is required in order to estimate the frequency of intervals such as time-series content part that consists of the starting and ending points. We realized this process by using two SOMs ($\mathcal{S}$ and $\mathcal{E}$ described in section 3.3.1.1) and the process shown in Fig. 9. The space and time complexity of the method using RDB and proposed method is as follows; where n is the number of the histogram bins (i.e., the number of partial content parts to be identified).

1. $R(t) = \langle p, m \rangle$ is inserted into the sorted array A:

   $\mathbf{\Omega}(\log t) + \mathbf{\Omega}(\log t)$

2. The cumulative integral array B of the array A is obtained:

   $\mathbf{\Omega}(\log t)$

3. From $b_i = i \times \dfrac{\big(\max(B) - \min(B)\big)}{n} \in B$, $1 \leq i \leq n$, b_i is calculated in order to extract partial content parts:

   $\mathbf{\Omega}(\log t)$

Space complexity is $\mathbf{\Omega}(t)$ (the SQL process described in section 3.1.2 is partly common to the processes of 1 and 2 above). The SOM of the proposed method ($\mathcal{F}$ in the previous section) that does learning which reflects frequency also makes it possible to conduct part of the processes of 2 and 3 above by combining the above mentioned methods.
On the other hand, the process described in this section (SOM $\mathcal{S}$ and $\mathcal{E}$) that estimates the cumulative distribution function by using SOM corresponds to part of the processes of 1 and 2 above. Use of SOM eliminates the need for sorting or determining the portion where data is inserted. The network conducts learning by inputting data observed as it is. Time complexity for this is as follows, independent from t.

4. $\mathrm{argmin}_i(||p - y_i||)$ in the SOM $\mathcal{S}$ or $\mathcal{E}$, and $R = \langle p, m \rangle$ are obtained:

   $\mathbf{O}(n) + \mathbf{O}(n)$

5. The feature vectors of the winning and neighborhood neurons are updated:

   $\mathbf{O}(1)$

6. S' and E' are obtained, z_i is accumulated, and the point e_w which is $Z < 0$ is obtained:

   $\mathbf{O}(n)$

Space complexity is only $\mathbf{\Omega}(n)$. Moreover, with respect to the part of the process 2 and 3, time complexity is as follows, while space complexity is $\mathbf{O}\big(|\mathcal{F}|\big)$.

proposed method		solutions calculated by	residuals
neuron number	converged position	cumulative distribution	(absolute values)
1	38.99	60.68	21.69
2	51.66	63.13	11.47
3	62.19	64.99	2.80
4	64.63	66.66	2.03
5	67.65	68.34	0.69
6	70.42	70.01	0.41
7	73.35	71.70	1.65
8	75.74	73.41	2.33
9	78.34	75.10	3.24
10	83.21	76.98	6.23
11	86.62	79.46	7.16

Table 2. Result of proposed versus true value.

7. One point within the section $[x(t), e_w(t)]$ is selected, and $\mathrm{argmin}_i \left(\|p - y_i\| \right), R = \langle p, m \rangle$ are obtained:

   $\mathbf{O}(|\mathcal{F}|)$

8. The feature vectors of the winning and neighborhood neurons are updated :

   $\mathbf{O}(1)$

With respect to the experiments described in section 3.3.2.1, we compare the result of the proposed method to approximation of the cumulative distribution function with rectangular integration (corresponding to the processes of 1–3 above in this section). Table 2 shows the result, where the open interval $(0, 1)$ was divided into 12 intervals. As for the neurons 3 through 9, the difference was 5% or less; we can say that the proposed method showed good results. However, a significant difference was observed for the four neurons other than the above. As shown in the upper pane of Fig. 10, the proposed method unfortunately locates neurons so as to cover the parts with comparatively low frequency. On the other hand, all results based on the cumulative distribution were located within the plateau shown on the upper pane of Fig. 10; however, from the viewpoint of balance of summarization, we consider that the neurons, or extracting content parts, should not be concentrated to this extent as these results. The difference become larger gradually neuron 5 to 1 and 6 to 11. Thus, it could be possible that neurons 1, 2, 10, and 11 pulled other neurons in the result of the proposed method.

The studies so far on summarization of multimedia content mainly focused on what information can be used for summarizing content and how to summarize by such information (e.g., Yu et al. (2003); Yamamoto & Nagao (2005)). In this chapter, we adopted comparatively simple method, and examined its effectiveness. We proposed a method using SOM, which has smaller time and space complexity, thus a more scalable process could be realized when compared to the method RDB.

Self-organization typified by SOM can be understood as identifying most characteristic elements based on data given by the environment. When self-organization is applied to

functional approximation or clustering, it identifies features of a limited number representing the entire function or all features.
Such property is good for summarization, because, also in summarization, fewer resources represent the all. Therefore, it is possible to regard the proposed method as a self-organizing approach for time-series digital content summarization. The concept of summarizing content by means of self-organization is proposed by this paper for the first time. The method itself is effective; there still remains the room for research on the relationship between self organization and summarization that may lead to extension of the proposed method.
The conventional software programs (e.g., *Video Browser Area61* (n.d.)) that capture still images from multimedia content seem to have difficulty in realizing functions other than the followings; (1) to capture still images at equal (time) intervals, and (2) to capture still image whose position are determined manually. Function (1) is a mechanical process, so that it is impossible to reflect viewing frequency as realized by the proposed method. If it is applied to the experiments described in section 3.3.2.3, whether the solution part is extracted or not depends on chance completely.
By function (2), the solution part will be extracted properly. However, the following two problems remain: the result might depend on a user charged with this particular task, so that the validity of the results should be ensured, and the burden for the identification task is significant.
For these problems, similar to Ishikawa et al. (2007), we proposed a method utilizing "wisdom of crowds." In other words, we expect to obtain good results by majority voting of viewing behavior of comparatively large number of viewers (not particular viewers).
This viewpoint lead to a solution for the second problem of the above mentioned function (2), too. By calculating the users' viewing behavior, the proposed method can naturally identify proper positions for capture; nobody needs to bear the burden for the capture position identification.
Given the above mentioned facts, we consider that the proposed method has the advantages to both functions (1), where it can identify the proper part, and (2), where it does not increase related task burden.
We proposed, in section 3.2, a SOM-like method that uses only a single system as the solution for the problems considered in this section. This method is simple and also clear in theory when compared to the method proposed in section 3.3, whereas the latter realizes a process with smooth convergence. Moreover, although the proposed method in section 3.3 requires learning of the three networks, only the neurons near to the winner are selected and updated their feature vectors. This fact shows that the proposed method in section 3.3 has advantage in the time complexity compared to the method proposed in section 3.2. We are going to have further discussions and examinations about the detailed comparison of the two methods and extension of the method, proposed in section 3.3, with only one network in the future.

4. Conclusion

We proposed in this chapter two kinds of SOM-like algorithm that accepts online input as the start and end of viewing of a multimedia content by many users; a one-dimensional map is then self-organized, providing an approximation of the density distribution showing how many users see a part of a multimedia content. In this way "viewing behavior of crowds" information is accumulated as experience accumulates, summarized into one SOM-like network as knowledge is extracted, and is presented to new users.

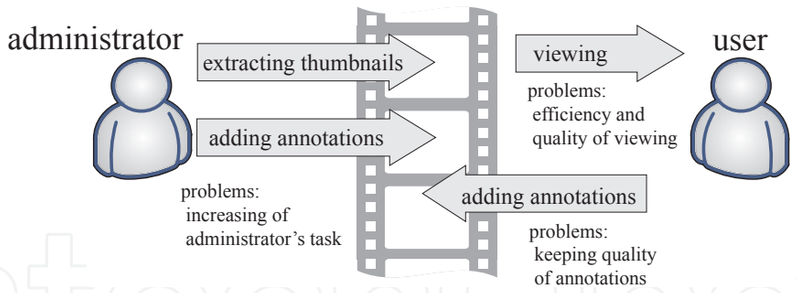


Fig. 14. Problems solved by the method proposed in Ishikawa et al. (2007).

SOM proposed by Kohonen can approximate frequency distribution of events and consequently can be used as a tool to form a "wisdom of crowds" or "collective intelligence" which is one of the important concepts in Web 2.0. But from its nature SOM cannot utilize interval data. However, the proposed algorithms, though limited to a one-dimensional case, approximate density of intervals from their starting and ending events.

Ishikawa et al. (2007) proposed, using RDB, a mechanism of providing only important parts within multimedia content by storing the viewing frequency of many viewers to those users who have just started viewing the content. This method realizes summarization which has been regarded as difficult without causing significant burden on creators of the content (e.g., He et al. (1999)), users, or computers.

In this chapter, we made an attempt to extend this method to promote additional automation. We tried to reduce the computational complexity in order to realize excellent scalability. We took up the concern of adaptively determining a proper time for capturing still images as the application case of the proposed method, and we described how to realize it.

We realized incremental as well as adaptive processes by means of learning, derived from the SOM algorithm. The process of SOM is originally an algorithm that accepts single points given as input; on the other hand, the proposed method extends it to process range data as input.

In terms of the process and concept, the method proposed by this chapter has a close relationship with what is called self-organization. We understand this proposed method as an approach to self-organize time-series digital content. We regard this proposed method as a promising approach, because it is expected that the effectiveness of this proposed method will be greatly improved by deepening our consideration of this relationship.

5. Acknowledgement

The research activity of first author is partly supported by Grant-in-Aid for Scientific Research (C), 20500842, MEXT, Japanese government.

6. References

Bak, P., Tang, C. & Wiesenfeld, K. (1988). Self-organized criticality, *Physical Review A* 38(1): 364–374.

Bonabeau, E., Theraulaz, G. & Dorigo, M. (eds) (1991). *Swarm Intelligence: From Natural to Artificial Systems*, Oxford University Press.

Buchanan, M. (2007). *The Social Atom: Why the Rich Get Richer, Cheaters Get Caught, and Your Neighbor Usually Looks Like You*, Bloomsbury.

Camazine, S., Deneubourg, J.-L., Franks, N. R., Sneyd, J., Theraulaz, G. & E., B. (eds) (2003). *Self-Organization in Biological Systems*, Princeton University Press.

Google (n.d.). `www.google.com`.

Haken, H. (1996). *Principles of Brain Functioning*, Springer.

He, L., Sanocki, E., Gupta, A. & Grudin, J. (1999). Auto-summarization of audio-video presentations, *Proc. of the 7th ACM Intl. Conf. on Multimedia*, pp. 489–498.

Ishikawa, K., Oka, M., Sakurai, A. & Kunifuji, S. (2007). Effective sharing of digital multimedia content viewing history, *Journal of ITE* 61(6): 860–867. in Japanese.

Johnson, S. (2002). *Emergence: The Connected Lives of Ants, Brains, Cities, and Software*, Scribner.

Kauffman, S. (1995). *At Home in the Universe: The Search for the Laws of Self-Organization and Complexity*, Oxford University Press.

Kohonen, T. (2001). *Self-organizing Maps*, 3rd edn, Springer-Verlag.

Krugman, P. R. (1996). *The Self-Organizing Economy*, Blackwell.

Mao Asada One Thousand Days (n.d.).
`www.japanskates.com/Videos/MaoAsadaOneThousandDayspt1.wmv`.

Nonaka, I. & Takeuchi, H. (1995). *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation*, Oxford University Press.

Ohmukai, I. (2006). Current status and challenges of Web2.0: Collective intelligence on Web2.0, *IPSJ Magazine* 47(11): 1214–1221. in Japanese.

O'Reilly, T. (2005). What is web 2.0: Design patterns and business models for the next generation of software. `oreilly.com/web2/archive/what-is-web-20.html`.

Page, S. E. (2007). *The Difference: How the Power of Diversity Creates Better Groups, Firms, Schools, and Societies*, Princeton University Press.

Surowiecki, J. (2004). *The Wisdom of Crowds: Why the Many Are Smarter Than the Few and How Collective Wisdom Shapes Business, Economies, Societies and Nations*, Doubleday.

Tapscott, D. & Williams, A. D. (2006). *Wikinomics: How Mass Collaboration Changes Everything*, Portfolio.

Van Hulle, M. M. (2000). *Faithful Representations and Topographic Maps: From distortion- to information-based self-organization*, John Wiley & Sons.

Video Browser Area61 (n.d.).
`www.vector.co.jp/vpack/browse/pickup/pw5/pw005468.html`.

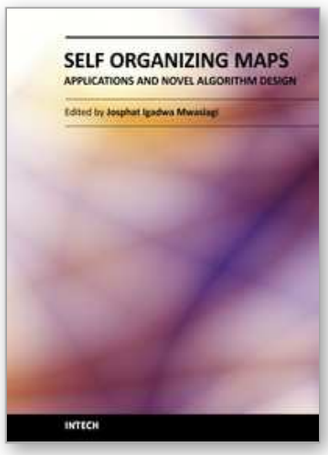
Yamamoto, D. & Nagao, K. (2005). Web-based video annotation and its applications, *Journal of JSAI* 20(1): 67–75. in Japanese.

Yin, H. & Allinson, N. M. (1995). On the distribution and convergence of feature space in self-organizing maps, *Neural Computation* 7(6): 1178–1187.

YouTube (n.d.). `www.youtube.com`.

Yu, B., Ma, W., Nahrstedt, K. & Zhang, H. (2003). Video summarization based on user log enhanced link analysis, *Proc. of the 11th ACM Intl. Conf. on Multimedia*, pp. 382–391.

Zhabotinsky, A. M. (1991). A history of chemical oscillations and waves, *Chaos* 1(4): 379–386.



Self Organizing Maps - Applications and Novel Algorithm Design

Edited by Dr Josphat Igadwa Mwasiagi

ISBN 978-953-307-546-4
Hard cover, 702 pages
Publisher InTech
Published online 21, January, 2011
Published in print edition January, 2011

Kohonen Self Organizing Maps (SOM) has found application in practical all fields, especially those which tend to handle high dimensional data. SOM can be used for the clustering of genes in the medical field, the study of multi-media and web based contents and in the transportation industry, just to name a few. Apart from the aforementioned areas this book also covers the study of complex data found in meteorological and remotely sensed images acquired using satellite sensing. Data management and envelopment analysis has also been covered. The application of SOM in mechanical and manufacturing engineering forms another important area of this book. The final section of this book, addresses the design and application of novel variants of SOM algorithms.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen