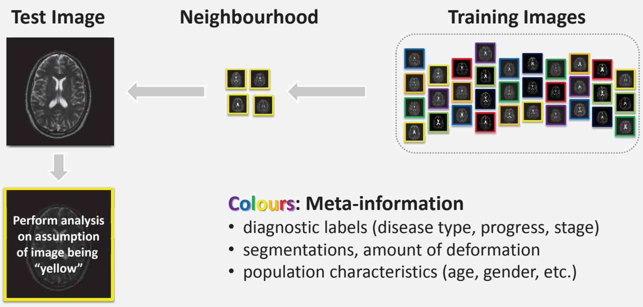
Neighbourhood Approximation Forests

Ender Konukoglu, Ben Glocker, Darko Zikic, Antonio Criminisi
Microsoft Research Cambridge

Microsoft Research

Neighbourhood-based Approaches (NbA)

Perform analysis on a new image by using the information from "neighbours" images



Colours: Meta-information
- diagnostic labels (disease type, progress, stage)
- segmentations, amount of deformation
- population characteristics (age, gender, etc.)

Properties of NbA
1. Neighbourhood is defined through pairwise distances on meta-information
2. No meta-information given for new image

Example Applications
- atlas selection: [Aljabar 2009], [Sabuncu 2009]
- label propagation: [Wolz 2010], [Coupe 2011]
- "manifold methods": [Hamm 2010], [Gray 2011]

Neighbourhood Approximation Forests (NAF)

Learning the relationship between appearance and meta-information

A general method for learning features which:
1. Captures neighbourhood defined by arbitrary distances
   [by using a distance-based objective during training stage]
2. Allows efficient k-NN estimation at test time
   [by exploiting inherently efficient decision trees]

This ***automatic*** and ***objective-based feature learning*** is in contrast to ***manual*** and ***heuristic feature design***

Training: Learning Neighbourhood Structure

Learn Neighbourhood Structure based on application-specific distance

Image-based appearance features
- Can be evaluated at test time, unlike meta-information-based distances
- High-dimensional feature space

Training: Determine the split function for each node based on features
→ Generates tree structure, and
→ Leaf-Statistics (Indices of images reaching leaf)
- Select feature space dimension, and threshold, to optimize the objective

Objective Function: Gain in compactness with respect to cluster size
→ Coupling of feature space and meta-information

Cluster Size

$$C_\rho(A) = \frac{1}{|A|^2}\sum_{I\in A}\sum_{J\in A}\rho(I,J)$$

- Images I, J
- Distance measure ρ
- A : Set of images at a given node

Gain in Compactness

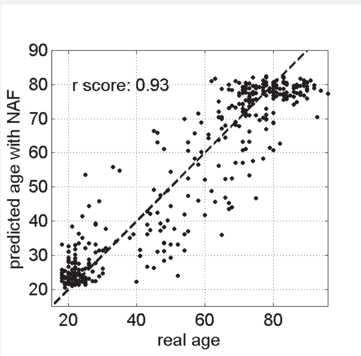
$$GC \triangleq C_\rho(A) - \frac{|A^L|}{|A|}C_\rho(A^L) - \frac{|A^R|}{|A|}C_\rho(A^R)$$

- A^L, A^R : Left/right subsets for split

Example Application 1:
Age Regression from brain MRI scans

$$\rho(I,J) = |\mathrm{age}(I) - \mathrm{age}(J)|$$

Experiment Setup:
- 355 images
- Leave-one-out tests
- 700 trees, depth 12
- 15-NN for age regression
- Features: Image Intensities at random locations



Finding Neighbours is Difficult

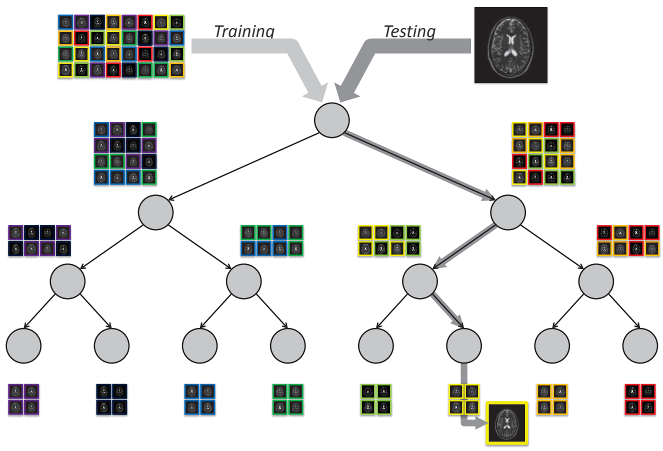
No meta-information for test image, only appearance is available

1. Direct search for neighbours:
   - Computationally expensive, e.g distance is amount of deformation
   - Not feasible: distance requires unknown meta-information, e.g. disease stage, segmentation
2. Approximate neighbours using **appearance-based features** and efficient search
   - Clustering: k-means [Sabuncu 2009], tree-based [Nister 2006], [Gray 2011]
   - Hashing [Weiss 2008], [Strecha 2012]

Definition of descriptive features is highly non-trivial:
- Should be low dimensional
- Need to capture the underlying neighbourhood structure
- Commonly hand-crafted and/or based on heuristics, e.g. similarity measure in ROI

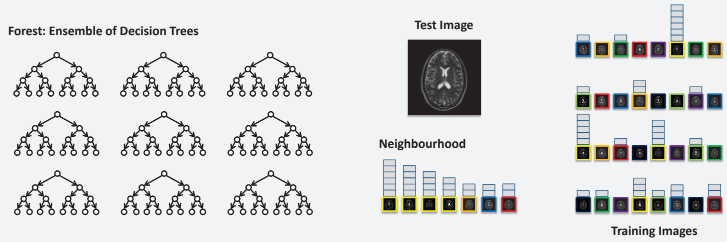
Neighbourhood Approximation Tree

Feature space partitioning with respect to distance-based objective



Testing: Approximate Neighbours for Test Image

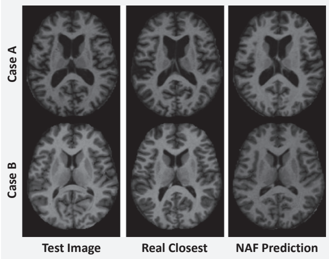
Count the number of co-appearance in leaf nodes of test and training images



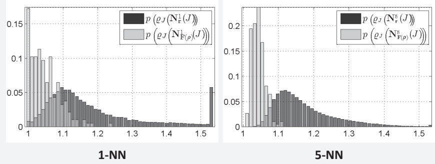
Example Application 2:
Choosing the closest images for non-linear registration

$$\rho(I,J) = \int_\Omega \log|\mathrm{Jac}(\Phi)| + \log|\mathrm{Jac}(\Phi^{-1})|\, dx$$

Experiment Setup:
- 169 training images / 186 test images
- 1500 trees, depth 6
- Features: Intensity difference between two random locations



Assessing Approximation Quality:
- histogram over all 186 tests: ratio of sum of distances to real neighbours and approximated neighbours predicted by NAF (value of one is perfect)
- two different cases: 1-NN (closest image) and 5-NN
- ratios are compared with random selection of neighbours



Runtime for one test image:
NAF takes maximum 10.2 seconds (C++ / Intel Xeon® 2.27 GHz)
(compared to 169 nonlinear registrations, with 1.9h on average)