EPiC Series in Computing

Volume 41, 2016, Pages 303–313

GCAI 2016. 2nd Global Conference on Artificial Intelligence

A Sparse Representation of High-Dimensional Input Spaces Based on an Augmented Growing Neural Gas

Jochen Kerdels and Gabriele Peters

University of Hagen - Chair of Human-Computer Interaction
Jochen.Kerdels@FernUni-Hagen.de,
WWW home page: http://mci.fernuni-hagen.de
Universitätsstrasse 1, 58097 Hagen - Germany

Abstract

The growing neural gas (GNG) algorithm is an unsupervised learning method that is able to approximate the structure of its input space with a network of prototypes. Each prototype represents a local input space region and neighboring prototypes in the GNG network correspond to neighboring regions in input space. Here we address two problems that can arise when using the GNG algorithm. First, the GNG network structure becomes less and less meaningful with increasing dimensionality of the input space as typical distance measures like the Euclidean distance loose their expressiveness in higher dimensions. Second, the GNG itself does not provide a form of output that retains the discovered neighborhood relations when compared with common distance measures. We show that a GNG augmented with *local input space histograms* can mitigate both of these problems. We define a sparse vector representation as output of the augmented GNG that preserves important neighborhood relations while pruning erroneous relations that were introduced due to effects of high dimensionality.

1 Introduction

Large, high-dimensional datasets can be difficult to analyze and process if little to no knowledge on potential structures or properties is available beforehand. In some cases unsupervised learning algorithms can facilitate the analysis of such data as they are able to discover certain structures like neighborhood relations without any apriori knowledge. One class of such algorithms are *topology representing networks* [9]. These algorithms employ forms of unsupervised, competitive Hebbian learning to approximate the structure of an input space with a network of units where each unit is typically associated with a *prototype* or *reference vector* that represents a local region of input space.

The growing neural gas (GNG) is one instance of this class of algorithms [2]. It uses a data-driven growth process to incrementally build a prototype-based network model of its input space where the resulting network structure forms an *induced Delaunay triangulation* of that space. This contrasts the GNG algorithm from similar approaches like the self-organizing map [7], which use a fixed network topology. On the one hand, this added flexibility allows the GNG to reflect the input space structure in more detail. For example, the GNG network can develop

disjoint sub-networks that indicate possible classes in the input data. On the other hand, if the input data originates from a high-dimensional manifold, the expressiveness of the resulting GNG network depends critically on the expressiveness of the used distance measure. In the default case, i.e., when the Euclidean distance is used, the GNG network tends to become fully connected losing most of its expressiveness. A second shortcoming of the GNG approach is the lack of an output representation that preserves the discovered neighborhood relations such that the output can be used in subsequent stages of processing that rely on common distance measures.

To mitigate the first problem the concept of *local input space histograms* (LISH) was recently introduced [5, 6]. Local input space histograms can augment prototype-based network models like the GNG by collecting additional information about the input space structure that is associated with the edges of such networks. Here we show that the additional information provided by the LISHs can also be used to mitigate the second problem. Based on the additional LISH information it is possible to form a sparse output representation of individual GNG units that retains important neighborhood relations discovered by the GNG and makes this information accessible to common distance measures. We explore the properties of this sparse representation by a series of experiments in which we try to discover neighborhood relations in the MNIST database of handwritten digits [8] in an unsupervised fashion.

The paper is organized as follows. The next section will briefly revisit the growing neural gas algorithm and point out specific problems of the GNG with respect to high-dimensional input spaces. Section 3 describes the concept of local input space histograms and introduces the *LISH average bin error* as one measure to utilize the additional information provided by the LISHs. Based on this measure a novel, sparse representation for the units in a prototype-based network model is proposed and investigated in section 4. Finally, the paper concludes with section 5.

2 Growing Neural Gas

The growing neural gas (GNG, [2]) is an unsupervised learning algorithm that can generate a prototype-based model of an input dataset or its *input space*. Here we provide a short, formal description of the GNG algorithm as presented by us in Kerdels and Peters, 2015 [6].

The growing neural gas is a network that consists of a set A of units and a set C of edges. Each unit $a \in A$ can be described by a tuple[1] (w, e) with the *prototype* $w \in \mathbb{R}^n$, n being the dimension of the input space, and the accumulated error variable $e \in \mathbb{R}$. Each edge $c \in C$ can be described by a tuple (a, b, t) with the units $a, b \in A \wedge a \neq b$ that are connected by the edge and the variable $t \in \mathbb{N}$ which stores the current age of the edge. The direct neighborhood D_a of unit a is defined as $D_a := \{b | \exists\,(a, b, t) \in C,\, b \in A,\, t \in \mathbb{N}\}$. The network is initialized with two units that have random prototypes and accumulated error variables set to zero.
A given input $\xi \in \mathbb{R}^n$ is processed by the network in the following way:

- Find the two units s_1 and s_2 whose prototypes are closest to the input ξ:

  $s_1 := \text{argmin}\,\{a.w - \xi | a \in A\}\,, \quad s_2 := \text{argmin}\,\{a.w - \xi | a \in A \setminus \{s_1\}\}.$

- Increment the age of all edges connected to s_1:

  $c.t := 0, \quad c \in C \ \wedge\ c.a = s_1 \ \wedge\ c.b = b, \forall b \in D_{s_1}.$

- If no edge exists between s_1 and s_2, create one:

[1] The notation $t.x$ is used to reference the element x within the tuple t.

$C := C \cup \{(s_1, s_2, 0)\}$.

- Reset the age of the edge between s_1 and s_2 to zero:

  $c.t := 0, \quad c \in C \ \wedge \ c.a = s_1 \ \wedge \ c.b = s_2.$

- Add the squared distance between the input ξ and the prototype of unit s_1 to the accumulated error of s_1:

  $\Delta s_1.e := \|s_1.w - \xi\|^2$

- Adapt the prototype of s_1 and all prototypes of its direct neighbors $b \in D_{s_1}$:

  $\Delta s_1.w := \epsilon_b \left(\xi - s_1.w\right), \quad \Delta b.w := \epsilon_n \left(\xi - b.w\right), \ \forall b \in D_{s_1}.$

- Remove all edges with an age above a given threshold $t_{\max}$ and remove all units that no longer have any edges connected to them.

- If an integer-multiple of λ inputs was presented to the network insert a new unit r. The new unit is inserted between the unit $q \in A$ with the maximum accumulated error and the unit $f \in D_q$ which has the largest accumulated error among the neighbors of q, i.e., the prototype of unit r is set to:

  $r.w := (q.w + f.w)/2.$

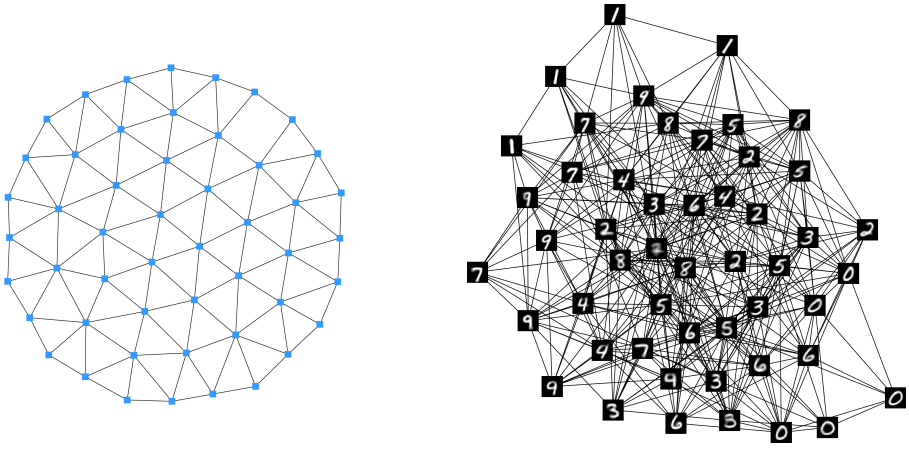
  Create edges between q and r as well as f and r, and remove the edge between the units q and f. Decrease the accumulated errors of q and f by a factor α and set the accumulated error of the new unit r to the decreased accumulated error of unit q.

- Finally, decrease the accumulated error of all units in A by a factor β.

Typically, the inputs ξ are randomly sampled from the input space and fed into the network until a given halting criterion (e.g., a maximum network size) is met. Alternatively, the algorithm can operate in an online fashion where the network is continuously updated with each further input but does not grow beyond a maximum number of units. The latter approach was used here. In all experiments described below the following parameter values were used:

$$\begin{array}{llllllll} \epsilon_b & = & 0.05, & \epsilon_n & = & 0.0005, & t_{\max} & = & 300, \\ \lambda & = & 1000, & \alpha & = & 0.5, & \beta & = & 0.0005. \end{array}$$

Once the GNG has processed a sufficient number of samples from the input space and has reached a given maximum size the resulting GNG network forms an *induced Delaunay triangulation* of the respective input space. Figure 1 shows two examples of such GNG networks for two different input spaces. In figure 1a the network represents a uniformly distributed, two-dimensional, circular input space. The position of each unit in the figure (blue squares) corresponds directly to the input space position encoded in the particular prototype vector. In this low-dimensional case each unit has a narrow local neighborhood that facilitates a clear distinction between units being close together or far apart. In contrast, the GNG network in figure 1b represents a high-dimensional input space based on the MNIST database of handwritten digits [8]. Each input is a 784-dimensional vector containing 28×28 pixel values. The position of each unit in the figure is determined by a common force-based graph drawing algorithm [3] and is visualized by a bitmap representation of the corresponding prototype vector. In this high-dimensional case the discrimination of units based on their local neighborhoods becomes more difficult as the local neighborhoods increase in size, which in turn decreases the average



(a) (b)

Figure 1: Examples of growing neural gas (GNG) networks with 50 units for two different input spaces. (**a**) A GNG network that processed inputs from a uniformly distributed, two-dimensional, circular input space. (**b**) A GNG network that has processed inputs from the MNIST database of handwritten digits where each input is a 784-dimensional vector (28×28 pixels).

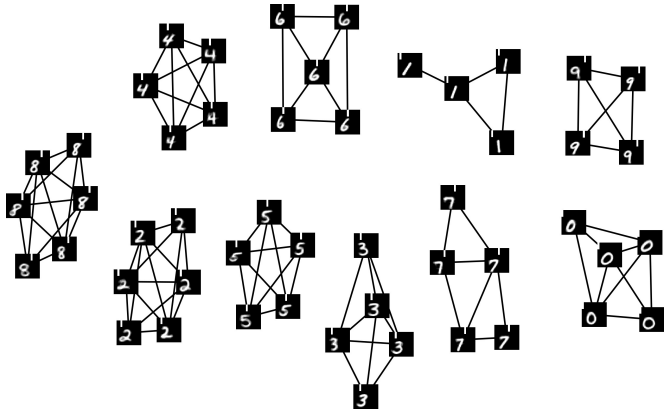


Figure 2: GNG network of an input space based on the MNIST database of handwritten digits where each input was augmented with the corresponding class of the shown digit (white mark above the digit represents the class).

path length between any two units. This increase in neighborhood size is caused by a loss of expressiveness of the employed distance measure – the Euclidean distance – with increasing dimensionality.

Common approaches to mitigate this problem include the use of non-standard distance metrics [1, 4, 10] or switching to a supervised learning strategy where extrinsic information augments the input samples and guides the algorithm to infer which samples are similar and

which are not. The result of such a supervised strategy is shown in figure 2. Here, the inputs based on the MNIST dataset were augmented by a mark that corresponds to the digit's class. Accordingly, the GNG was able to form 10 distinct groups that match these classes. However, this approach requires a labeled training dataset which is not available in the general case, i.e., when unsupervised learning is required.

Recently, we introduced an alternative approach to mitigate the problem of discriminating high-dimensional prototypes in prototype-based network models called *local input space histograms* (LISH) [5, 6]. In essence, a LISH collects additional information on the distribution of input samples that lie between two prototypes of a model and ties this information to the edge that connects these prototypes. The idea of a LISH, which is described in detail in the next section, can be applied to any unsupervised learning algorithm that uses a network of prototypes to model an input space and can augment approaches that use more advanced, non-standard distance metrics as well.

3 Local Input Space Histograms

Prototype-based network models approximate an input space piecewise by a set of prototypes and their relation to one another. In case of a GNG an edge between two GNG units indicates that the input space between the corresponding prototypes is not empty. However, the GNG edges do not convey any further information about the possible structure of the underlying input space. To gather more information in this regard the concept of *local input space histograms* (LISH) was introduced [5, 6]. A LISH is a small histogram $H = \{h_0, \ldots, h_{k-1}\}$, e.g., with $k = 16$ bins, that is added to each edge $c \in C,\ c = (a, b, t, H)$ of a GNG network. With each input ξ the LISH on the edge between the best and second best matching units s_1 and s_2 is updated based on a distance ratio r:

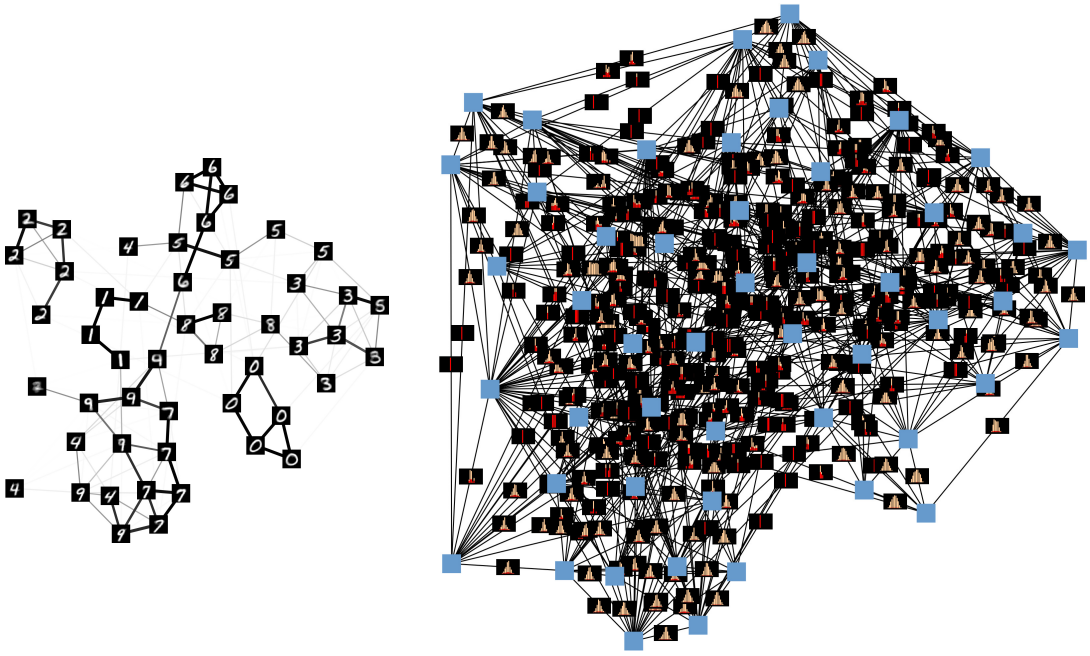
$$r := \frac{\|s_1.w - \xi\| - \|s_2.w - \xi\|}{\|s_1.w - s_2.w\|} + 1,$$

with $s_1.w$ and $s_2.w$ the prototypes of the BMUs with respect to the input ξ. The ratio r ranges from 0 to 1 and describes the relative distance of the current input ξ to the prototypes of the best and second best matching units. If r is close to zero, the input ξ lies close to the prototype of unit s_1. If r is close to one, the input ξ has about the same distance to units s_1 and s_2, i.e., the input lies either halfway between or very far away from the prototypes. Since every LISH $c.H$ is shared by the two units $c.a$ and $c.b$ that are connected by the edge c, the ratio r is used to either update the upper or the lower half of the histogram $c.H$ depending on which of the units $c.a$ or $c.b$ is the BMU s_1:

$$\Delta h_u = 1, \quad u = \begin{cases} \lfloor k\,(r/2) \rfloor & \text{if } c.a = s_1, \\ \lfloor k\,(1 - r/2) \rfloor & \text{if } c.b = s_1, \end{cases} \quad h_u \in c.H = \{h_0, \ldots, h_{k-1}\}\,.$$

As a result, the set of partial histograms associated with a single unit a that lie on all edges connected to a collectively represent the distribution of inputs in the Voronoi cell surrounding the prototype of unit a. Therefore, a single histogram $c.H$ of a particular edge c combines the partial distributions of inputs from two neighboring Voronoi cells. It represents the distribution of inputs that lie approximately between the units $c.a$ and $c.b$.

This additional information on the input space structure can be used in various ways. For example, it can be used to approximate the input space density between two connected units $c.a$



(a) (b)

Figure 3: Example of using local input space histograms to improve the force-based graph drawing of the GNG network shown in figure 1b. (**a**) Using the LISH average bin error passed through a non-linearity to determine the strength of the network edges. The resulting drawing reveals more details of the input space structure compared to the graph rendered with equal edge weights (fig. 1b). (**b**) Same graph as in (a) but with all edges and local input space histograms visible. GNG units represented by light blue squares. Light orange bars in histograms represent counts, dark red bars in histograms represent square roots of counts.

and $c.b$ by calculating the *average bin error* $\bar{e}_H$ of the histogram $c.H$:

$$\bar{e}_H := \frac{1}{k}\sum_{i=0}^{k-1} e_i, \quad e_i := \begin{cases} \sqrt{h_i}/h_i & \text{if } h_i > 0, \\ 1 & \text{if } h_i = 0, \end{cases} \quad h_i \in H = \{h_0, \ldots, h_{k-1}\}\,.$$

In case the density of the respective input space region is low, the value of $\bar{e}_H$ will be near one. If the density is high, the value of $\bar{e}_H$ will be near zero.

It was previously shown that visualization and clustering of GNG-based representations can be improved by using the LISH average bin error to estimate the density of input space regions [6]. One example is shown in figure 3a where the strength of the network edges was determined by passing the LISH average bin error of each edge through a non-linearity. Compared to the force-based graph drawing of the same network shown in figure 1b the LISH-weighted version reveals much more details of the underlying input space structure. Figure 3b shows the same graph as in figure 3a but with all edges and local input space histograms visible. The histograms display a wide variety of distributions including those that indicate a sparse input space between the corresponding units.

4 Sparse Representation

The previous section introduced the concept of local input space histograms and referred briefly to their benefit for visualization and clustering of GNG-based input space representations [6]. Another use case of prototype-based models is vector quantization where the set of units is used as a finite set of discrete entities to which the inputs are mapped. Typically, the output of this quantization for a particular input ξ is either the prototype of the respective BMU or a one-hot encoded representation of the BMU itself. The former output format can be used for tasks like reconstructing incomplete or noisy inputs, whereas the latter format is more suitable for, e.g., building histograms that describe the distribution of inputs. However, both output formats are not particularly useful in cases where the output of the GNG is used as input to a subsequent processing stage as neither output format preserves any neighborhood relations discovered by the GNG. Though it may be possible to construct a ranking of nearest prototypes according to a given distance measure when using the prototype-based output, such a ranking would be of limited value in the high-dimensional case as the actual similarity of the different entries is not described well by common distance measures like the Euclidean distance.

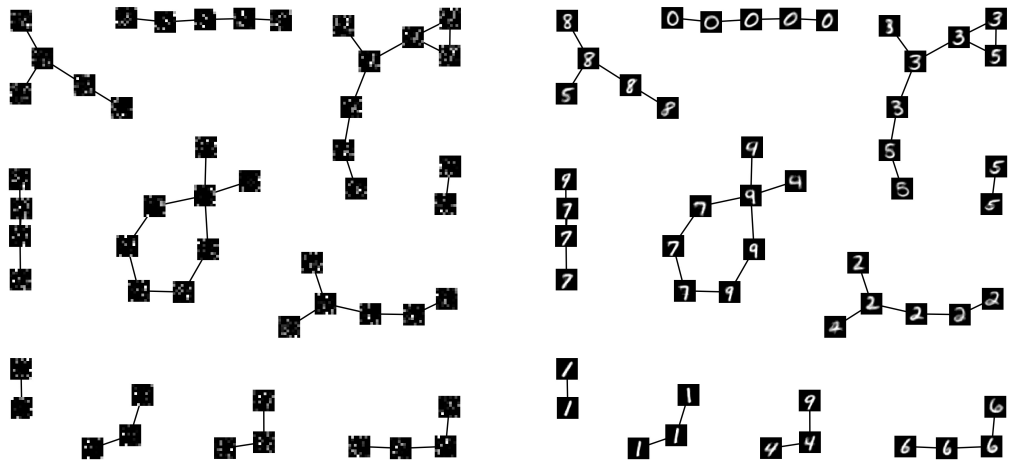
As a possible solution to this problem we propose a novel, *relaxed* one-hot encoding that represents not only a single unit of the GNG but also its direct neighborhood. Given a GNG with M units $\{a_0, \ldots, a_{M-1}\}$ we define the output vector $y := (y_0, \ldots, y_{M-1})$ of the GNG in response to an input ξ as:

$$y_i := \begin{cases} 1 & \text{if} \quad a_i = s_1, \\ 1 - \bar{e}_{c\bullet H} & \text{if} \quad c \in C \ \wedge \ c\bullet a = s_1 \ \wedge \ c\bullet b \in D_{s_1}, \\ 0 & \text{otherwise,} \end{cases}$$

with $i \in \{0, \ldots, M-1\}$, s_1 the BMU with respect to input ξ, and D_{s_1} the direct neighborhood of s_1 as defined above. In this representation the additional information provided by the local input space histograms is used to describe the direct neighborhood of a unit a in terms of the average bin error $\bar{e}_H$, i.e., the estimated input space density. If the input space between unit a and one of its neighbors b is dense, the corresponding entry in the output vector will be close to one. If the input space is sparse, the entry will be close to zero. As a consequence, output vectors of units that are direct neighbors or that share common neighbors will overlap in the corresponding entries allowing regular distance measures to detect these relationships. Using the LISH average bin error to quantify the strength of the neighborhood relations emphasizes those relations that are backed up by a continuum of input samples spanning the input space region between the particular units.

To investigate the proposed encoding we trained a GNG h with samples from the MNIST database and fed the output of h into a second GNG r using the relaxed one-hot encoding described above. Figure 4 shows the resulting structure of GNG r when both h and r have 50 GNG units. Subfigure 4a renders the prototypes of r as bitmaps with high values represented by light colors and low values represented by dark colors. The shown prototypes reveal the sparse patterns that result from the relaxed one-hot encoded input. As expected, neighboring units clearly exhibit similar sparse patterns corresponding to neighboring local regions in the original MNIST input space. To illustrate the latter relation figure 4b shows prototypes of GNG h superimposed on the network of GNG r. Each prototype was chosen according to the highest entry in the prototype vector of the respective unit in r. The overlay substantiates that neighboring units in the second GNG r do indeed refer to neighboring regions in the original MNIST input space of the underlying GNG h.

Since both GNGs h and r have the same number of units, the second GNG r can allocate a single unit for each possible input pattern originating from h. As a consequence, the edges in r

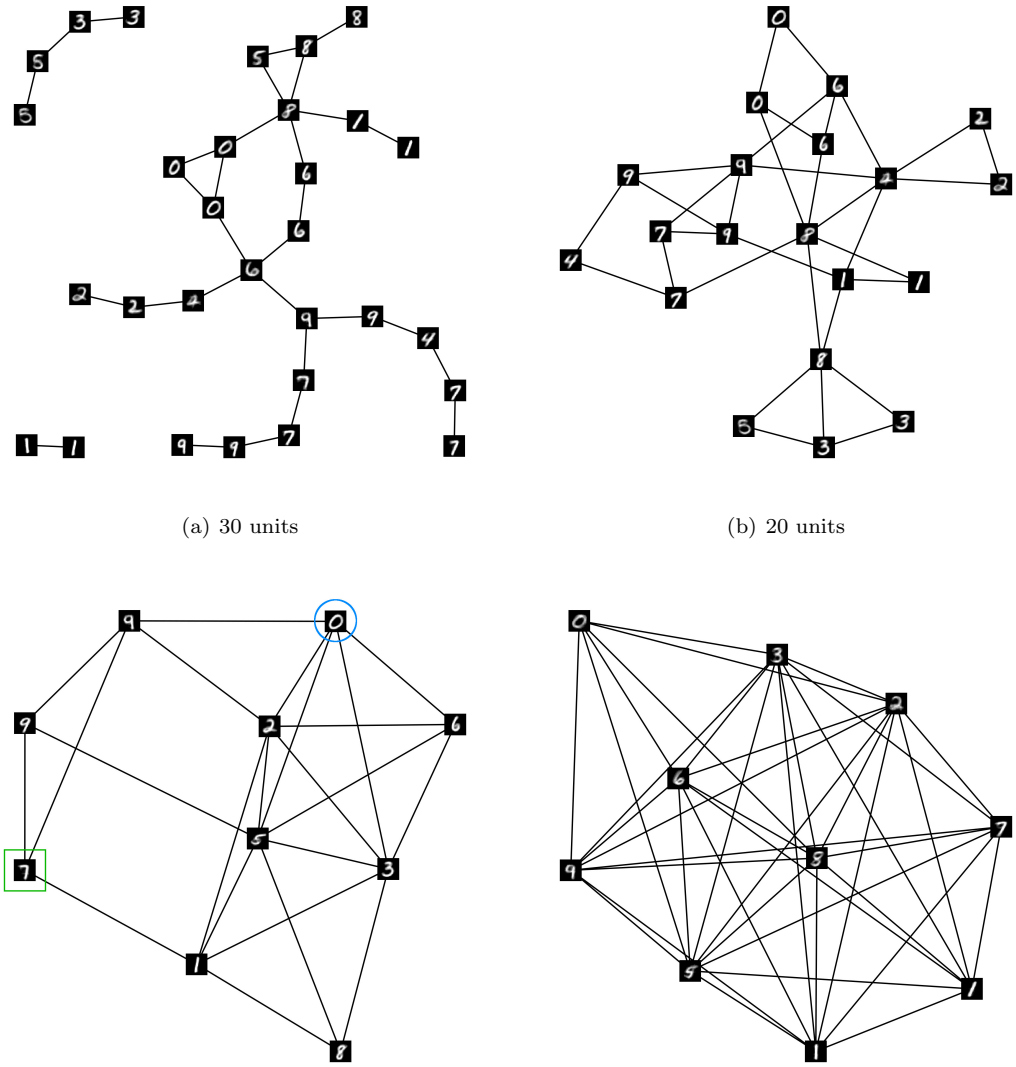


(a) (b)

Figure 4: Example of a GNG r that processed the relaxed one-hot encoded output of a GNG h that processed samples from the MNIST database of handwritten digits. (**a**) Prototypes of GNG r reveal the sparse patterns resulting from the relaxed one-hot encoded outputs of GNG h. (**b**) Prototypes of GNG h superimposed on the network of GNG r. The shown prototypes correspond to the entry with highest value in the respective prototypes of GNG r.

form mostly chains of units that refer pairwise to the most similar units in h with regard to the LISH average bin error between these units. Thus, it remains to be shown if the relaxed one-hot encoding can retain meaningful information on the relation of units in GNG h if the number of units in GNG r decreases and the resulting prototypes in r have to represent more than one unit of GNG h, i.e., a bigger local region of the underlying input space. Figures 5a-c show the networks of three GNGs $\{r_{30}, r_{20}, r_{10}\}$ with decreasing numbers of units that processed the relaxed one-hot encoded outputs of an underlying GNG h with 50 units trained on the MNIST input space. As in figure 4b the shown prototypes originate from the underlying GNG h and are chosen according to the highest entry in the respective prototype vector of GNG r_i. The resulting network structures indicate that meaningful neighborhood relations can indeed be retained by the respective prototypes of the GNGs r_i. Even in the case of GNG r_{10} with only 10 units (fig. 5c) the average degree of the units remains low and the shown prototypes of neighboring units exhibit plausible similarities. In contrast, if the MNIST input space is processed directly by a GNG with only 10 units, the average degree per unit is significantly higher such that almost every unit has every other unit as direct neighbor (fig. 5d).

To provide a more detailed look at the composition of prototype vectors in GNG r_{10} figure 6 shows all vector entries with a value above 0.05 for two prototype vectors of r_{10}, which are marked by a blue circle and a green square in figure 5c. In addition to their numerical value the shown entries are supplemented with renderings of the corresponding prototypes from the underlying GNG h. The entries are sorted left-to-right and top-to-bottom by value in descending order. The degree of similarity between the prototype associated with the highest value (top-



(a) 30 units

(b) 20 units

(c) 10 units

(d) 10 units direct MNIST

Figure 5: (**a-c**) Three GNGs (*a*) r_{30}, (*b*) r_{20}, and (*c*) r_{10} with decreasing numbers of units that processed the relaxed one-hot encoded outputs of an underlying GNG h with 50 units trained on the MNIST input space. Shown prototypes originate from h and were chosen as in figure 4b. Prototype entries of the units marked with a blue circle and a green square (c) are shown in figure 6. (**d**) For comparison: GNG with 10 units that was trained on the MNIST input space directly.

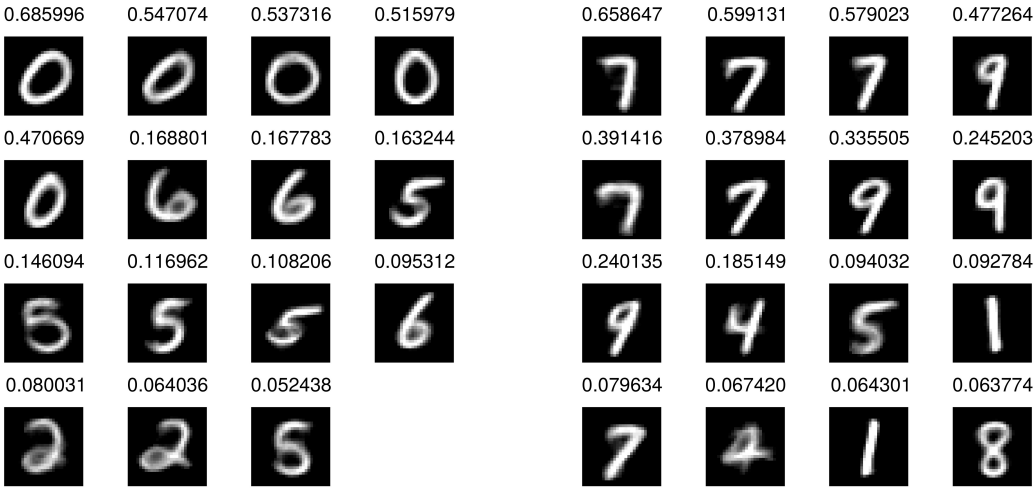


Figure 6: Vector entries with a value above 0.05 of two prototype vectors from the GNG r_{10} shown in figure 5c. For each entry its value and the corresponding prototype of the underlying GNG h are shown. (**a**) Prototype entries of the unit marked with a blue circle in figure 5c. (**b**) Prototype entries of the unit marked with a green square in figure 5c.

left) and the remaining prototypes appears to correspond well to the numerical differences between the respective values. Furthermore, the number of prototype entries of substantial size is small compared to the overall number of entries (50), i.e., although the prototype vectors of GNG r_{10} refer to larger local regions in the underlying MNIST input space they remain sparse.

The ability to learn a single, meaningful sparse representation that covers multiple, similar input regions may prove to be an important property. Common prototype-based models represent their input space by a set of prototypes where each prototype represents a local region of input space. The shape of this local region depends on the used distance measure. In case of the widely used Euclidean distance each local region is a convex polyhedron. As a consequence, complex regions in the input space, e.g., a region that corresponds to a certain class of inputs ("number 0", "flowers", "faces", etc.) have to be approximated piecewise by a group of prototypes. Identifying which prototypes belong to which group in an unsupervised fashion is a nontrivial task. The proposed relaxed one-hot encoding may facilitate this process by enabling the identification of neighboring prototypes through common distance measures. For example, the individual prototypes of the GNG r_{10} (fig. 5c) seem to capture at least some characteristics of the numerical classes, i.e., complex regions, that are present in the MNIST input space. Assuming that the resulting sparse representations of such a secondary model do represent classes or meaningful attributes, the sparseness of the representation may also be utilized to form combined representations, i.e., *feature vectors* that represent multiple attributes at once.

5 Conclusion

We presented a sparse representation for units of prototype-based network models that is based on the additional information provided by local input space histograms. The results of our experiments indicate that this representation is able to retain important neighborhood relations discovered by the prototype-based model while pruning erroneous relations that were introduced to the model due to effects of high dimensionality. The proposed representation offers a new method to encode the units of a prototype-based model in such a way that the relations discovered by the model can be recognized by common distance measures in onward stages of processing, e.g., in a cascade of prototype-based models where the prototypes of subsequent models can learn to represent more complex regions of the original input space.

The presented ideas are not specific to the GNG algorithm. Essentially, they can be adapted to any machine learning algorithm that forms a graph-based representation of its input space. Thus, the proposed sparse output format can be seen as a rather general approach to preserve learned neighborhood relations for any type of further processing that just operates on vector inputs.

Open questions related to the presented methods involve the effects of non standard distance measures on the ratio r and the resulting LISHs, the exploration of alternatives to the LISH average bin error for characterizing associated regions of input space, or the investigation if and how multiple vectors encoded with the proposed sparse representation could be combined to form, e.g., feature vectors.

References

[1] Michael Biehl, Barbara Hammer, and Thomas Villmann. *Distance Measures for Prototype Based Classification*, pages 100–116. Springer International Publishing, Cham, 2014.

[2] Bernd Fritzke. A growing neural gas network learns topologies. In *Advances in Neural Information Processing Systems 7*, pages 625–632. MIT Press, 1995.

[3] Thomas M. J. Fruchterman and Edward M. Reingold. Graph drawing by force-directed placement. *Softw. Pract. Exper.*, 21(11):1129–1164, nov 1991.

[4] Barbara Hammer and T. Villmann. Classification using non standard metrics. In M. Verleysen, editor, *ESANN'05*, pages 303–316. d-side publishing, 2005.

[5] Jochen Kerdels and Gabriele Peters. Supporting gng-based clustering with local input space histograms. In Michael Verleysen, editor, *Proceedings of the 22nd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pages 559–564, Louvain-la-Neuve, Belgique, April 2014.

[6] Jochen Kerdels and Gabriele Peters. Analysis of high-dimensional data using local input space histograms. *Neurocomputing*, 2015.

[7] Teuvo Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43(1):59–69, 1982.

[8] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, Nov 1998.

[9] Thomas M. Martinetz and Klaus Schulten. Topology representing networks. *Neural Networks*, 7:507–522, 1994.

[10] David Nova and Pablo A. Estévez. A review of learning vector quantization classifiers. *Neural Computing and Applications*, 25(3):511–524, 2014.