Visualization of Massive Streaming Data using an Artificial Neural Network

Arunim Talwar

Dr. Masahiro Takatsuka

School of Information Technologies

FACULTY OF ENGINEERING & INFORMATION TECHNOLOGIES

Aims

- To develop a Data Visualization solution which is able to efficiently handle large-scale streaming data
- We will build this Visualization by building upon the existing Self-Organizing Map algorithm, as described by T. Kohonen
- The SOM is an example of an unsupervised Artificial Neural Network, and by expanding upon it, we will prove that we can use an ANN to create a visualization of high-dimensionality streaming data

Problem & Motivation

Data Visualization

Data Visualization is an incredibly useful way of capturing the statistical nature of a set of data, and presenting that analysis in a way which is understandable to most people. Often, analysis of data can lead to complex statistical models, where the final results may be rich in content and incredibly beneficial, but those benefits are not able to be communicated or used in a substantial way.

With Data Visualization, one can create a graphical representation of highly complex data, such that it is more digestible to the common reader, and can therefore have a larger impact on the industry around it.

Large-Scale Streaming Data

The field of Data Visualization is one which has seen a large amount of research in recent years. However, most of that research has dealt with creating visualization solutions for constant-size databases – as in, the amount of data is usually known beforehand.

When it comes to data where the database size is NOT known beforehand – for example, when the data is being streamed in real-time – there has yet to be an elegant solution to the visualization problem.

Large-scale streaming data, in this context, refers to data of a high dimensionality (as in, data with several features), which is communicated in real-time over a network.

Artifical Neural Networks

Many of the existing solutions to dealing with traditional data visualization (i.e. data that is not massive streaming data), involve using Artificial Neural Networks. However, these solutions tend to scale poorly when dealing with large-scale streaming data.

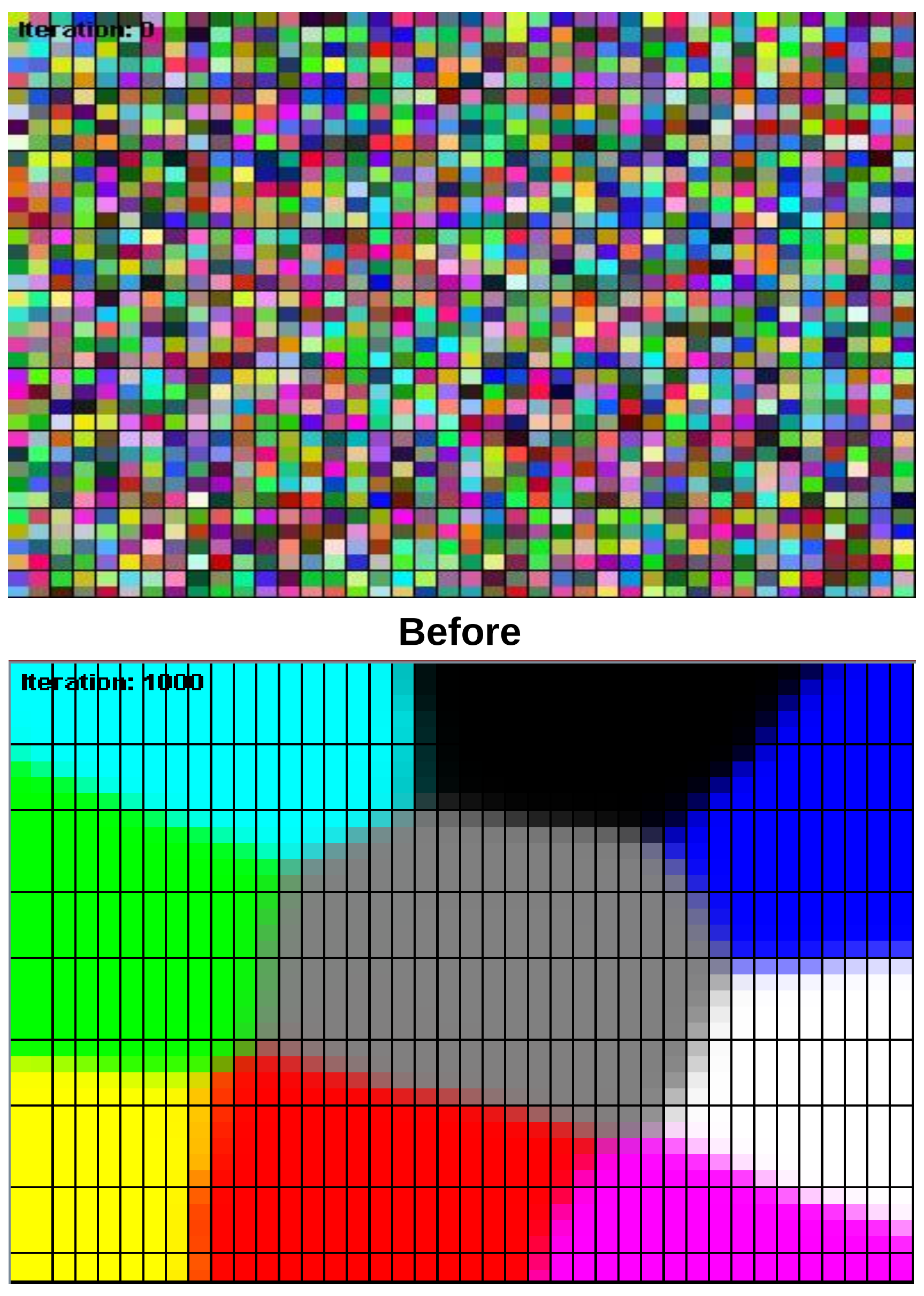
We want to create a Visualization solution which is based on existing Artificial Neural Network algorithms, and is able to scale to deal with massive streaming data.

Methodology

In order to build our Artificial Neural Network solution, we need to first look at the existing ANN solutions, and then see how we can expand upon them in order to accommodate data of high dimensionality, that is streamed over a network.

Self-Organizing Map

Kohonen's Self-Organizing Map (1982) is an Artificial Neural Network algorithm which allows for a two-dimensional representation of high-dimensionality data. By making use of a *neighbourhood function*, a map can be created where the topological properties of the input space are conserved – that is, statistically similar data points are grouped together in the map.



Before

After

Fig. 1. An example of a Self-Organizing Map where the similarly-coloured nodes end up grouped together

Growing Self-Organizing Map

The problem with using a traditional Self-Organizing Map is that the dataset must be known in it's entirety before we can train the map. Obviously, this is not suitable for streaming data, as data is arriving in real-time and we must be able to grow our map accordingly.

As such, we can adapt the Self-Organizing Map algorithm to allow it to be grown as new data is presented to the network.

The resulting algorithm is called a **Growing Self-Organizing Map**.

While there already exists an algorithm for the Growing Self-Organizing Map already, we have altered the algorithm, and our implementation of it, to account for the fact that the data being presented to the network is being *streamed*.

Implementation

Implementation of the Algorithm

In order to implement the Growing Self-Organizing Map algorithm, we needed to make sure that we not only preserved the mechanics of the existing algorithm, but that we are also able to handle streaming data.

To that end, we initially coded the basic algorithm as it is best known. We initialize four random nodes (that is, with random numbers between 0 and 1), and then calculate the *growth threshold* using the dimensionality of the data and the data's *spread factor*.

After this, we implemented the Growth Phase, which involves presenting the input to the network and actually growing and training the map, and the Smoothing Phase, which allows us to remove any noise from the visualization.

Once we had the basic algorithm implemented, we had to adapt it to deal with streaming data. In order to deal with streaming data, we implemented a system where the first time the network was trained, the data was picked up directly from the data stream, for X number of iterations (where X can be any positive whole number). Once this initial visualization is created, the next time the network is trained, the new data nodes are added on to the existing network, for Y number of iterations (where Y can be any positive whole number).

X and Y serve to act as constraints on the network, since we can theoretically get infinitely large maps otherwise (since data can keep being streamed to the map).

Adaptation of the Streaming Data

The data we used for our experiments is data which pertains to consumer usage of an online video application **(NB: The source of this data is currently classified)**. The data took the form of a series of data nodes, where each data node had a high number of numeric and non-numeric (that is, categorical) dimensions.

In order to use the data to our algorithm, we had to convert it into a form understandable by the algorithm. As such, we converted the categorical data into numeric data using simple rules and checks, which came from a contextual understanding of the data. Once we had all numeric data, we *normalized* the data, such that our numbers were on roughly the same scale.

In order to receive the data itself, we made use of the Splunk API in Java.

Results

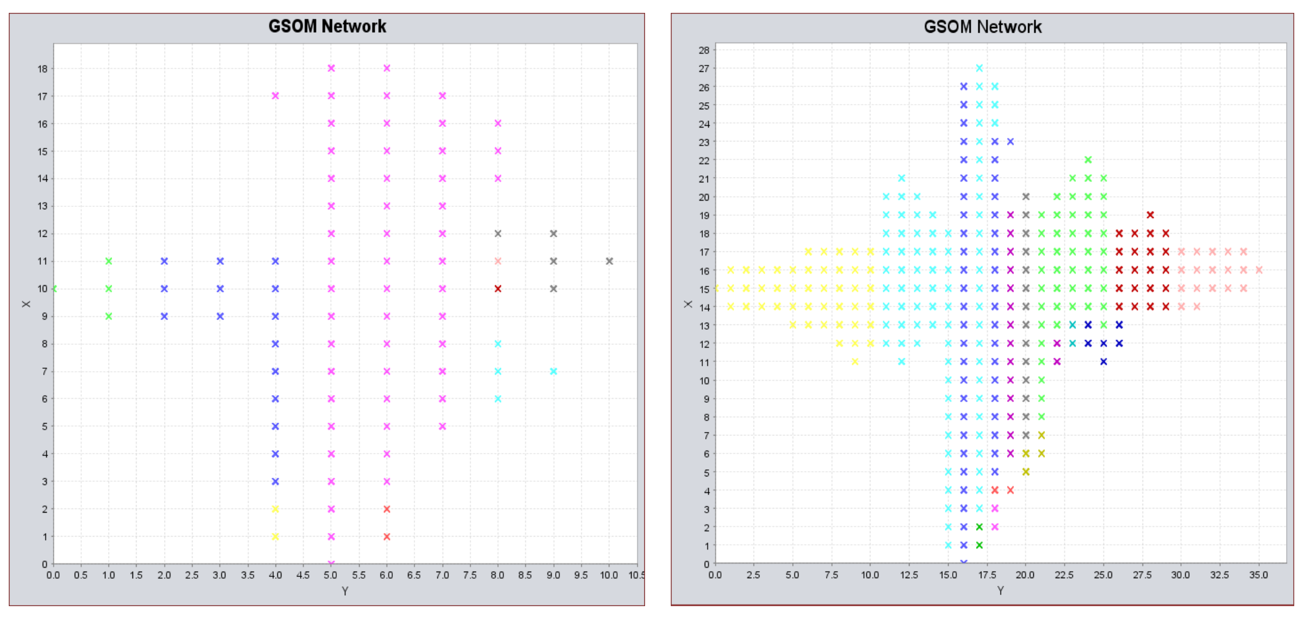


Fig. 2. An initial visualization (**left**), and a second visualization (**right**), of the streaming data presented to the network

Conclusion

Our results show that we can, in fact, use an Artificial Neural Network to create visualization of massive streaming data. Such a visualization would be immensely useful when analyzing applications which interact with people in real-time, or for systems which have high-dimensionality data that needs to be represented in a low-dimensionality plane.