ETH zürich D INFK

CVPR 2015 BOSTON

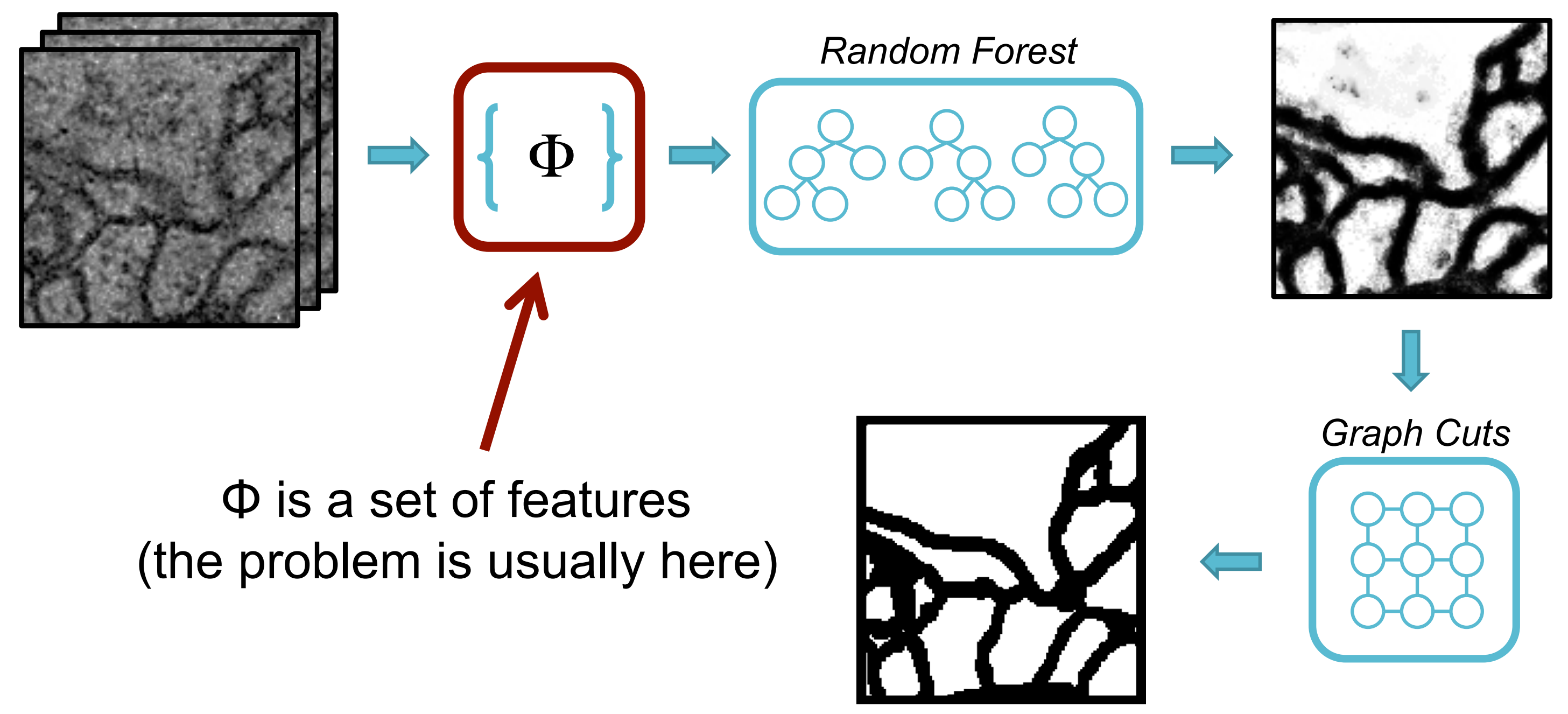
Transformation-Invariant Convolutional Jungles (TICJ)

Dmitry Laptev and Joachim M. Buhmann,
Machine Learning Institute, ETH Zurich, Switzerland

1. Computer Vision Features

Usual segmentation (and classification) pipeline:

1. Describe every patch (or image) with a feature vector.
2. Pass these vectors to a machine learning algorithm.
3. Smooth the results to get the final segmentation/classification.

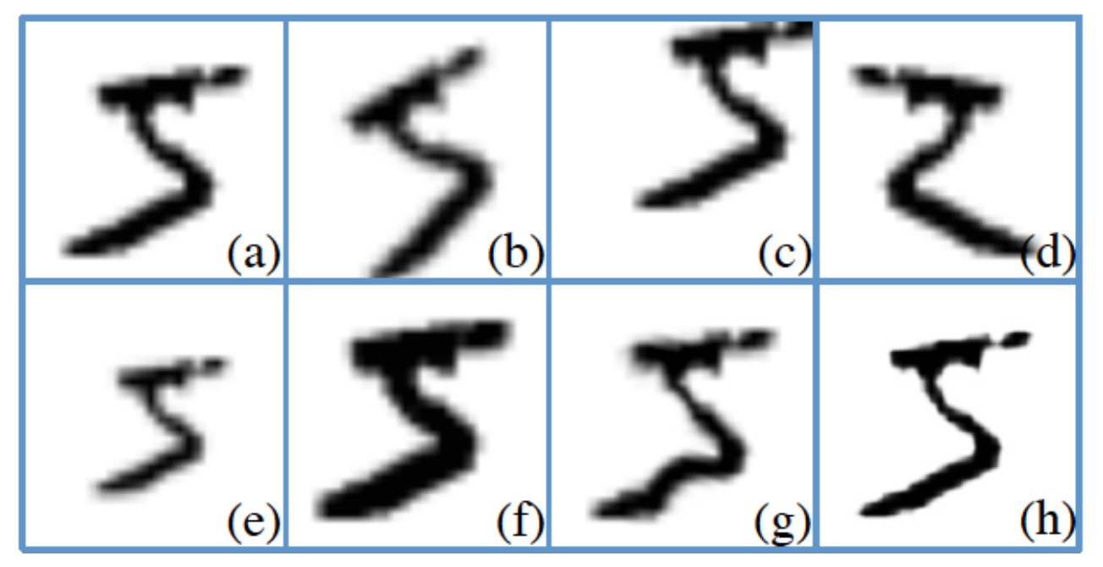


Φ is a set of features
(the problem is usually here)

Features can be:

1. predefined to encode expert insights (generic vs. domain),
2. learned automatically (supervised vs. unsupervised).

2. Transformation Invariance



Human can easily recognize images under many different transformations: rotations, shifts, mirroring, scale, morphological operations, non-linear distortions, color change.

Most data sources are subject to some variances, that can usually be specified a priori:

1. for hand-written digits, only slight rotations are possible,
2. rigid object recognition can include only rigid transformations,
3. in medical domain (microscopy), scale often cannot change.

Transformation-invariance can lead to better generalization of Computer Vision algorithms:

1. Dataset augmentation: include transformed images.
2. Predefined features: SIFT (scale-invariant), RIFT (rotation), Line Filter Transform (rotation), etc.
3. Learned features: Bag of Visual Words (shift), Sparse Coding (shift), Deep Neural Networks (limited transformations), ours.

3. Transformation-invariant features

Transformation-invariant feature parameterized by θ:

$$f_\theta(x) = \max_{\phi \in \Phi} \theta^T \phi(x)$$

Vectorized image (patch)

Transformations considered

Convolution kernel (parameters)

Lemma 1. The feature of the image X defined above is transformation-invariant if the set Φ of all possible transformations forms a group[1], i.e. satisfies the axioms of closure, associativity, invertibility and identity.

Feature Learning: looking for the best parameters θ.

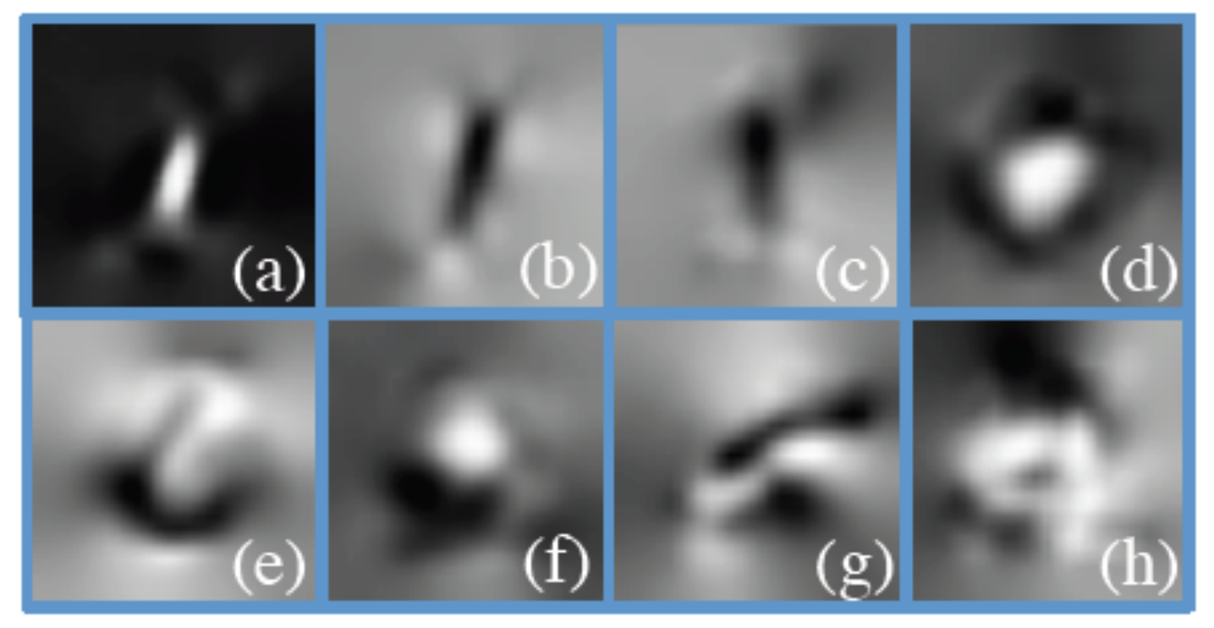
Kernel gradient regularization

$$\theta = \arg\min_\theta E(\theta) = \arg\min_\theta \quad \lambda ||\Gamma\theta||_2^2 + \sum_{i:\ y_i = c_1 \text{ or } y_i = c_2} \left(f_\theta(X_i) + [y_i = c_1] - [y_i = c_2]\right)^2$$

[•] equals 1 if • is true and 0 otherwise

Two classes to split

The problem is convex, but not continuously differentiable.



Examples of learned features for membrane segmentation. Features (a)–(c) detect direct membranes, (d) denotes the contrast of the center pixels comparing to the surroundings, (e) and (f) detect corners and curvatures (non-straight membranes), (g) and (h) – high-frequency features.

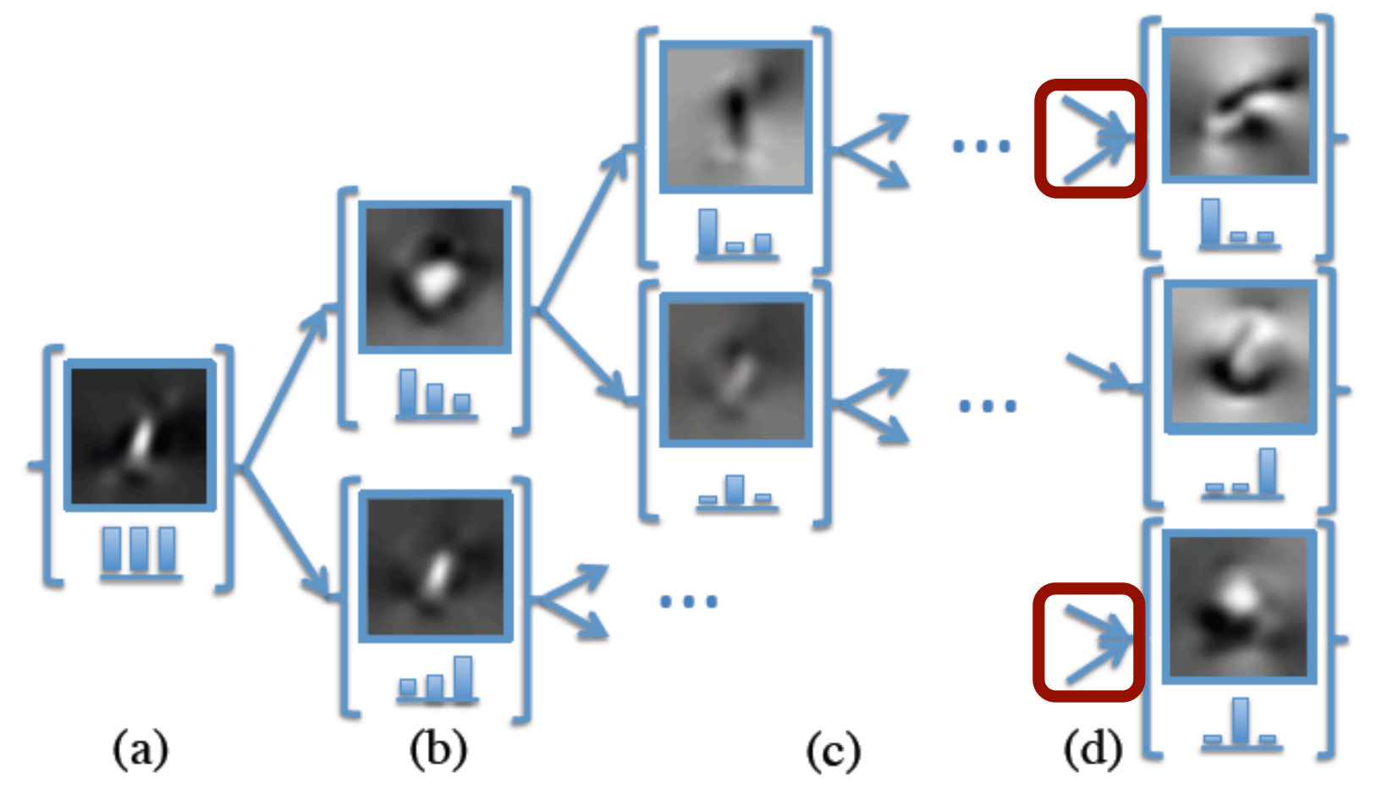
[1] If Φ is not a group, we can modify the feature definition such that transformation-invariance still holds.

4. Convolutional Decision Jungles

Oblique decision trees:
- use multivariate, not univariate splits.

Decision jungles:
- merge nodes when max width is achieved.

1. Given a dataset and a pair of classes, we can find feature parameters θ;
2. θ defines a predicate that splits the data: $l_1 = \{i : f_\theta(X_i) > 0\}$ $l_2 = \{i : f_\theta(X_i) \le 0\}$
3. Selecting new pairs of classes, we can recursively build a tree from l_1 and l_2
4. When maximum width M is achieved, merge similar subsets of the dataset.



(a) shows the root node (the whole dataset is an input). Feature parameters θ are learned and the dataset is split in two subsets (input for two other nodes) (b). The algorithm proceeds until the maximum width M is achieved (c). Then some of the data subsets close in distribution can be joined together (d).

5. Experiments

1. Neuronal segmentation.

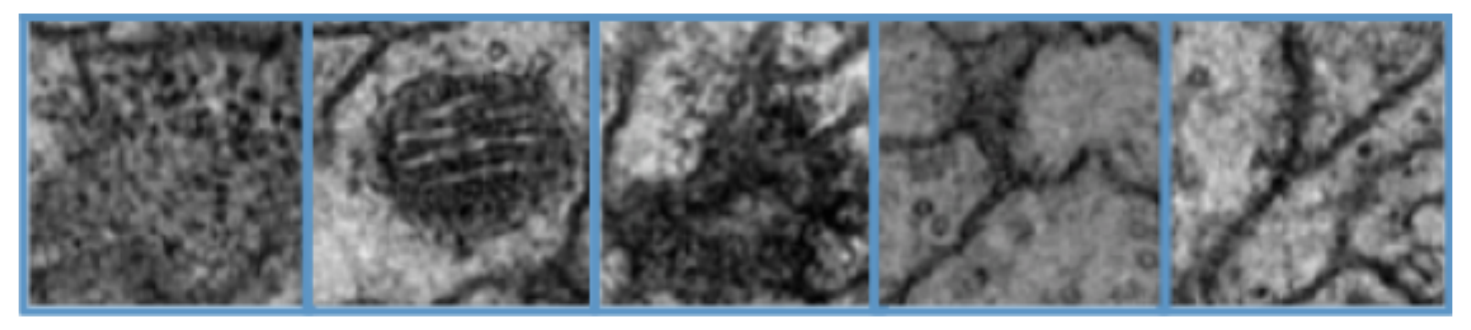
Data:
- ISBI 2012 challenge.

Two classes:
- membrane vs. neuron.

Images:
- 60 images 512x512,
- 31x31 patches.

Invariances:
- rotations (24 angles).

TICJ results:
- same F-score as for CNN,
- 100 times faster than CNN.



Example patches from the dataset: sometimes very blurred and unclear.

Method	1–F-score	Time
RF+3d	7.9	unknown
CDT	6.8	8h (CPU)
CNN	**6.0**	7d (GPU)
TICJ (ours)	**6.0**	**4h (CPU)**

Best teams results: Random Forest with hundreds of features, Convolutional Decision Trees, Deep Neural Networks.

2. Face recognition

Data:
- Yale face database,
- very small (165 images).

15 classes for:
- 15 individuals,
- 11 images per person.

Images:
- cropped version 32x32.

Invariances:
- small shifts, rotations,
- illumination-invariance.

TICJ results:
- best with no external data.

Example faces of one person from the dataset.

Method	Error, %
Cai et al.	18.3
Cai et al. (u)	14.7
Hua et al.	13.2
Shan et al.	**8.2**
TICJ	**12.9**

TICJ outperforms every team except one. Shan et al. shows significantly better results but uses additional training data.

Conclusions

Transformation-Invariant Convolutional Jungles:
- much faster than deep neural networks (4 hours vs. 7 days),
- in some tasks can match their performance,
- require no special hardware (but can also benefit from GPU),
- can work with small datasets (165 images in Yale dataset).

Consist of three main components:
- max-pooling over transformations (ensures invariance),
- strictly convex regularization term (produces smooth kernels),
- decision jungles algorithm (prevents overfitting and speeds up).