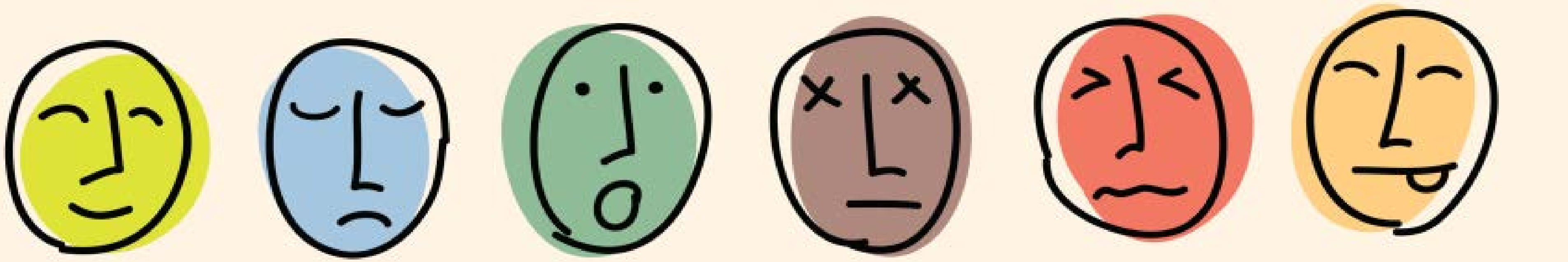
AMFG 2015

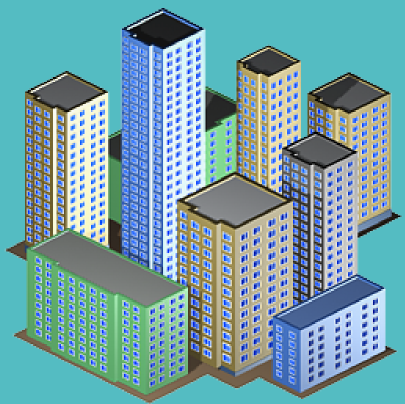
Sparse Coding Trees with Application to Emotion Classification



Kevin H.C. Chen
Marcus Z. Comiter
H. T. Kung
Bradley McDanel
Harvard University

Application Motivation

Emotion Classification for IoT and Beyond



Business Applications

User feedback systems, advertising, security systems

Politics

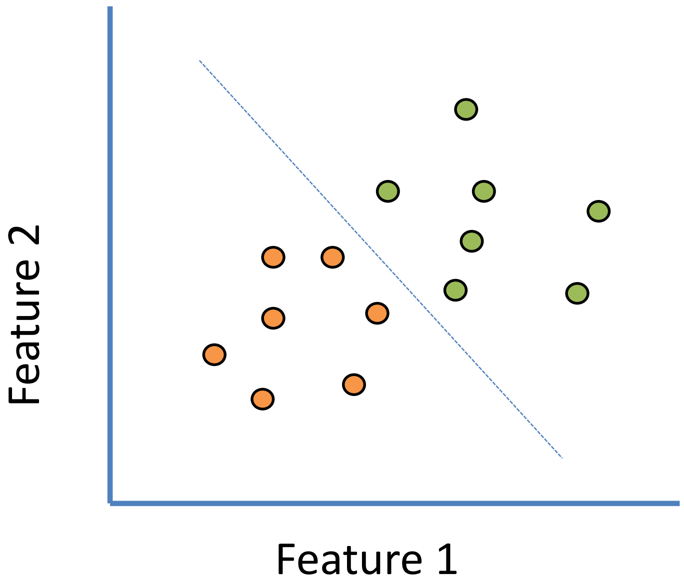
Automatic derivation of voter preferences, focus group testing

Medicine

Additional metrics for patient care, helping children with autism

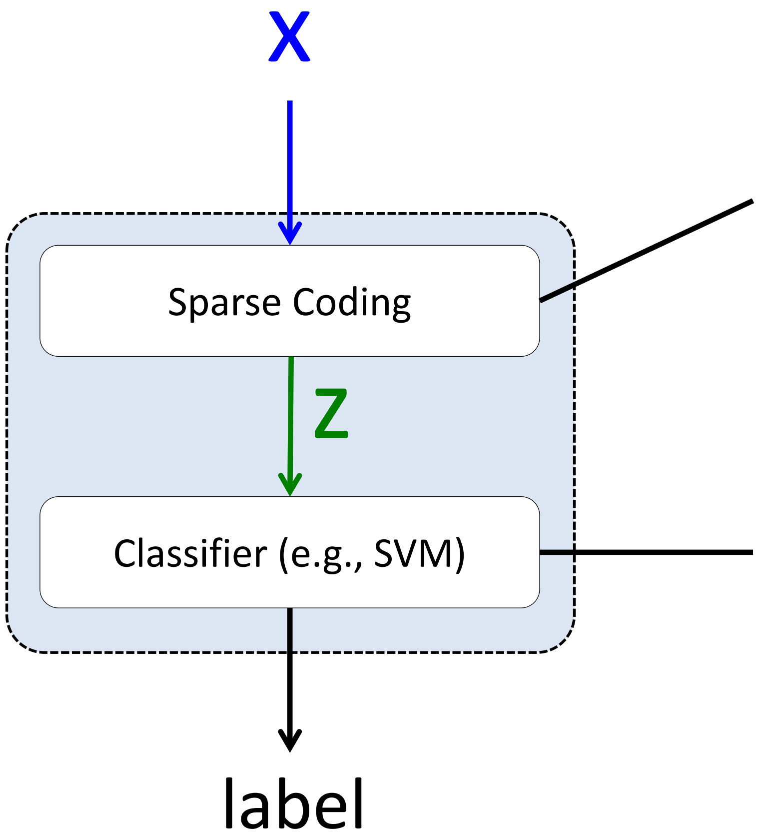
Methodology Motivation

Machine Learning and Unsupervised Feature Extraction



- Sparse coding makes data more linearly separable
- Labels are not required

Sparse Coding Pipeline for Classification



Unsupervised

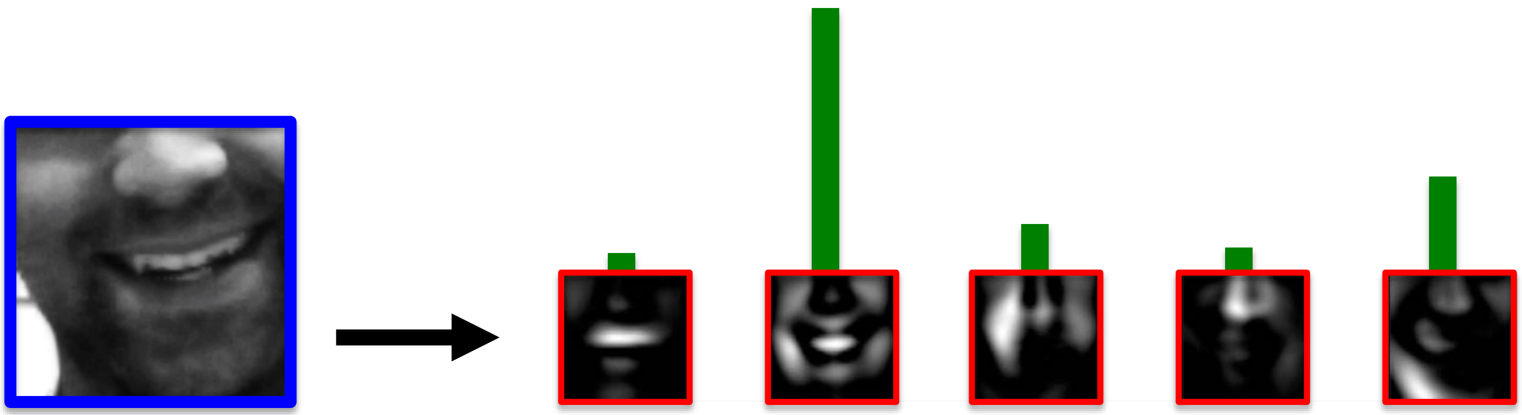
Transform data into feature representation

Supervised

Prediction with simple classifiers such as SVM

Representation by Sparse Coding

Express the **input signal (x)** as the **weighted (z)** sum of a few **features (D)**



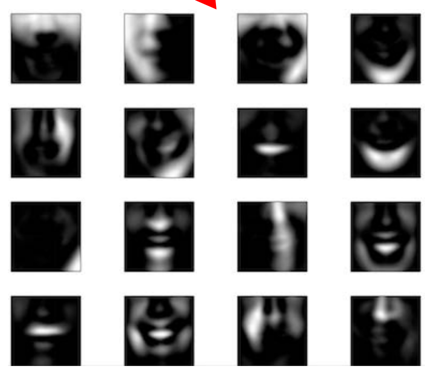
$$\mathrm{argmin}_z \; \|x - Dz\|_2^2 + \lambda \|z\|_0$$

Note: we can also penalize L1 norm instead of L0 norm

Dictionary Learning

- Finds common patterns in training data
- Solved by alternating updates of D and Z

$$\mathrm{argmin}_{D,Z} \; ||X - DZ||_2^2 + \lambda ||Z||_0$$



Sparse Coefficients

Our Enhancement to SC

Sparse Coding tree (SC-tree)
to learn features with hierarchy

Non-negative constraints
to mitigate over-fitting in SC

Mirroring
to increase variation tolerance

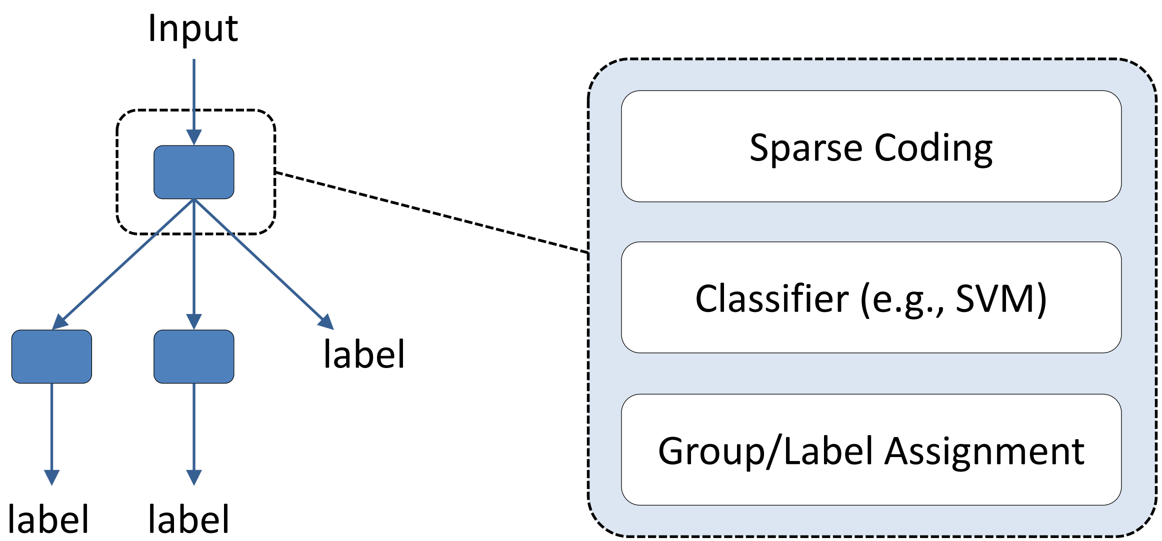
Sparse Coding Tree
Learning Features for Hard Cases

- Some discriminating features can be subtle
- Finding clusters within clusters, similar to how hierarchical k-means works

Fear can be confused with happiness because they both display teeth

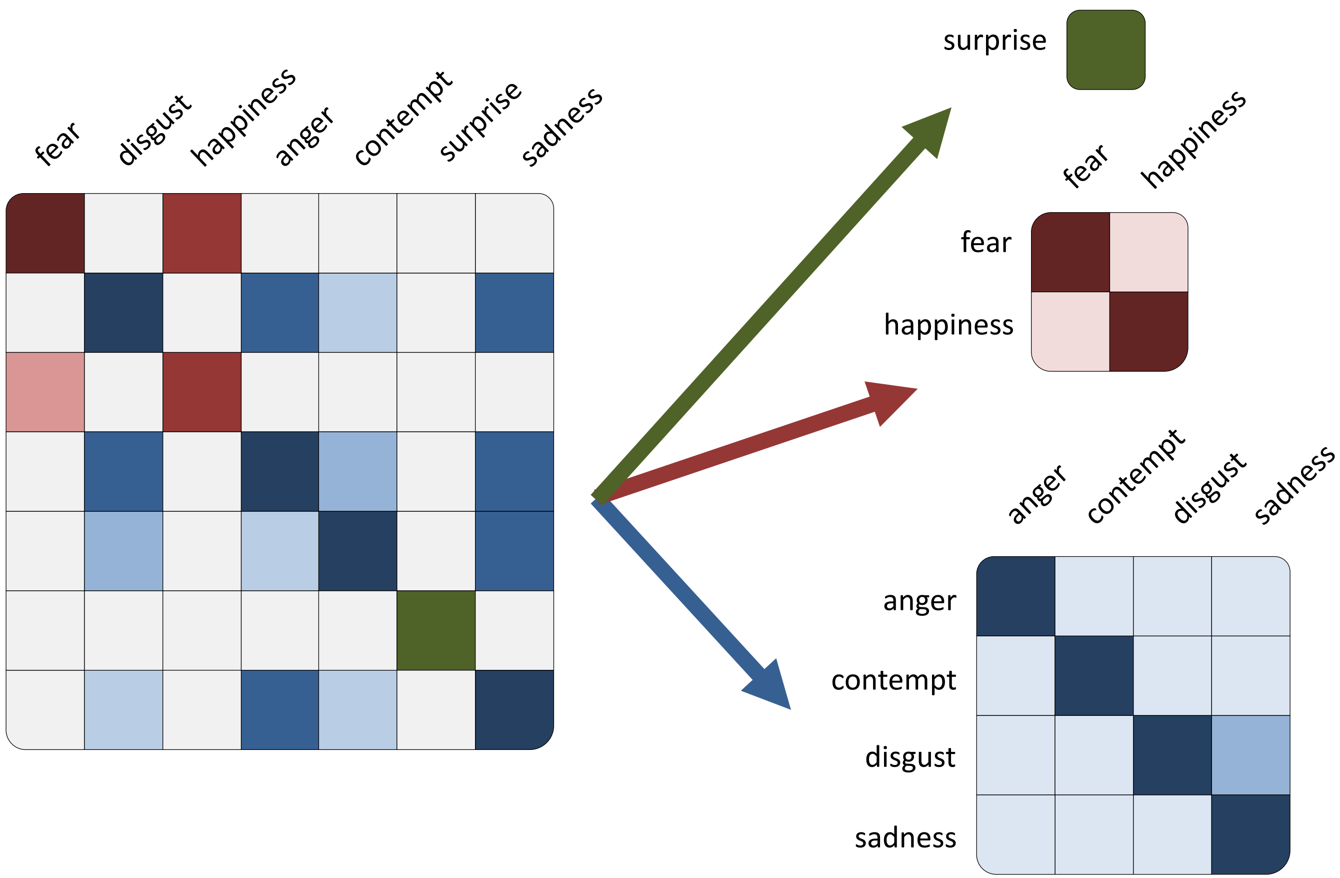
Constructing Sparse Coding Tree

If certain classes get confused consistently,
put them through another layer of feature extraction

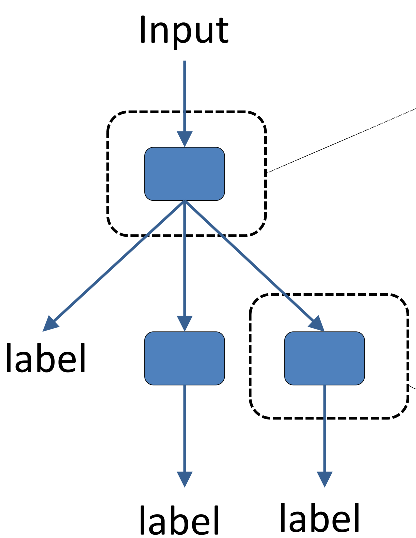


Branching in Sparse Coding Tree

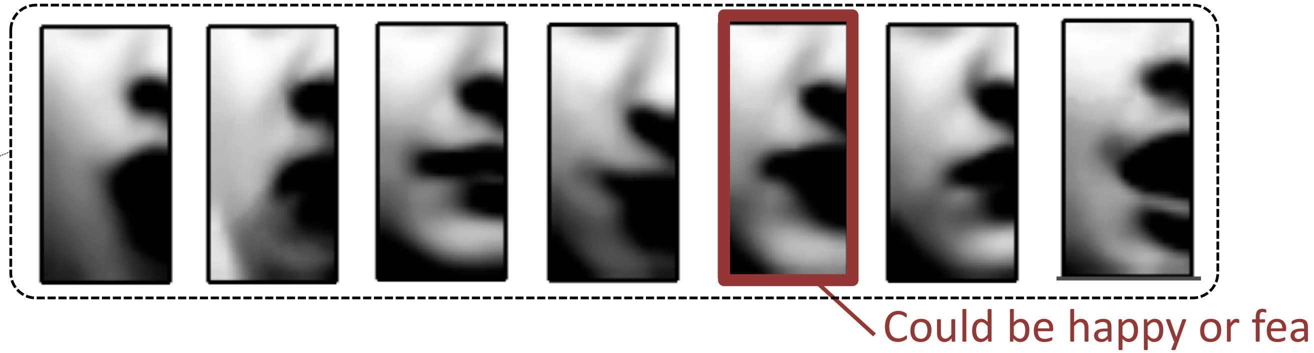
Based on the confusion matrix from the coarse predictor



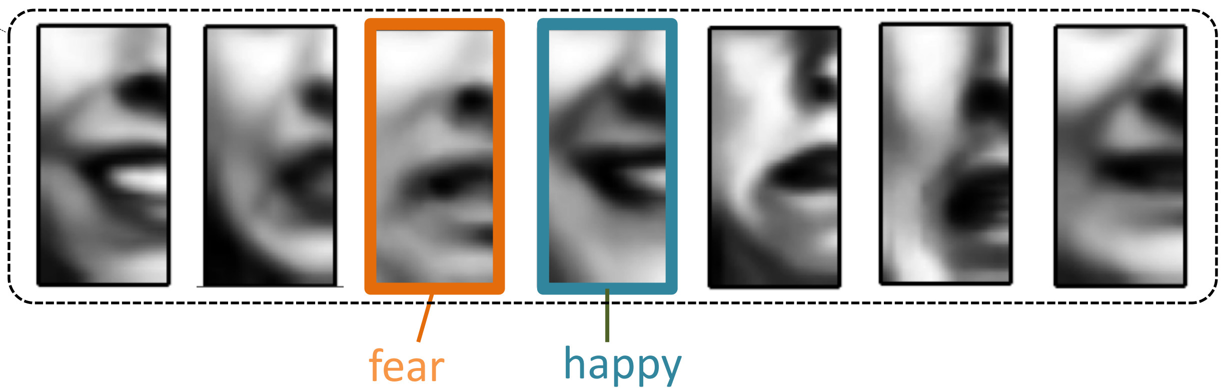
Features Learned in SC-tree



Features learned in the root node

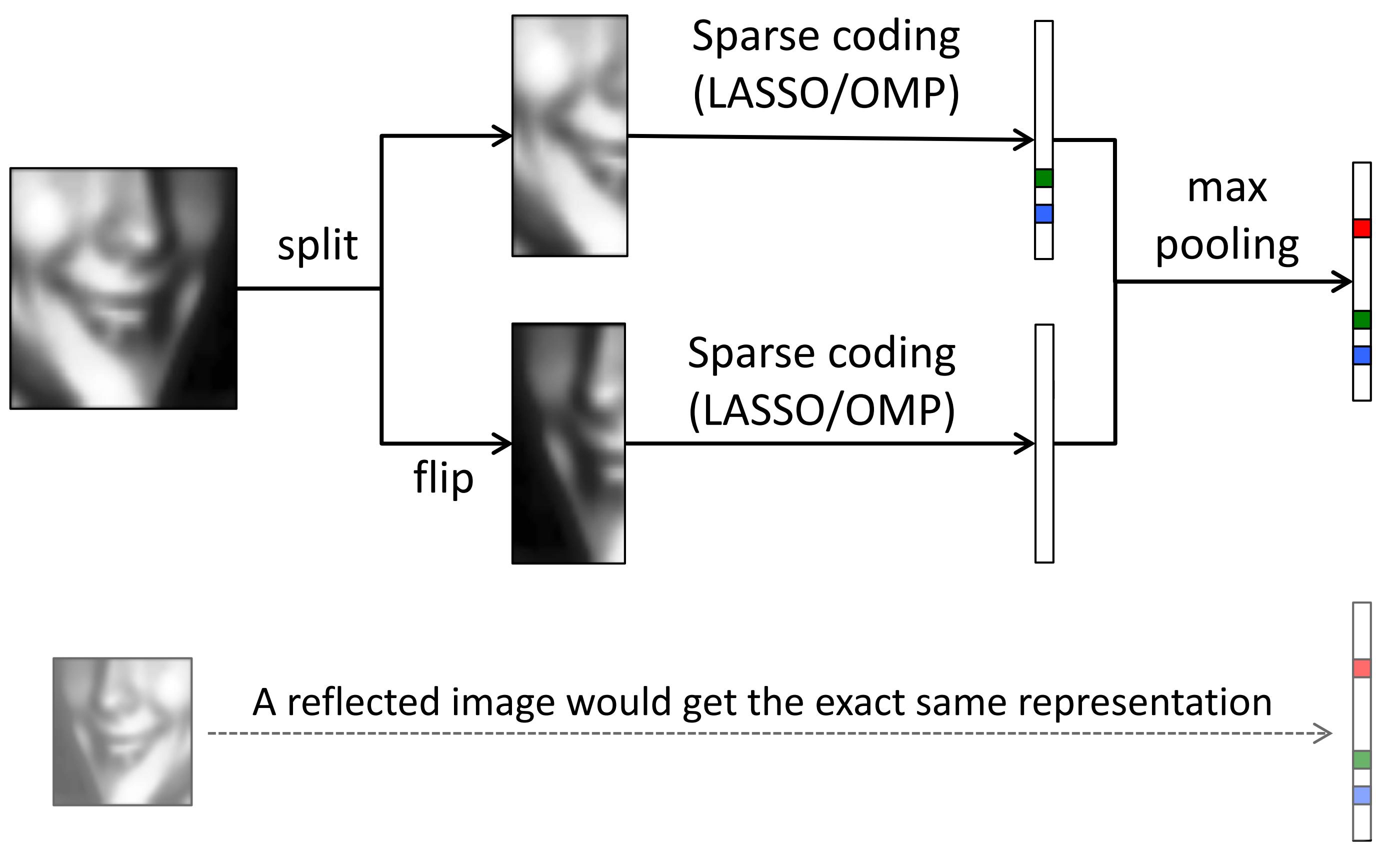


Features learned in a happy v.s. fear node



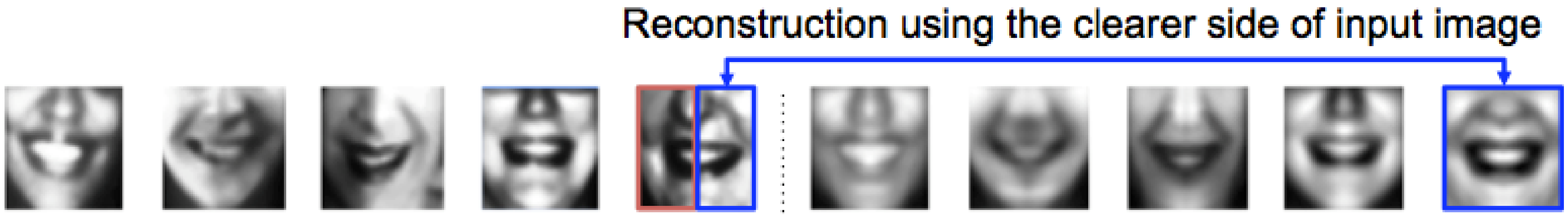
Mirroring for Reflection Invariance

Using max pooling to capture the horizontal symmetry inherent in emotion classification



Improved Robustness with Mirroring

With max pooling, we always pick up response from the side of face with stronger features



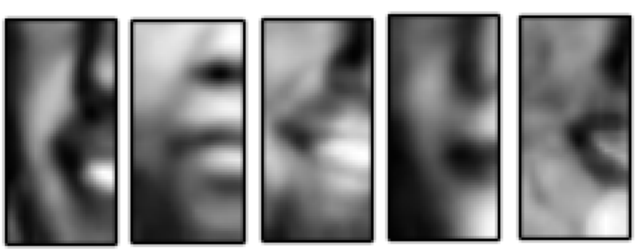
(a) Original images (b) Reconstructed images

Nonnegative Sparse Coding

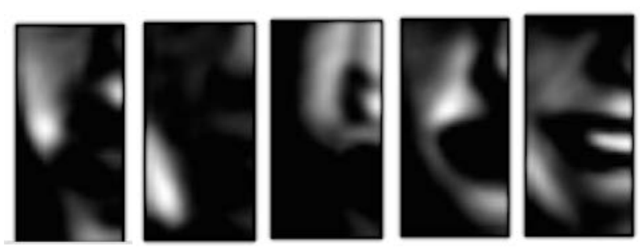
Nonnegativity prevents cancelation of components, and therefore mitigates over-fitting

$$\operatorname{argmin}_z \; ||x - Dz||_2^2 + \lambda ||z||_0$$
$$\text{s.t. } D \geq 0, \; z \geq 0$$

D without NN-constraint



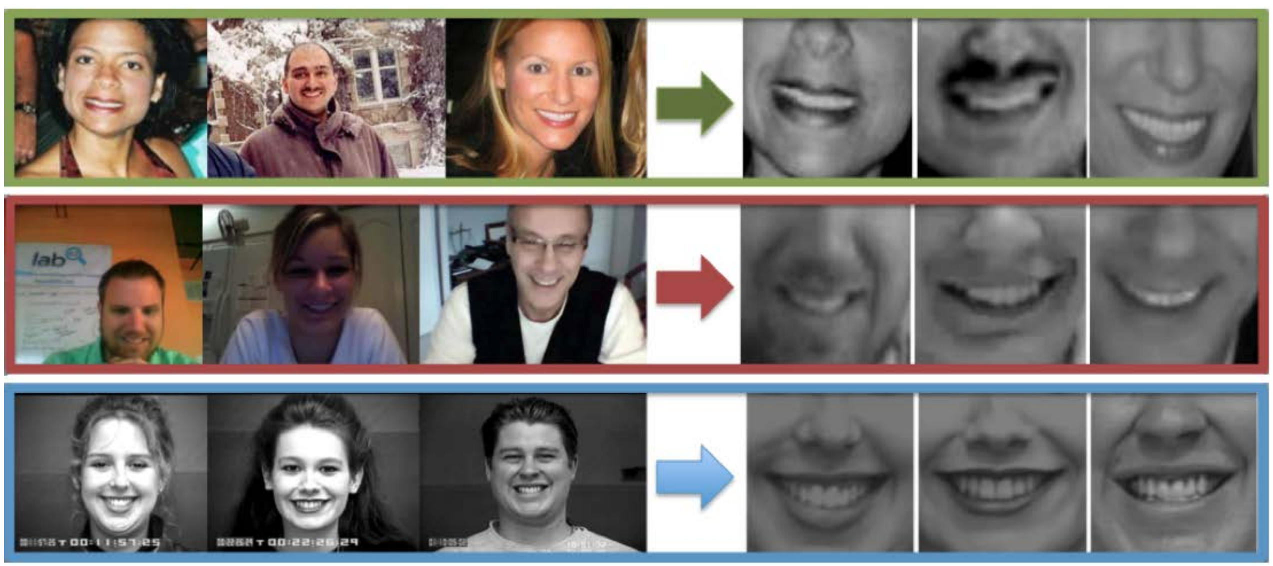
D with NN-constraint



Tends to learn regional components

Datasets

Multi-class	Cohn-Kanade Extended Dataset (CK+)
Multi-class	Emotions in the Wild Dataset (EitW)
Binary	GENKI-4K Dataset
Binary	AM-FED Dataset



AM-FED

GENKI

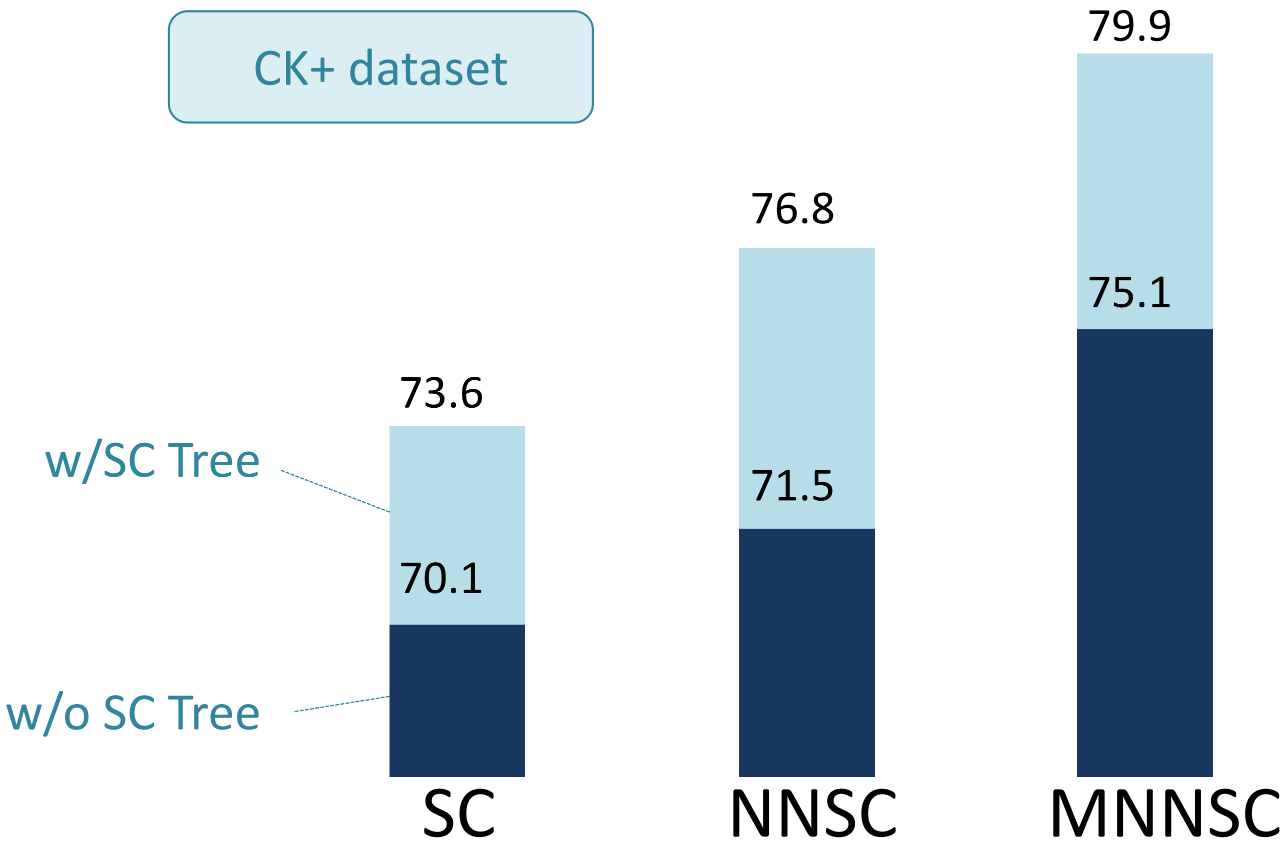
CK+

original data

after pre-processing

Performance on Emotion Classification

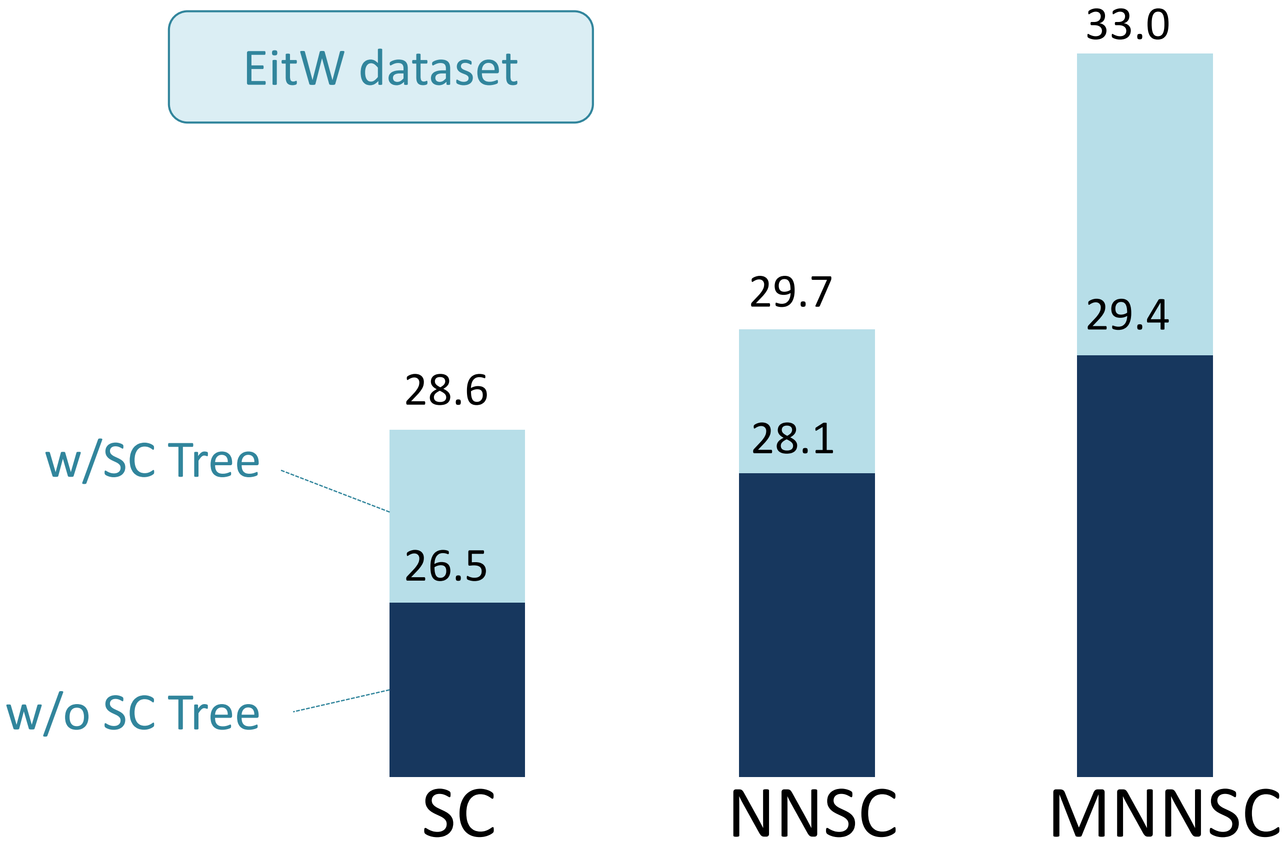
The sparse coding tree improves the performance of our pipeline consistently.



Results reported in average recall

Performance on Emotion Classification

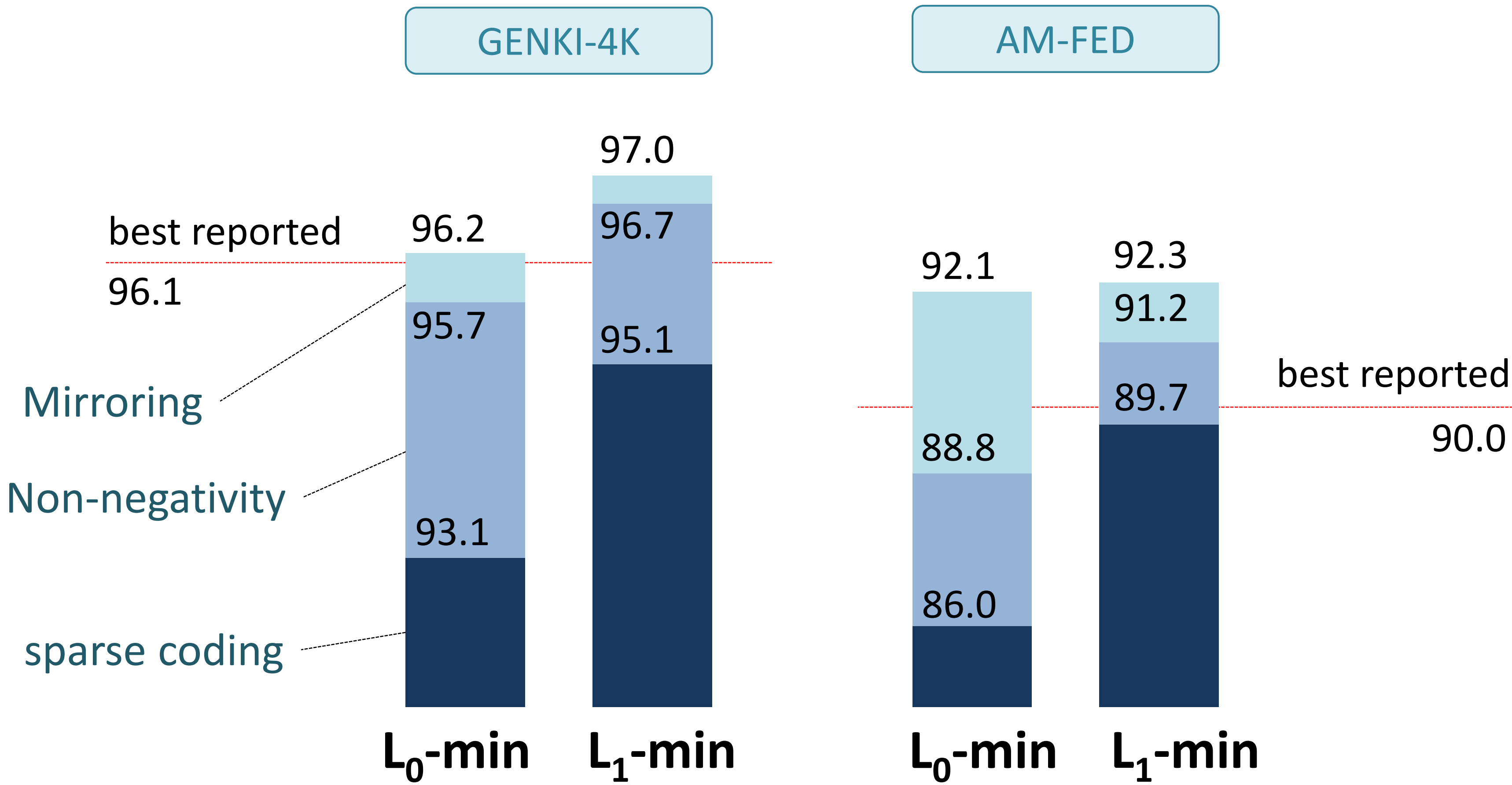
The sparse coding tree improves the performance of our pipeline consistently.



Results reported in average recall

MNNSC Performance

with Mirroring and the non-negativity constraint,
even greedy methods like OMP (L0) can be competitive



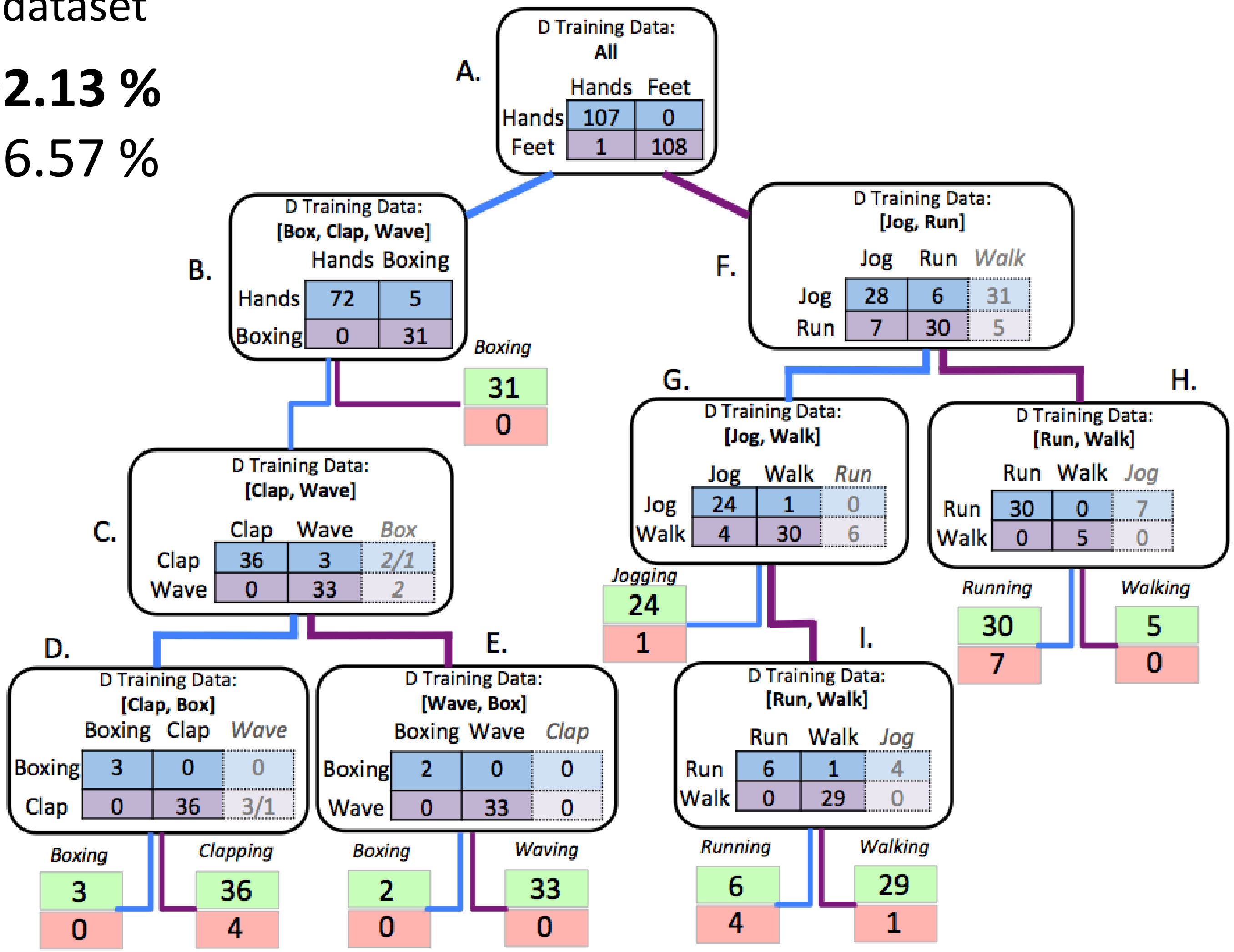
Results reported in area under curve

Applying Sparse Coding Tree to Action Recognition

Tested on KTH dataset

with SC Tree **92.13 %**

without: 86.57 %



Conclusion

Sparse coding, as an effective feature extraction method, can be enhanced by these techniques:

Sparse Coding tree (SC-tree)
to learn features with hierarchy

Non-negative constraints
to mitigate over-fitting in SC

Mirroring
to increase variation tolerance