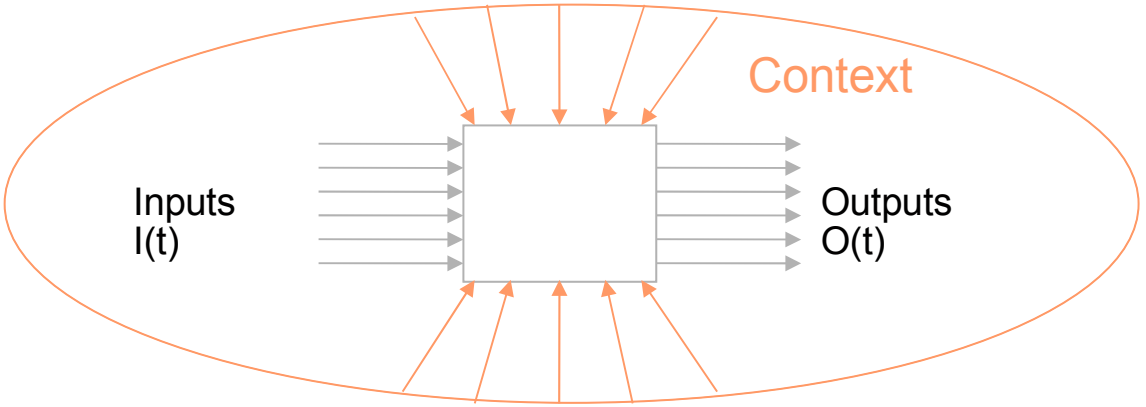
Extending the Growing Neural Gas Classifier for Context Recognition

Eurocast 2007, Workshop on Heuristic Problem Solving, Paper #9.24
14. February 2007, 10:30

Rene Mayrhofer
Lancaster University, UK
Johannes Kepler University Linz, AT

Motivation: Context and Context Awareness

- What is Pervasive Computing?
  - The vision to integrate computer systems with everyday objects
  - Implicit, unobtrusive user interfaces instead of explicit, importunate ones
- What is Context?
  - In the area of pervasive computing: the environment/situation in which something happens
  - Context is everything except the explicit in- and outputs [LS 2000].

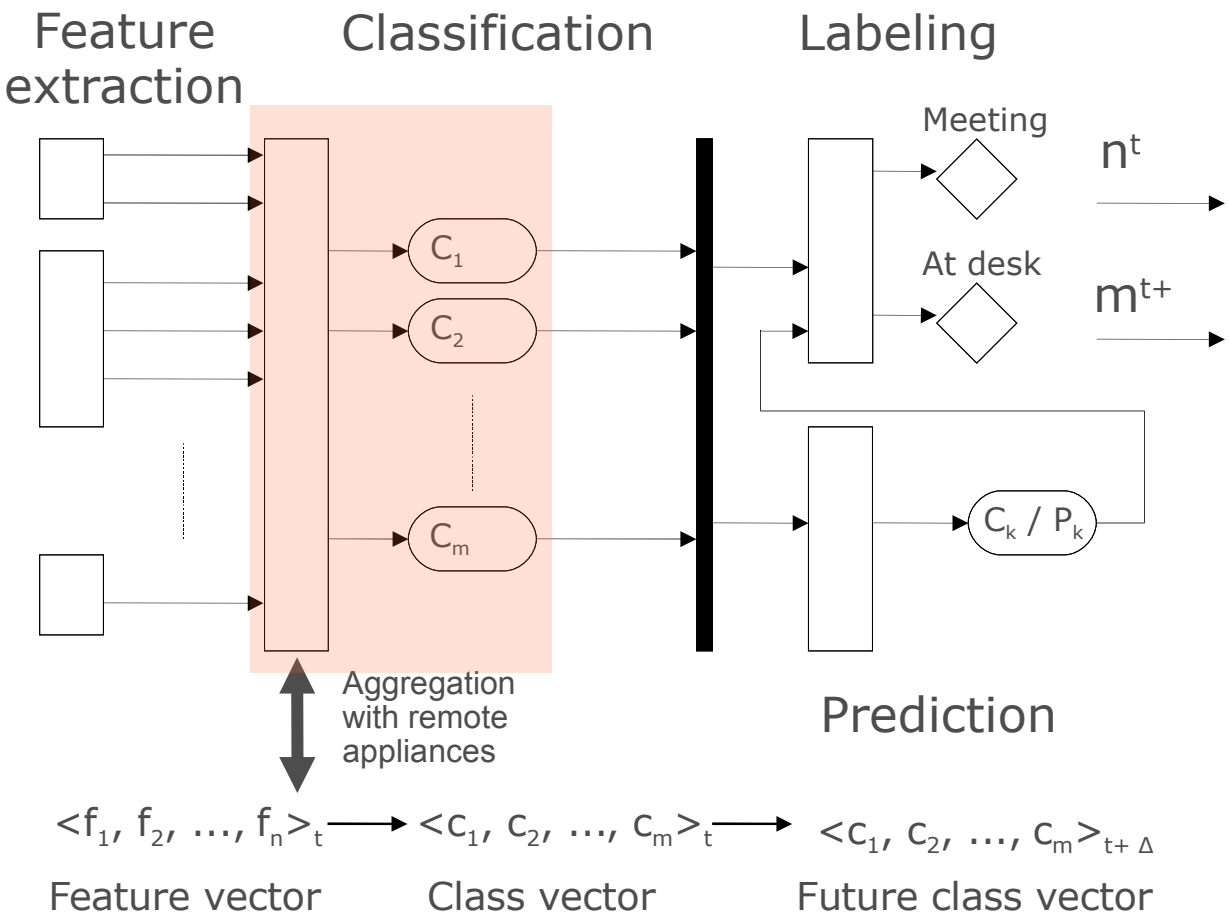


[LS 2000] H. Lieberman, T. Selker: „Out of context: Computer systems that adapt to, and learn from, context“

Motivation: The Big Picture

- Problem description of this research:

  How can the current context of a device or its user be determined autonomously from sensor information and the future context be predicted from recorded context histories?



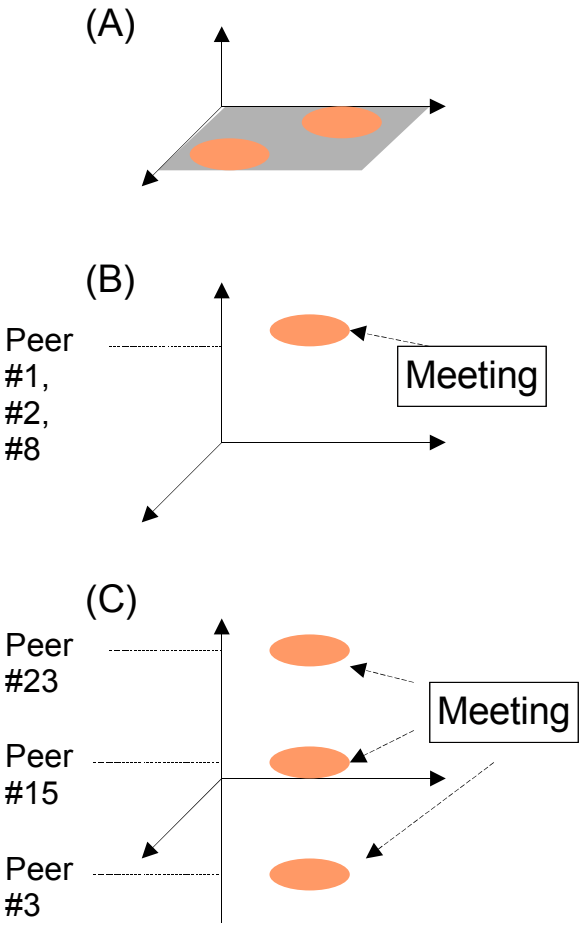
[MRF 2003a] R. Mayrhofer, H. Radi, A. Ferscha: „Recognizing and Predicting Context by Learning From User Behavior“, Proceedings of MoMM2003, OCG, September 2003

Heterogeneous Input Space

- Different types of features:
  - Numerical (continuous): e.g. brightness, heart rate, microphone
  - Numerical (discrete): e.g. number of access points in range
  - Ordinal: e.g. day of week
  - Nominal: e.g. current WLAN SSID, list of WLAN/BT devices in range

Coding Nominal Input Dimensions

- Typically: encode as multiple input dimensions
  - one (binary) feature for each nominal value ("*one-of-C*")
  - only feasible for small set of possible values, problematic with large or unbound sets (e.g. WLAN MAC addresses with 2^{48} possible nominal values)
- Problem: set of actually occurring values out of large sets is not known a priori for many sensors
- Possible solutions:
  - a) Feature vectors with variable (increasing) number of features ⇒ variable number of dimensions
  - b) Coding binary combinations as numerical discrete values
  - c) Coding binary values as numerical discrete values ⇒ multiple concurrent points in the input space

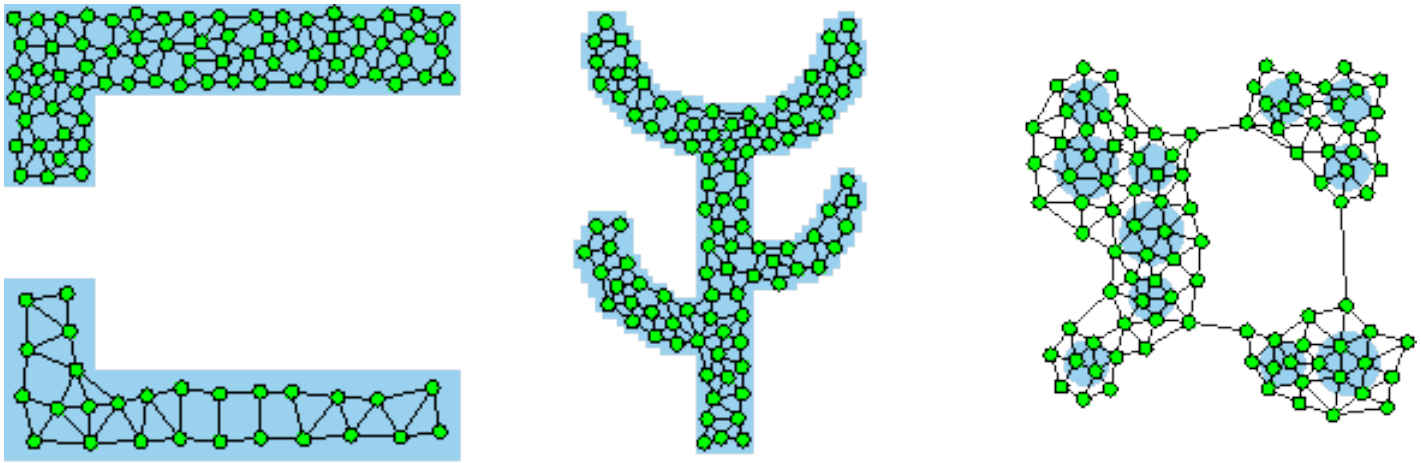


Additional Issues for Classifying Context

- High-dimensional input space
- Limited ressources
- Classification (clustering) should be:
  - „soft" classification
  - unsupervised
  - online/incremental
  - adaptive w.r.t. trends and changes in the environment
  - dynamic number of classes and dynamic structure
  - robust against noise
  - interpretability of found classes

Growing Neural Gas (GNG) „Features“

- Online/incremental and adaptive number of clusters (not many algorithms with these criteria...)
- Soft clustering possible with extensions
- Familiar concepts (family of Cell Structures, minimizes same cost function as k-means)



[Fri 2003] B. Fritzke: „A Growing Neural Gas Network learns Topologies“, Advances in Neural Information Processing Systems (7), MIT Press

Basic Methodology

Growing Neural Gas (GNG)

- Builds upon simple, spherical clusters and edges between clusters
- Algorithm starts with 2 connected clusters and inserts new ones (growing)
- For each input point, a "winning" cluster is determined and adapted in the direction of the new input; edge between "winner" and "second" cluster is created or renewed
- Edges with high age are removed, clusters without edges pruned

Lifelong GNG (LLGNG): Modification of GNG for life-long learning [Ham 2001]

- Difference: applies local criteria for each cluster to prevent the unbounded insertion of new clusters (instead of a global limit on the number of clusters)

[Ham 2001] F.H. Hamker: „Life-long Learning Cell Structures – continuously learning without Catastrophic Interference", Neural Networks (14)

Shortcomings for Context Classification

- Only defined for numerical (continuous) input space
- Output space is the vector of clusters with varying dimensionality, but: clusters do not have any high-level meaning in terms of context
- Computationally expensive (in comparison to e.g. k-means)

Mapping Heterogeneous Dimension

Dealing with nominal and ordinal inputs:

- Don't code, but use in their original range (coding changes the meaning)
- Only two operations are necessary for each feature dimension
  - similarity measure (distance metric)
  - adaptation operator

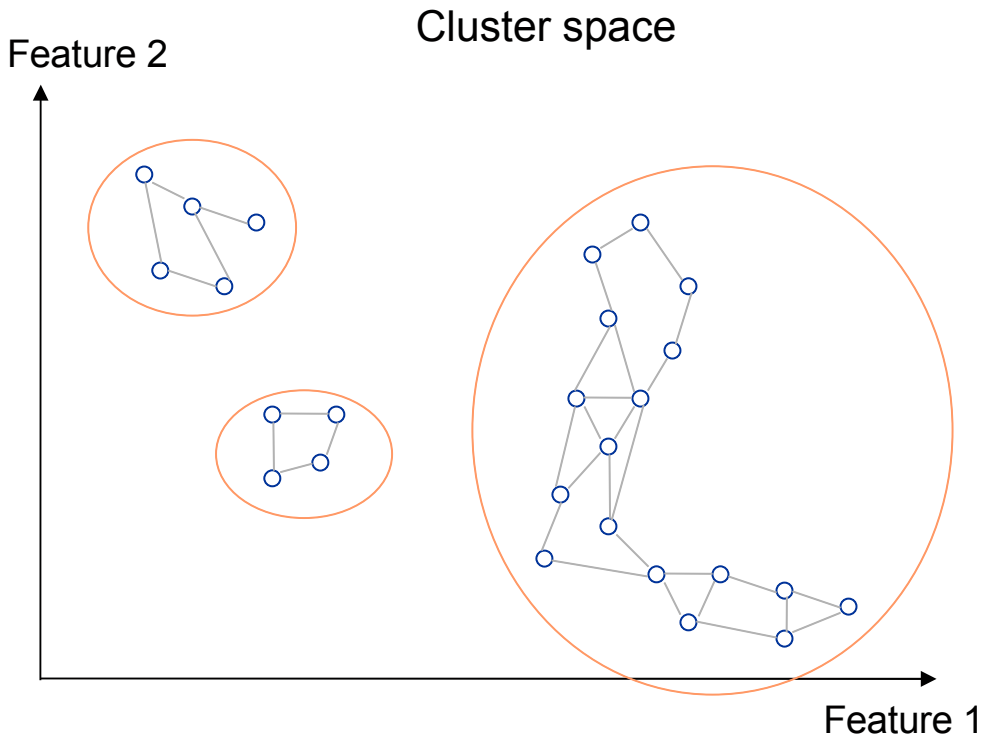
$$\alpha(f,g,a) := \begin{cases} f & if \quad a \leq 0.5 \quad if \quad f \geq g \\ g & if \quad a > 0.5 \quad if \quad f < g \end{cases}$$

[MRF 2003b] R. Mayrhofer, H. Radi, A. Ferscha: „Feature Extraction in Wireless Personal and Local Area Networks“, Proceedings of 6th IFIP MWCN 2003, World Scientific, October 2003

Meta Clusters for Higher-Level Classification

- Extensions to LLGNG for context recognition:
  - Usage of a heterogeneous feature space
  - Utilizing the internal topology to allow arbitrarily shaped clusters (components) with dimensionality of input space

⇒ Meta clusters as additional level of abstraction

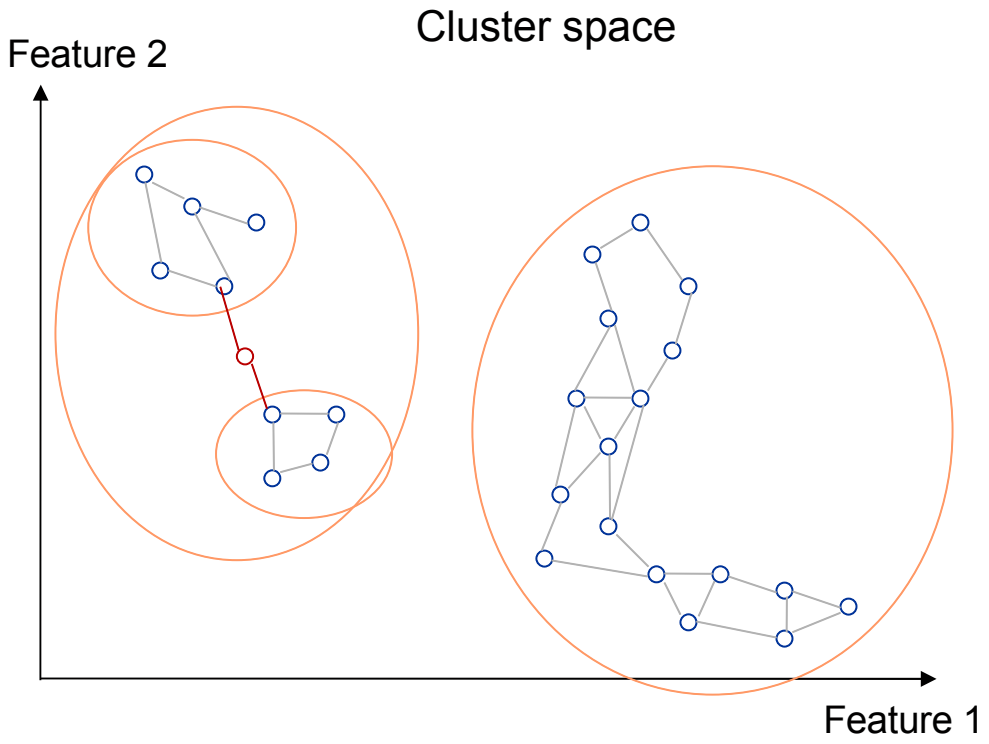


Meta Clusters for Higher-Level Classification

- Connectivity of clusters changes during run-time
- Creating and removing edges can have influence on connectivity of meta-clusters
  - Merge: new edge created between meta-clusters when input point occurs in between
  - Split: aged edge can cause a meta-cluster to fall apart into two

⇒ Merge can be prevented by “veto” from higher layers when both meta-clusters already mapped

⇒ Need to decide which meta-cluster retains mapping during split



Dealing with Limited CPU Resources

Splaytrees

- Finding nearest and second nearest cluster to an input point is a common task but expensive because similarity measure needs to be computed for every dimension
- Limit the amount of comparisons when they can not be optimized
  - Splaytrees keep recently accessed nodes on top of a sorted tree
  - Assumption 1: input points (feature values) are not evenly distributed in cluster space
  - Assumption 2: input points do not change spontaneously too often (complete change in sensor readings means context switch)

Caching

- Deletion criterion depends on local similarity of clusters
- Distance of a cluster to each of its neighbors needs to be computed

⇒ Computationally expensive because similarity measure needs to be computed for *all* neighbors in *all* dimensions

- Caching local similarity at each node improves performance (but recalculation necessary when a cluster or one of its neighbors changes its position)

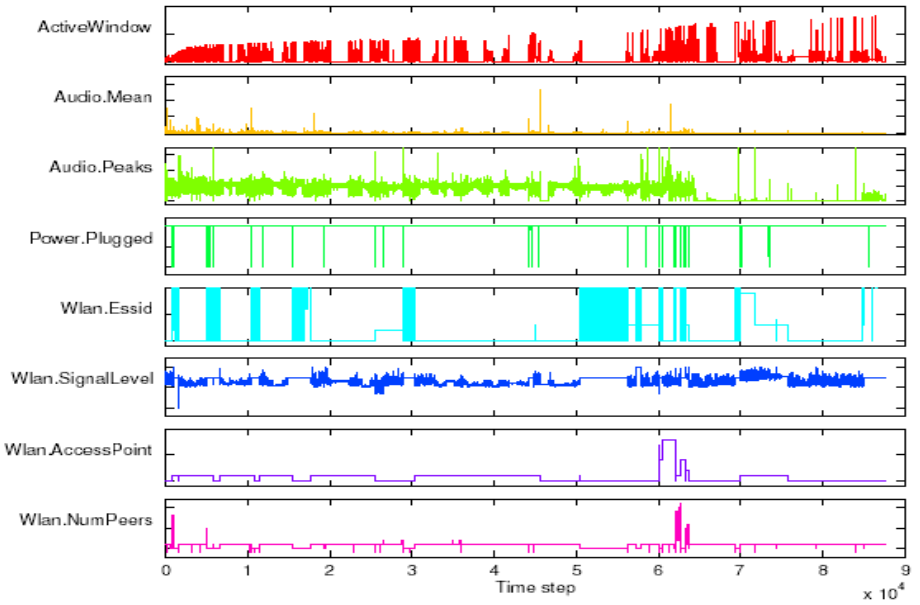
Dealing with Limited CPU Resources

Preventing split before removal

- Procedure for removing edges:
  - remove the edge
  - check if previously adjacent clusters are still connected (caching!)
  - if no longer connected, assign new meta cluster ID to one of the meta clusters (split)
- When a single cluster is split from a meta cluster, it will be deleted afterwards (pruning: clusters without any edge are deleted)
- Do not complete the split in such a case as it is not necessary

Initial Evaluation with Real-World Data

- Recorded with a standard laptop
- Recording completely in the background
  ⇒ unobtrusive operation
- Time frame: ca. 2 months
- 28 feature dimensions
- ⇒ 90000 data points



Evaluation of k-means, SOM and LLGNG

- K-means: smallest error with 6 clusters: 0,7451
- SOM: smallest error with 71x21 (=1491) neurons in output layer: 0,5659
- Extended LLGNG: 9 meta clusters with 109 clusters: 0,0069

⇒ LLGNG produces a smaller error with less clusters than SOM, extended variant offers the additional concept of meta cluster

Results of Optimizations

- Evaluation system:
  - Intel P3 Mobile, 850 MHz, 256 MB RAM
  - Linux 2.4.22, Glibc 2.3.2, g++ 3.3.2 with Debugging-Code
- 131041 points of artificial test data (simulated sensor values every 2 minutes for a period of 26 weeks)

	Run time	Faster by	Speedup
Splaytree only	165 s	0 %	1
With Caching	88 s	46,67 %	1,875
With Split prevention	166 s	- 0,6 %	0,994
All	86 s	**47,87 %**	**1,919**

Open Research Questions

- Better handling of noise in input space
- User-level feedback loop for meta-cluster mapping during merge and split
- Dealing with meta-clusters where some input dimensions are not contributing to classification (detecting clusters in sub-spaces)
- Analyzing influence of heterogeneous input space on various criteria
- Preserving topology between meta-clusters: mapping to lower dimensionality?
- Ideas for solutions:
  - Maintaining statistical values per meta-cluster (e.g. “center”, maximum distance between neighbors, average distance, “extent” of the whole meta-cluster, etc.)
    ⇒ possibly sufficient for topology preservation
  - Tracking global parameters (e.g. error frequency w.r.t. to number of clusters)
  - Maintaining a “change vector” for each cluster (last change and moving average)
    ⇒ possibly helpful for detecting and dealing with noise

„I find that a great part of the information I have was acquired by looking up something and finding something else on the way.“

Franklin P. Adams
Journalist

Thank you for your attention!

Slides: http://www.mayrhofer.eu.org/presentations
Sources: http://www.mayrhofer.eu.org/contextprediction
Later questions: rene@mayrhofer.eu.org

OpenPGP key: 0xC3C24BDE
7FE4 0DB5 61EC C645 B2F1 C847 ABB4 8F0D C3C2 4BDE