

Presentation of the Paper “Granular Neural Networks and Their Development Through Context-Based Clustering and Adjustable Deminsionality of Receptive Fields” by Ho-Sung Park, Witold Pedrycz, and Sung-Kwun Oh, and other related papers

Ondřej Sýkora

December 2, 2009

Introduction

The title of the paper... explained

- ▶ Granular Neural Networks and Their Development Through Context-Based Clustering and Adjustable Deminsionality of Receptive Fields
 - ▶ Everything is said already...

Introduction

The title of the paper... explained

- ▶ Granular Neural Networks and Their Development Through Context-Based Clustering and Adjustable Dimensionality of Receptive Fields
 - ▶ Everything is said already...
Granular
Neural Networks
and Their Development
Through Context-Based
Clustering
and Adjustable Dimensionality of Receptive Fields

Introduction

Radial-Basis Function Networks

- ▶ Feedforward neural network, single hidden layer

Introduction

Radial-Basis Function Networks

- ▶ Feedforward neural network, single hidden layer
- ▶ RBF units in the hidden layer, usually the Gaussian basis

function: $\Phi_i(\mathbf{x}) = e^{-\frac{\|\mathbf{x}-\mathbf{v}_i\|^2}{2r_i^2}}$

Introduction

Radial-Basis Function Networks

- ▶ Feedforward neural network, single hidden layer
- ▶ RBF units in the hidden layer, usually the Gaussian basis

function: $\Phi_i(\mathbf{x}) = e^{-\frac{\|\mathbf{x}-\mathbf{v}_i\|^2}{2r_i^2}}$

- ▶ Linear (summative) unit(s) in the output layer:

$$y(\mathbf{x}) = \sum_{i=1}^K w_i \Phi_i(\mathbf{x})$$

Introduction

Radial-Basis Function Networks

- ▶ Feedforward neural network, single hidden layer
- ▶ RBF units in the hidden layer, usually the Gaussian basis

function: $\Phi_i(\mathbf{x}) = e^{-\frac{\|\mathbf{x}-\mathbf{v}_i\|^2}{2r_i^2}}$

- ▶ Linear (summative) unit(s) in the output layer:

$$y(\mathbf{x}) = \sum_{i=1}^K w_i \Phi_i(\mathbf{x})$$

- ▶ “Topologic” interpretation of the hidden layer

Introduction

Radial-Basis Function Networks

- ▶ Feedforward neural network, single hidden layer
- ▶ RBF units in the hidden layer, usually the Gaussian basis

function: $\Phi_i(\mathbf{x}) = e^{-\frac{\|\mathbf{x}-\mathbf{v}_i\|^2}{2r_i^2}}$

- ▶ Linear (summative) unit(s) in the output layer:

$$y(\mathbf{x}) = \sum_{i=1}^K w_i \Phi_i(\mathbf{x})$$

- ▶ “Topologic” interpretation of the hidden layer
 - ▶ Partitioning of the input space via receptive fields of the neurons

Introduction

Learning of Radial-Basis Function Networks

1. Use your favorite gradient method to find the “best” parameters

Introduction

Learning of Radial-Basis Function Networks

1. Use your favorite gradient method to find the “best” parameters
 - ▶ All functions are smooth (in case of Gaussian function)

Introduction

Learning of Radial-Basis Function Networks

1. Use your favorite gradient method to find the “best” parameters
 - ▶ All functions are smooth (in case of Gaussian function)
 - ▶ Back-propagation and variations, Levenberg-Marquardt, ...

Introduction

Learning of Radial-Basis Function Networks

1. Use your favorite gradient method to find the “best” parameters
 - ▶ All functions are smooth (in case of Gaussian function)
 - ▶ Back-propagation and variations, Levenberg-Marquardt, ...
2. Use the “topological” properties

Introduction

Learning of Radial-Basis Function Networks

1. Use your favorite gradient method to find the “best” parameters
 - ▶ All functions are smooth (in case of Gaussian function)
 - ▶ Back-propagation and variations, Levenberg-Marquardt, ...
2. Use the “topological” properties
 - ▶ First, create the receptive fields

Introduction

Learning of Radial-Basis Function Networks

1. Use your favorite gradient method to find the “best” parameters
 - ▶ All functions are smooth (in case of Gaussian function)
 - ▶ Back-propagation and variations, Levenberg-Marquardt, ...
2. Use the “topological” properties
 - ▶ First, create the receptive fields
 - ▶ Assign the values to them later

Introduction

Learning of Radial-Basis Function Networks

1. Use your favorite gradient method to find the “best” parameters
 - ▶ All functions are smooth (in case of Gaussian function)
 - ▶ Back-propagation and variations, Levenberg-Marquardt, ...
2. Use the “topological” properties
 - ▶ First, create the receptive fields
 - ▶ Assign the values to them later
 - ▶ Multiple approaches to these tasks

Introduction

Learning of Radial-Basis Function Networks

1. Use your favorite gradient method to find the “best” parameters
 - ▶ All functions are smooth (in case of Gaussian function)
 - ▶ Back-propagation and variations, Levenberg-Marquardt, ...
2. Use the “topological” properties
 - ▶ First, create the receptive fields
 - ▶ Assign the values to them later
 - ▶ Multiple approaches to these tasks
 - ▶ Assigning values is easy (least squares fitting, ...)

Introduction

Learning of Radial-Basis Function Networks

1. Use your favorite gradient method to find the “best” parameters
 - ▶ All functions are smooth (in case of Gaussian function)
 - ▶ Back-propagation and variations, Levenberg-Marquardt, ...
2. Use the “topological” properties
 - ▶ First, create the receptive fields
 - ▶ Assign the values to them later
 - ▶ Multiple approaches to these tasks
 - ▶ Assigning values is easy (least squares fitting, ...)
 - ▶ Multiple approaches to partitioning the space...

Introduction

Learning of Radial-Basis Function Networks

1. Use your favorite gradient method to find the “best” parameters
 - ▶ All functions are smooth (in case of Gaussian function)
 - ▶ Back-propagation and variations, Levenberg-Marquardt, ...
2. Use the “topological” properties
 - ▶ First, create the receptive fields
 - ▶ Assign the values to them later
 - ▶ Multiple approaches to these tasks
 - ▶ Assigning values is easy (least squares fitting, ...)
 - ▶ Multiple approaches to partitioning the space...
 - ▶ ... usually by clustering the sample data (K-means, ...)

Introduction

Learning of Radial-Basis Function Networks

1. Use your favorite gradient method to find the “best” parameters
 - ▶ All functions are smooth (in case of Gaussian function)
 - ▶ Back-propagation and variations, Levenberg-Marquardt, ...
2. Use the “topological” properties
 - ▶ First, create the receptive fields
 - ▶ Assign the values to them later
 - ▶ Multiple approaches to these tasks
 - ▶ Assigning values is easy (least squares fitting, ...)
 - ▶ Multiple approaches to partitioning the space...
 - ▶ ... usually by clustering the sample data (K-means, ...)
 - ▶ Problem: Taking the output values into account

Granular Neural Networks

Introduction and definitions

- ▶ An extension to the RBF Neural Network model

Granular Neural Networks

Introduction and definitions

- ▶ An extension to the RBF Neural Network model
 - ▶ The same approach to architecture of the network

Granular Neural Networks

Introduction and definitions

- ▶ An extension to the RBF Neural Network model
 - ▶ The same approach to architecture of the network
 - ▶ Emphasis on the partitioning of the input space

Granular Neural Networks

Introduction and definitions

- ▶ An extension to the RBF Neural Network model
 - ▶ The same approach to architecture of the network
 - ▶ Emphasis on the partitioning of the input space
 - ▶ Extensions to the calculation in the output layer

Granular Neural Networks

Introduction and definitions

- ▶ An extension to the RBF Neural Network model
 - ▶ The same approach to architecture of the network
 - ▶ Emphasis on the partitioning of the input space
 - ▶ Extensions to the calculation in the output layer
- ▶ Information Granules

Granular Neural Networks

Introduction and definitions

- ▶ An extension to the RBF Neural Network model
 - ▶ The same approach to architecture of the network
 - ▶ Emphasis on the partitioning of the input space
 - ▶ Extensions to the calculation in the output layer
- ▶ Information Granules
 - ▶ “Clusters” in the input space formed by the receptive fields of the neurons in the hidden layer

Granular Neural Networks

Introduction and definitions

- ▶ An extension to the RBF Neural Network model
 - ▶ The same approach to architecture of the network
 - ▶ Emphasis on the partitioning of the input space
 - ▶ Extensions to the calculation in the output layer
- ▶ Information Granules
 - ▶ “Clusters” in the input space formed by the receptive fields of the neurons in the hidden layer
 - ▶ A single granule may be formed by multiple receptive fields

Granular Neural Networks

Introduction and definitions

- ▶ “Contexts” formed by the distribution of the data in the output space

Granular Neural Networks

Introduction and definitions

- ▶ “Contexts” formed by the distribution of the data in the output space
- ▶ “Functional” weights to the output layer layer

Granular Neural Networks

Introduction and definitions

- ▶ “Contexts” formed by the distribution of the data in the output space
- ▶ “Functional” weights to the output layer layer
 - ▶ Each unit in the hidden layer is assigned a linear function

Granular Neural Networks

Introduction and definitions

- ▶ “Contexts” formed by the distribution of the data in the output space
- ▶ “Functional” weights to the output layer layer
 - ▶ Each unit in the hidden layer is assigned a linear function
 - ▶ For unit i : $\hat{y}_i(\mathbf{x}) = y_p + \mathbf{a}_i^T(\mathbf{x} - \mathbf{v}_i)$
 - ▶ p denotes the context, to which the unit belongs

Granular Neural Networks

Introduction and definitions

- ▶ “Contexts” formed by the distribution of the data in the output space
- ▶ “Functional” weights to the output layer layer
 - ▶ Each unit in the hidden layer is assigned a linear function
 - ▶ For unit i : $\hat{y}_i(\mathbf{x}) = y_p + \mathbf{a}_i^T(\mathbf{x} - \mathbf{v}_i)$
 - ▶ p denotes the context, to which the unit belongs
- ▶ The output of the network is a weighted sum of the linear functions:

Granular Neural Networks

Introduction and definitions

- ▶ “Contexts” formed by the distribution of the data in the output space
- ▶ “Functional” weights to the output layer layer
 - ▶ Each unit in the hidden layer is assigned a linear function
 - ▶ For unit i : $\hat{y}_i(\mathbf{x}) = y_p + \mathbf{a}_i^T(\mathbf{x} - \mathbf{v}_i)$
 - ▶ p denotes the context, to which the unit belongs
- ▶ The output of the network is a weighted sum of the linear functions:

- ▶ $\hat{y}(\mathbf{x}) = \sum_i u_i(\mathbf{x})\hat{y}_i(\mathbf{x})$, where $u_i = \frac{1}{\sum_{j=1}^c (\|\mathbf{x} - \mathbf{v}_i\| / \|\mathbf{x} - \mathbf{v}_j\|)}$

Granular Neural Networks

The Data-Driven Development Method

1. Determine structure in the output space

Granular Neural Networks

The Data-Driven Development Method

1. Determine structure in the output space
 - ▶ *K*-means clustering of the output values

Granular Neural Networks

The Data-Driven Development Method

1. Determine structure in the output space
 - ▶ *K*-means clustering of the output values
2. Create “contexts” based on the prototypes from the previous step

Granular Neural Networks

The Data-Driven Development Method

1. Determine structure in the output space
 - ▶ *K*-means clustering of the output values
2. Create “contexts” based on the prototypes from the previous step
 - ▶ Each context is a fuzzy set in the output space

Granular Neural Networks

The Data-Driven Development Method

1. Determine structure in the output space
 - ▶ *K*-means clustering of the output values
2. Create “contexts” based on the prototypes from the previous step
 - ▶ Each context is a fuzzy set in the output space
3. For each context, determine the structure of the input space that supports it

Granular Neural Networks

The Data-Driven Development Method

1. Determine structure in the output space
 - ▶ *K*-means clustering of the output values
2. Create “contexts” based on the prototypes from the previous step
 - ▶ Each context is a fuzzy set in the output space
3. For each context, determine the structure of the input space that supports it
 - ▶ Context-Based Fuzzy C-Means Clustering of the values in the input space corresponding to the context

Granular Neural Networks

The Data-Driven Development Method

1. Determine structure in the output space
 - ▶ *K*-means clustering of the output values
2. Create “contexts” based on the prototypes from the previous step
 - ▶ Each context is a fuzzy set in the output space
3. For each context, determine the structure of the input space that supports it
 - ▶ Context-Based Fuzzy C-Means Clustering of the values in the input space corresponding to the context
4. Adjust the “free” parameters of the network

Granular Neural Networks

The Data-Driven Development Method

1. Determine structure in the output space
 - ▶ *K*-means clustering of the output values
2. Create “contexts” based on the prototypes from the previous step
 - ▶ Each context is a fuzzy set in the output space
3. For each context, determine the structure of the input space that supports it
 - ▶ Context-Based Fuzzy C-Means Clustering of the values in the input space corresponding to the context
4. Adjust the “free” parameters of the network
 - ▶ Coefficients of the linear functions

Granular Neural Networks

The Data-Driven Development Method

1. Determine structure in the output space
 - ▶ *K*-means clustering of the output values
2. Create “contexts” based on the prototypes from the previous step
 - ▶ Each context is a fuzzy set in the output space
3. For each context, determine the structure of the input space that supports it
 - ▶ Context-Based Fuzzy C-Means Clustering of the values in the input space corresponding to the context
4. Adjust the “free” parameters of the network
 - ▶ Coefficients of the linear functions
 - ▶ Least Squares fitting

Data-Driven Development Method

Preliminaries

Data-Driven Development Method

Preliminaries

- ▶ N training samples in the form $(\mathbf{x}_i, y_i), i = 1, \dots, N$

Data-Driven Development Method

Preliminaries

- ▶ N training samples in the form $(\mathbf{x}_i, y_i), i = 1, \dots, N$
- ▶ Input: n -dimensional vectors \mathbf{x}_i

Data-Driven Development Method

Preliminaries

- ▶ N training samples in the form $(\mathbf{x}_i, y_i), i = 1, \dots, N$
- ▶ Input: n -dimensional vectors \mathbf{x}_i
- ▶ Output: scalar (real) values y_i

Data-Driven Development Method

Structure of the output space

Data-Driven Development Method

Structure of the output space

- ▶ K -means clustering of the output values into $P - 2$ clusters

Data-Driven Development Method

Structure of the output space

- ▶ K -means clustering of the output values into $P - 2$ clusters
 - ▶ Yields prototypes $y_2 \leq y_3 \leq \dots \leq y_{P-1}$

Data-Driven Development Method

Structure of the output space

- ▶ K -means clustering of the output values into $P - 2$ clusters
 - ▶ Yields prototypes $y_2 \leq y_3 \leq \dots \leq y_{P-1}$
- ▶ Create prototypes y_1 and y_P for the minimal and maximal output values.

Data-Driven Development Method

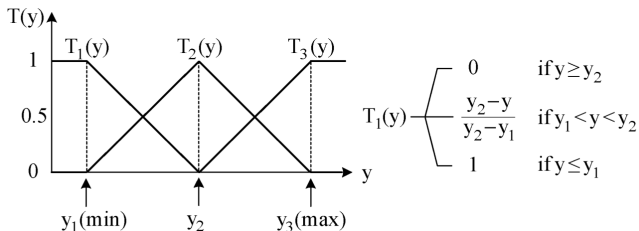
Structure of the output space

- ▶ K -means clustering of the output values into $P - 2$ clusters
 - ▶ Yields prototypes $y_2 \leq y_3 \leq \dots \leq y_{P-1}$
- ▶ Create prototypes y_1 and y_P for the minimal and maximal output values.
- ▶ For each prototype y_i define a context as a fuzzy set with a triangular membership function $T_i(y)$:

Data-Driven Development Method

Structure of the output space

- ▶ K -means clustering of the output values into $P - 2$ clusters
 - ▶ Yields prototypes $y_2 \leq y_3 \leq \dots \leq y_{P-1}$
- ▶ Create prototypes y_1 and y_P for the minimal and maximal output values.
- ▶ For each prototype y_i define a context as a fuzzy set with a triangular membership function $T_i(y)$:



Data-Driven Development Method

Structure of the input space

- ▶ Context-Based Fuzzy C-Means Clustering

Data-Driven Development Method

Structure of the input space

- ▶ Context-Based Fuzzy C-Means Clustering
 - ▶ Takes into account the membership of the samples to the “context”

Data-Driven Development Method

Structure of the input space

- ▶ Context-Based Fuzzy C-Means Clustering
 - ▶ Takes into account the membership of the samples to the “context”
 - ▶ For each context j create partition matrix $\mathbf{U}(T_j) = (u_{ik})_{c \times N}$, such that:

$$u_{ik} \in [0, 1], \sum_{i=1}^c u_{ik} = T_j(\mathbf{x}_k) \forall k, 0 < \sum_{k=1}^N u_{ik} < N \forall i,$$

that minimizes $V = \sum_{i=1}^c \sum_{k=1}^N u_{ik}^m \|\mathbf{x}_k - \mathbf{z}_i\|^2$

Data-Driven Development Method

Structure of the input space

- ▶ Context-Based Fuzzy C-Means Clustering
 - ▶ Takes into account the membership of the samples to the “context”
 - ▶ For each context j create partition matrix $\mathbf{U}(T_j) = (u_{ik})_{c \times N}$, such that:

$$u_{ik} \in [0, 1], \sum_{i=1}^c u_{ik} = T_j(\mathbf{x}_k) \forall k, 0 < \sum_{k=1}^N u_{ik} < N \forall i,$$

that minimizes $V = \sum_{i=1}^c \sum_{k=1}^N u_{ik}^m \|\mathbf{x}_k - \mathbf{z}_i\|^2$

- ▶ c is the number of units in the context

Data-Driven Development Method

Structure of the input space

- ▶ Context-Based Fuzzy C-Means Clustering
 - ▶ Takes into account the membership of the samples to the “context”
 - ▶ For each context j create partition matrix $\mathbf{U}(T_j) = (u_{ik})_{c \times N}$, such that:

$$u_{ik} \in [0, 1], \sum_{i=1}^c u_{ik} = T_j(\mathbf{x}_k) \forall k, 0 < \sum_{k=1}^N u_{ik} < N \forall i,$$

that minimizes $V = \sum_{i=1}^c \sum_{k=1}^N u_{ik}^m \|\mathbf{x}_k - \mathbf{z}_i\|^2$

- ▶ c is the number of units in the context
- ▶ \mathbf{z}_i are the prototypes of the clusters

Data-Driven Development Method

Structure of the input space

- ▶ Context-Based Fuzzy C-Means Clustering
 - ▶ Takes into account the membership of the samples to the “context”
 - ▶ For each context j create partition matrix $\mathbf{U}(T_j) = (u_{ik})_{c \times N}$, such that:

$$u_{ik} \in [0, 1], \sum_{i=1}^c u_{ik} = T_j(\mathbf{x}_k) \forall k, 0 < \sum_{k=1}^N u_{ik} < N \forall i,$$

that minimizes $V = \sum_{i=1}^c \sum_{k=1}^N u_{ik}^m \|\mathbf{x}_k - \mathbf{z}_i\|^2$

- ▶ c is the number of units in the context
- ▶ \mathbf{z}_i are the prototypes of the clusters
- ▶ $m > 1$ is the fuzzification coefficient

Data-Driven Development Method

Structure of the input space

- ▶ Context-Based Fuzzy C-Means Clustering

Data-Driven Development Method

Structure of the input space

- ▶ Context-Based Fuzzy C-Means Clustering
 - ▶ Iterative algorithm, similar to Fuzzy C-Means

Data-Driven Development Method

Structure of the input space

- ▶ Context-Based Fuzzy C-Means Clustering
 - ▶ Iterative algorithm, similar to Fuzzy C-Means
 - ▶ Repeat the following steps until convergence:

Data-Driven Development Method

Structure of the input space

- ▶ Context-Based Fuzzy C-Means Clustering
 - ▶ Iterative algorithm, similar to Fuzzy C-Means
 - ▶ Repeat the following steps until convergence:
 1. Update the partition matrix:

$$u_{ik} = \frac{T_j(\mathbf{x}_k)}{\sum_{l=1}^c \left(\frac{\|\mathbf{x}_k - \mathbf{z}_l\|}{\|\mathbf{x}_k - \mathbf{z}_j\|} \right)^{2/(m-1)}}, i = 1, 2, \dots, c, k = 1, 2, \dots, N$$

Data-Driven Development Method

Structure of the input space

- ▶ Context-Based Fuzzy C-Means Clustering
 - ▶ Iterative algorithm, similar to Fuzzy C-Means
 - ▶ Repeat the following steps until convergence:
 1. Update the partition matrix:

$$u_{ik} = \frac{T_j(\mathbf{x}_k)}{\sum_{l=1}^c \left(\frac{\|\mathbf{x}_k - \mathbf{z}_l\|}{\|\mathbf{x}_k - \mathbf{z}_i\|} \right)^{2/(m-1)}}, i = 1, 2, \dots, c, k = 1, 2, \dots, N$$

2. Use the new partition matrix to update the prototypes

$$\mathbf{z}_i = \frac{\sum_{k=1}^N u_{ik}^m \mathbf{x}_k}{\sum_{k=1}^N u_{ik}^m}, i = 1, 2, \dots, c$$

Data-Driven Development Method

Structure of the input space

- ▶ Context-Based Fuzzy C-Means Clustering
 - ▶ Iterative algorithm, similar to Fuzzy C-Means
 - ▶ Repeat the following steps until convergence:
 1. Update the partition matrix:

$$u_{ik} = \frac{T_j(\mathbf{x}_k)}{\sum_{l=1}^c \left(\frac{\|\mathbf{x}_k - \mathbf{z}_l\|}{\|\mathbf{x}_k - \mathbf{z}_i\|} \right)^{2/(m-1)}}, i = 1, 2, \dots, c, k = 1, 2, \dots, N$$

2. Use the new partition matrix to update the prototypes

$$\mathbf{z}_i = \frac{\sum_{k=1}^N u_{ik}^m \mathbf{x}_k}{\sum_{k=1}^N u_{ik}^m}, i = 1, 2, \dots, c$$

- ▶ Converges to a (locally) optimal state with respect to V

Data-Driven Development Method

Structure of the input space

- ▶ The optimal number of clusters in a context

Data-Driven Development Method

Structure of the input space

- ▶ The optimal number of clusters in a context
 - ▶ Start with a fixed number of clusters

Data-Driven Development Method

Structure of the input space

- ▶ The optimal number of clusters in a context
 - ▶ Start with a fixed number of clusters
 - ▶ “Reconstructibility criterion” – used by the authors of the paper

Data-Driven Development Method

Structure of the input space

- ▶ The optimal number of clusters in a context
 - ▶ Start with a fixed number of clusters
 - ▶ “Reconstructibility criterion” – used by the authors of the paper
 - ▶ For a context j define L_j as the set of clusters in the context

Data-Driven Development Method

Structure of the input space

- ▶ The optimal number of clusters in a context
 - ▶ Start with a fixed number of clusters
 - ▶ “Reconstructibility criterion” – used by the authors of the paper
 - ▶ For a context j define L_j as the set of clusters in the context
 - ▶ Define $\Omega_j = \{\mathbf{x}_k : \arg \max_l u_{lk} \in L_j\}$ (u_{lk} here contains the membership for all clusters in all contexts!)

Data-Driven Development Method

Structure of the input space

- ▶ The optimal number of clusters in a context
 - ▶ Start with a fixed number of clusters
 - ▶ “Reconstructibility criterion” – used by the authors of the paper
 - ▶ For a context j define L_j as the set of clusters in the context
 - ▶ Define $\Omega_j = \{\mathbf{x}_k : \arg \max_l u_{lk} \in L_j\}$ (u_{lk} here contains the membership for all clusters in all contexts!)
 - ▶ Define $\hat{\mathbf{x}}_k = \frac{\sum_{i \in L_j} g_{ik}^m \mathbf{z}_i}{\sum_{i \in L_j} g_{ik}^m}$, where

$$g_{ik} = \frac{1}{\sum_{i \in L_j} (\|\mathbf{x}_k - \mathbf{v}_i\| / \|\mathbf{x}_k - \mathbf{v}_l\|)^{2/(m-1)}}$$

Data-Driven Development Method

Structure of the input space

- ▶ The optimal number of clusters in a context
 - ▶ Start with a fixed number of clusters
 - ▶ “Reconstructibility criterion” – used by the authors of the paper
 - ▶ For a context j define L_j as the set of clusters in the context
 - ▶ Define $\Omega_j = \{\mathbf{x}_k : \arg \max_l u_{lk} \in L_j\}$ (u_{lk} here contains the membership for all clusters in all contexts!)
 - ▶ Define $\hat{\mathbf{x}}_k = \frac{\sum_{i \in L_j} g_{ik}^m z_i}{\sum_{i \in L_j} g_{ik}^m}$, where

$$g_{ik} = \frac{1}{\sum_{i \in L_j} (\|\mathbf{x}_k - \mathbf{v}_i\| / \|\mathbf{x}_k - \mathbf{v}_l\|)^{2/(m-1)}}$$

- ▶ Define $V(j) = \sum_{k \in \Omega_j} \|\mathbf{x}_k - \hat{\mathbf{x}}_k\|^2$

Data-Driven Development Method

Structure of the input space

- ▶ The optimal number of clusters in a context
 - ▶ Start with a fixed number of clusters
 - ▶ “Reconstructibility criterion” – used by the authors of the paper
 - ▶ For a context j define L_j as the set of clusters in the context
 - ▶ Define $\Omega_j = \{\mathbf{x}_k : \arg \max_l u_{lk} \in L_j\}$ (u_{lk} here contains the membership for all clusters in all contexts!)
 - ▶ Define $\hat{\mathbf{x}}_k = \frac{\sum_{i \in L_j} g_{ik}^m z_i}{\sum_{i \in L_j} g_{ik}^m}$, where

$$g_{ik} = \frac{1}{\sum_{i \in L_j} (\|\mathbf{x}_k - \mathbf{v}_i\| / \|\mathbf{x}_k - \mathbf{v}_l\|)^{2/(m-1)}}$$

- ▶ Define $V(j) = \sum_{k \in \Omega_j} \|\mathbf{x}_k - \hat{\mathbf{x}}_k\|^2$
 - ▶ Increase the number of clusters as needed to reduce $V(j)$

Data-Driven Development Method

Structure of the input space

- ▶ The optimal number of clusters in a context
 - ▶ Start with a fixed number of clusters
 - ▶ “Reconstructibility criterion” – used by the authors of the paper
 - ▶ For a context j define L_j as the set of clusters in the context
 - ▶ Define $\Omega_j = \{\mathbf{x}_k : \arg \max_l u_{lk} \in L_j\}$ (u_{lk} here contains the membership for all clusters in all contexts!)
 - ▶ Define $\hat{\mathbf{x}}_k = \frac{\sum_{i \in L_j} g_{ik}^m z_i}{\sum_{i \in L_j} g_{ik}^m}$, where

$$g_{ik} = \frac{1}{\sum_{i \in L_j} (\|\mathbf{x}_k - \mathbf{v}_i\| / \|\mathbf{x}_k - \mathbf{v}_l\|)^{2/(m-1)}}$$

- ▶ Define $V(j) = \sum_{k \in \Omega_j} \|\mathbf{x}_k - \hat{\mathbf{x}}_k\|^2$
- ▶ Increase the number of clusters as needed to reduce $V(j)$
 - ▶ Take the least $\min_j V(j)$ as a reference, try to reach it with other contexts'

Data-Driven Development Method

Adjust the “free” parameters

Data-Driven Development Method

Adjust the “free” parameters

- ▶ Consider the output of the network:

$$\hat{y}(\mathbf{x}) = \sum_i u_i(\mathbf{x}) \hat{y}_i(\mathbf{x}) = \sum_i u_i(\mathbf{x}) [y_{p_i} + \mathbf{a}_i^T (\mathbf{x} - \mathbf{v}_i)]$$

Data-Driven Development Method

Adjust the “free” parameters

- ▶ Consider the output of the network:

$$\hat{y}(\mathbf{x}) = \sum_i u_i(\mathbf{x}) \hat{y}_i(\mathbf{x}) = \sum_i u_i(\mathbf{x}) [y_{p_i} + \mathbf{a}_i^T (\mathbf{x} - \mathbf{v}_i)]$$

- ▶ The only free parameters is \mathbf{a}_i , all other values were determined by the clustering algorithms

Data-Driven Development Method

Adjust the “free” parameters

- ▶ Consider the output of the network:

$$\hat{y}(\mathbf{x}) = \sum_i u_i(\mathbf{x}) \hat{y}_i(\mathbf{x}) = \sum_i u_i(\mathbf{x}) [y_{p_i} + \mathbf{a}_i^T (\mathbf{x} - \mathbf{v}_i)]$$

- ▶ The only free parameters is \mathbf{a}_i , all other values were determined by the clustering algorithms
- ▶ Note, that \hat{y} is linear in \mathbf{a}_i for all i

Data-Driven Development Method

Adjust the “free” parameters

- ▶ Consider the output of the network:

$$\hat{y}(\mathbf{x}) = \sum_i u_i(\mathbf{x}) \hat{y}_i(\mathbf{x}) = \sum_i u_i(\mathbf{x}) [y_{p_i} + \mathbf{a}_i^T (\mathbf{x} - \mathbf{v}_i)]$$

- ▶ The only free parameters is \mathbf{a}_i , all other values were determined by the clustering algorithms
- ▶ Note, that \hat{y} is linear in \mathbf{a}_i for all i
 - ▶ Least squares method can be used to minimize $\sum_{k=1}^N (\hat{y}(\mathbf{x}_k) - y_k)^2$, with respect to \mathbf{a}_i

Data-Driven Development Method

Summary

Data-Driven Development Method

Summary

- ▶ *K*-means clustering of the output space

Data-Driven Development Method

Summary

- ▶ *K*-means clustering of the output space
 - ▶ Leads to partitioning the training samples into the contexts

Data-Driven Development Method

Summary

- ▶ K -means clustering of the output space
 - ▶ Leads to partitioning the training samples into the contexts
- ▶ Context-Based Fuzzy C-Means Clustering of the input space with respect to the contexts

Data-Driven Development Method

Summary

- ▶ *K*-means clustering of the output space
 - ▶ Leads to partitioning the training samples into the contexts
- ▶ Context-Based Fuzzy C-Means Clustering of the input space with respect to the contexts
 - ▶ Gives the clusters which form the hidden layer of the network

Data-Driven Development Method

Summary

- ▶ *K*-means clustering of the output space
 - ▶ Leads to partitioning the training samples into the contexts
- ▶ Context-Based Fuzzy C-Means Clustering of the input space with respect to the contexts
 - ▶ Gives the clusters which form the hidden layer of the network
- ▶ Least Squares fitting to find a linear function for each cluster, that best fits the training data

Reduction of dimensionality

Introduction

- ▶ Motivation:

Reduction of dimensionality

Introduction

- ▶ Motivation:
 - ▶ Eliminate (highly) irrelevant input variables, simplify the network, improve generalization, . . .

Reduction of dimensionality

Introduction

- ▶ Motivation:
 - ▶ Eliminate (highly) irrelevant input variables, simplify the network, improve generalization, . . .
- ▶ Method:

Reduction of dimensionality

Introduction

- ▶ Motivation:
 - ▶ Eliminate (highly) irrelevant input variables, simplify the network, improve generalization, . . .
- ▶ Method:
 - ▶ Select the optimal subset of the input variables for each context

Reduction of dimensionality

Introduction

- ▶ Motivation:
 - ▶ Eliminate (highly) irrelevant input variables, simplify the network, improve generalization, . . .
- ▶ Method:
 - ▶ Select the optimal subset of the input variables for each context
 - ▶ Use Genetic Algorithms to find the optimal subset

Reduction of dimensionality

Quality of reduction

Reduction of dimensionality

Quality of reduction

- ▶ Idea: keep such variables, that the structure discovered in the original input space is retained in the reduced input space

Reduction of dimensionality

Quality of reduction

- ▶ Idea: keep such variables, that the structure discovered in the original input space is retained in the reduced input space
 - ▶ The structure is the same, if the partitioning of the training samples is the same

Reduction of dimensionality

Quality of reduction

- ▶ Idea: keep such variables, that the structure discovered in the original input space is retained in the reduced input space
 - ▶ The structure is the same, if the partitioning of the training samples is the same
 - ▶ Partitioning matrices cannot be compared directly!

Reduction of dimensionality

Quality of reduction

- ▶ Idea: keep such variables, that the structure discovered in the original input space is retained in the reduced input space
 - ▶ The structure is the same, if the partitioning of the training samples is the same
 - ▶ Partitioning matrices cannot be compared directly!
- ▶ Idea: Use the proximity between samples to compare the structure

Reduction of dimensionality

Quality of reduction

- ▶ Idea: keep such variables, that the structure discovered in the original input space is retained in the reduced input space
 - ▶ The structure is the same, if the partitioning of the training samples is the same
 - ▶ Partitioning matrices cannot be compared directly!
- ▶ Idea: Use the proximity between samples to compare the structure
 - ▶ For two samples \mathbf{x}_{k_1} , and \mathbf{x}_{k_2} , define
$$Prox(k_1, k_2) = \sum_{i=1}^C \min(u_{ik_1}, u_{ik_2})$$

Reduction of dimensionality

Quality of reduction

- ▶ Idea: keep such variables, that the structure discovered in the original input space is retained in the reduced input space
 - ▶ The structure is the same, if the partitioning of the training samples is the same
 - ▶ Partitioning matrices cannot be compared directly!
- ▶ Idea: Use the proximity between samples to compare the structure
 - ▶ For two samples \mathbf{x}_{k_1} , and \mathbf{x}_{k_2} , define
$$Prox(k_1, k_2) = \sum_{i=1}^C \min(u_{ik_1}, u_{ik_2})$$
 - ▶ For a partitioning matrix U_p , define
$$Prox(U_p) = (Prox(i, j))_{N \times N}$$

Reduction of dimensionality

Quality of reduction

- ▶ Consider a selection of input variables I

Reduction of dimensionality

Quality of reduction

- ▶ Consider a selection of input variables I
 - ▶ The proximity can be defined in the reduced input space as well

Reduction of dimensionality

Quality of reduction

- ▶ Consider a selection of input variables l
 - ▶ The proximity can be defined in the reduced input space as well
 - ▶ Take $u_{ik}^l = \frac{T_p(\mathbf{x}_k)}{\sum_{j=1}^c \left(\frac{\|\mathbf{x}_k^l - \mathbf{v}_j^l\|}{\|\mathbf{x}_k^l - \mathbf{v}_j^l\|} \right)^{2/(m-1)}}$, where \mathbf{x}_k^l are the input samples

restricted to the reduced input space, and \mathbf{v}_j^l are the prototypes of the clusters found for the reduced input space

Reduction of dimensionality

Quality of reduction

- ▶ Consider a selection of input variables l
 - ▶ The proximity can be defined in the reduced input space as well
 - ▶ Take $u_{ik}^l = \frac{T_p(\mathbf{x}_k)}{\sum_{j=1}^c \left(\frac{\|\mathbf{x}_k^l - \mathbf{v}_j^l\|}{\|\mathbf{x}_k^l - \mathbf{v}_j^l\|} \right)^{2/(m-1)}}$, where \mathbf{x}_i^l are the input samples restricted to the reduced input space, and \mathbf{v}_j^l are the prototypes of the clusters found for the reduced input space
 - ▶ Let $U_p^l = (u_{ik}^l)_{N \times N}$ be the partitioning matrix in the reduced input space.

Reduction of dimensionality

Quality of reduction

- ▶ Consider a selection of input variables I
 - ▶ The proximity can be defined in the reduced input space as well
 - ▶ Take $u_{ik}^I = \frac{T_p(\mathbf{x}_k)}{\sum_{j=1}^c \left(\frac{\|\mathbf{x}_k^I - \mathbf{v}_j^I\|}{\|\mathbf{x}_k^I - \mathbf{v}_j^I\|} \right)^{2/(m-1)}}$, where \mathbf{x}_k^I are the input samples

restricted to the reduced input space, and \mathbf{v}_j^I are the prototypes of the clusters found for the reduced input space

- ▶ Let $U_p^I = (u_{ik}^I)_{N \times N}$ be the partitioning matrix in the reduced input space.
- ▶ Define $V_p^I = \| \text{Prox}(U_p) - \text{Prox}(U_p^I) \| = \sum_{k_1=1}^N \sum_{k_2 > k_1}^N | \text{Prox}^{U_p}(k_1, k_2) - \text{Prox}^{U_p^I}(k_1, k_2) |$

Reduction of dimensionality

Quality of reduction

- ▶ Consider a selection of input variables I
 - ▶ The proximity can be defined in the reduced input space as well
 - ▶ Take $u_{ik}^I = \frac{T_p(\mathbf{x}_k)}{\sum_{j=1}^c \left(\frac{\|\mathbf{x}_k^I - \mathbf{v}_j^I\|}{\|\mathbf{x}_k^I - \mathbf{v}_j^I\|} \right)^{2/(m-1)}}$, where \mathbf{x}_k^I are the input samples restricted to the reduced input space, and \mathbf{v}_j^I are the prototypes of the clusters found for the reduced input space
 - ▶ Let $U_p^I = (u_{ik}^I)_{N \times N}$ be the partitioning matrix in the reduced input space.
 - ▶ Define $V_p^I = \| \text{Prox}(U_p) - \text{Prox}(U_p^I) \| = \sum_{k_1=1}^N \sum_{k_2 > k_1}^N | \text{Prox}^{U_p}(k_1, k_2) - \text{Prox}^{U_p^I}(k_1, k_2) |$
- ▶ Goal: find selection of input variables I , that minimizes V_p^I

Reduction of Dimensionality

Embedding into the development method

Reduction of Dimensionality

Embedding into the development method

- ▶ Idea: use a Genetic Algorithm to find the best selection of input variables

Reduction of Dimensionality

Embedding into the development method

- ▶ Idea: use a Genetic Algorithm to find the best selection of input variables
 - ▶ Make the GA a part of the process of the development of each context

Reduction of Dimensionality

Embedding into the development method

- ▶ Idea: use a Genetic Algorithm to find the best selection of input variables
 - ▶ Make the GA a part of the process of the development of each context
 - ▶ The number of selected input variables is chosen by the user

Reduction of Dimensionality

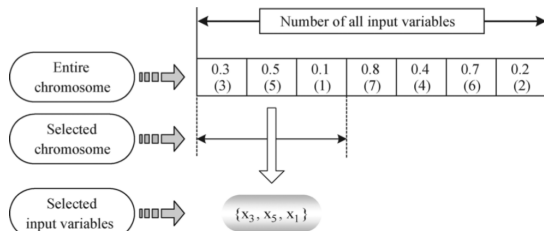
Embedding into the development method

- ▶ Idea: use a Genetic Algorithm to find the best selection of input variables
 - ▶ Make the GA a part of the process of the development of each context
 - ▶ The number of selected input variables is chosen by the user
- ▶ Encoding of the variables into a chromosome

Reduction of Dimensionality

Embedding into the development method

- ▶ Idea: use a Genetic Algorithm to find the best selection of input variables
 - ▶ Make the GA a part of the process of the development of each context
 - ▶ The number of selected input variables is chosen by the user
- ▶ Encoding of the variables into a chromosome



Reduction of Dimensionality

The Genetic Algorithm

Reduction of Dimensionality

The Genetic Algorithm

- ▶ Run separately for each context

Reduction of Dimensionality

The Genetic Algorithm

- ▶ Run separately for each context
- ▶ The parameters of the GA (population size, generations, ...) are determined by the user

Reduction of Dimensionality

The Genetic Algorithm

- ▶ Run separately for each context
- ▶ The parameters of the GA (population size, generations, ...) are determined by the user
- ▶ Fitness defined using the difference in proximity

Reduction of Dimensionality

The Genetic Algorithm

- ▶ Run separately for each context
- ▶ The parameters of the GA (population size, generations, ...) are determined by the user
- ▶ Fitness defined using the difference in proximity
 - ▶ Calculate the clustering for each selection of the input variables I

Reduction of Dimensionality

The Genetic Algorithm

- ▶ Run separately for each context
- ▶ The parameters of the GA (population size, generations, ...) are determined by the user
- ▶ Fitness defined using the difference in proximity
 - ▶ Calculate the clustering for each selection of the input variables l
 - ▶ Define fitness function as $F(l) = \frac{1}{V_l^p}$

Reduction of Dimensionality

The Genetic Algorithm

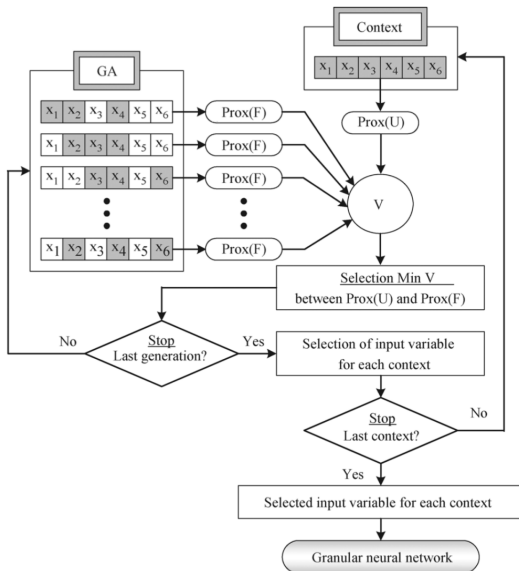
- ▶ Run separately for each context
- ▶ The parameters of the GA (population size, generations, ...) are determined by the user
- ▶ Fitness defined using the difference in proximity
 - ▶ Calculate the clustering for each selection of the input variables I
 - ▶ Define fitness function as $F(I) = \frac{1}{V_p^I}$
- ▶ Roulette-wheel selection, elitism

Reduction of Dimensionality

The complete development process

Reduction of Dimensionality

The complete development process



Reduction of Dimensionality

Summary

Reduction of Dimensionality

Summary

- ▶ The Data-Driven Development Method is extended with input variable selection

Reduction of Dimensionality

Summary

- ▶ The Data-Driven Development Method is extended with input variable selection
- ▶ Different selections for different contexts

Reduction of Dimensionality

Summary

- ▶ The Data-Driven Development Method is extended with input variable selection
- ▶ Different selections for different contexts
- ▶ Optimal selection found by a Genetic Algorithm

Experimental Results

- ▶ Compared with a “standard RBF NN”

Experimental Results

- ▶ Compared with a “standard RBF NN”
- ▶ Compared on synthetic data, as well as data from ML repositories

Experimental Results

- ▶ Compared with a “standard RBF NN”
- ▶ Compared on synthetic data, as well as data from ML repositories
- ▶ RMSE of the proposed network significantly lower than those of the “standard” RBF NNs

Summary

Summary

- ▶ A new type of neural networks derived from the RBF NN

Summary

- ▶ A new type of neural networks derived from the RBF NN
- ▶ Development method based on clustering in the output and input space

Summary

- ▶ A new type of neural networks derived from the RBF NN
- ▶ Development method based on clustering in the output and input space
 - ▶ Context-Based Clustering to guide the creation of receptive fields

Summary

- ▶ A new type of neural networks derived from the RBF NN
- ▶ Development method based on clustering in the output and input space
 - ▶ Context-Based Clustering to guide the creation of receptive fields
- ▶ Reduction of dimensionality of receptive fields

References

- ▶ Ho-Sung Park, Witold Pedrycz, Sung-Kwun Oh. Granular Neural Networks and Their Development Through Context-Based Clustering and Adjustable Dimensionality of Receptive Fields. To appear in IEEE Transactions on Neural Networks. Vol. 9, Issue 6. 2009
- ▶ Ho-Sung Park, Witold Pedrycz, Sung-Kwun Oh. A granular-oriented development of functional radial basis function neural networks. In Neurocomputing. Vol. 72. 2008
- ▶ Witold Pedrycz. Conditional Fuzzy C-Means. In Pattern Recognition Letters. Vol. 17. 1996
- ▶ Witold Pedrycz. Conditional Fuzzy Clustering in the Design of Radial Basis Function Neural Networks. In IEEE Transactions on Neural Networks. Vol. 9. Issue 4. 1998.

▶ Questions, comments?