Columbia University
in the City of New York

IBM-Columbia

Semantic Indexing Task

Noel C. F. Codella
Michele Merler, Leiguang Gong, Liangliang Cao, Matthew Hill, John R. Smith

Contact: nccodell@us.ibm.com
TRECVID 2012

Disclaimer

- Supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center contract number D11PC20070. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright annotation thereon.
- Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/NBC, or the U.S. Government.

IBM

Columbia University
in the City of New York

Overview

- **Background**
  - Review of some existing methods
- **Our approach**
  - Modeling
    - Roughly Balanced Feature Bags
    - Forward Model Selection with Weighting
  - Features
    - Brief Review of Features and Granularities
    - Learning Semantics by Semantics
  - Massively Parallel Hadoop Learning Implementation
- **Results**
- **Conclusions and Future Work**

Background

Two Major Challenges

Background

- Generating semantic classifiers is challenging from multiple perspectives:
- **1)** Data is very large & typically unbalanced.
  - Hundreds of thousands of examples, must scale
- **2)** Concepts are difficult to model
  - Requires broad low-level feature representation to cover various aspects of each concept.

Review of Some Existing Methods

1

SCALE

- **Model Algorithm Based Approaches to Imbalance.**
  - Power SVM [Zhang et. Al. CVPR 2012]
    - Show good performance, but specific to particular types of models.
    - Doesn't really address scale
- **Data Based Approaches to Imbalance & Scale**
  - EasyEnsemble / BalanceCascade [Liu et al. TSMC 2009]
  - SMOTE [Chawla et. al. JAIR 2002]
  - More easily generalized / parallelized.

2

COMPLEXITY

- **Early fusion of a variety of features**
  - Very large dimensionality, slow learning and evaluation.
  - Difficult to select feature combinations
  - Not practical with large numbers of examples.
- **Random Subspace Bagging [Ho, TPAMI 1998]**

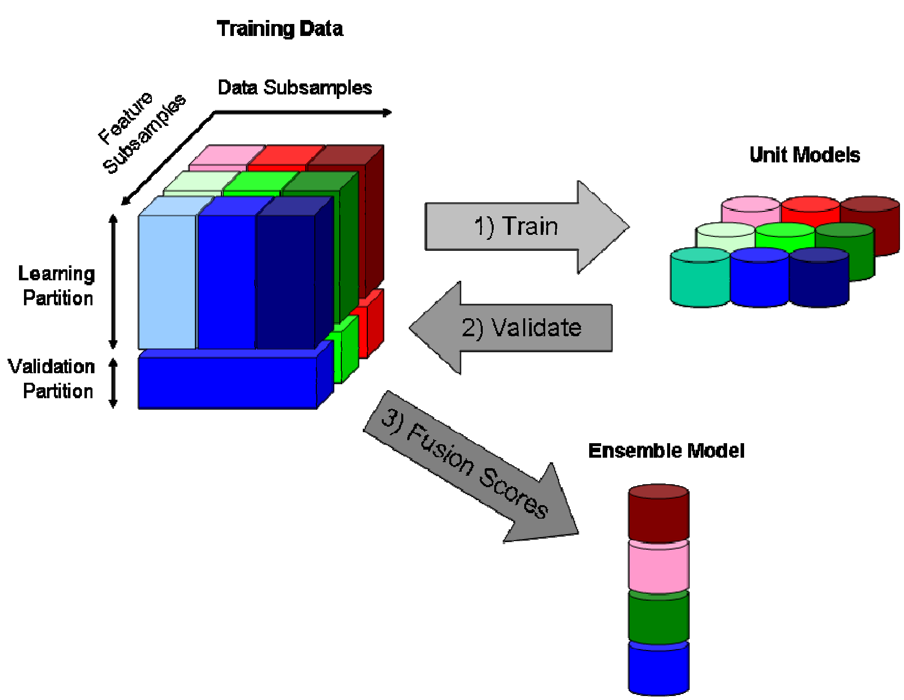
Approach

Modeling

Roughly Balanced Feature Bags

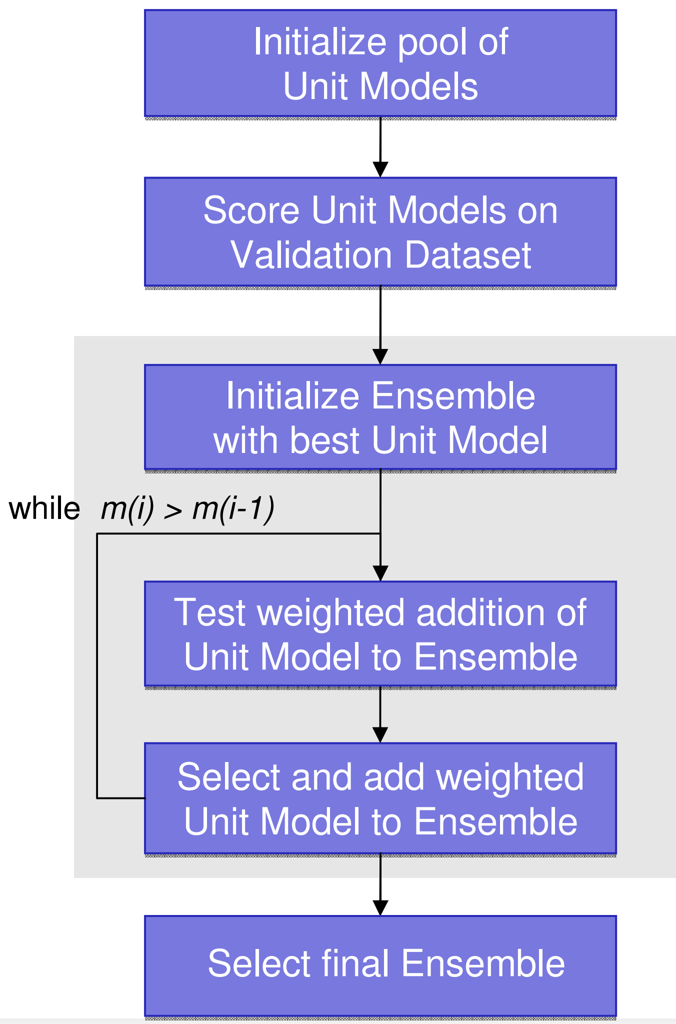
1 2

An approach to address both scalability and feature selection.



- Large scale learning problems pose several challenges:
  - Size
  - Imbalance
  - Broad domain
- RBBag is "Divide and Conquer" Learning Approach via "Bagging"
- Permits parallelization
  - Both Learning & Scoring
- Reduces computational complexity
  - $O(x(n/x)^c) < O(n^c)$
- Balances Data
- Implements Feature Selection

Forward Model Selection



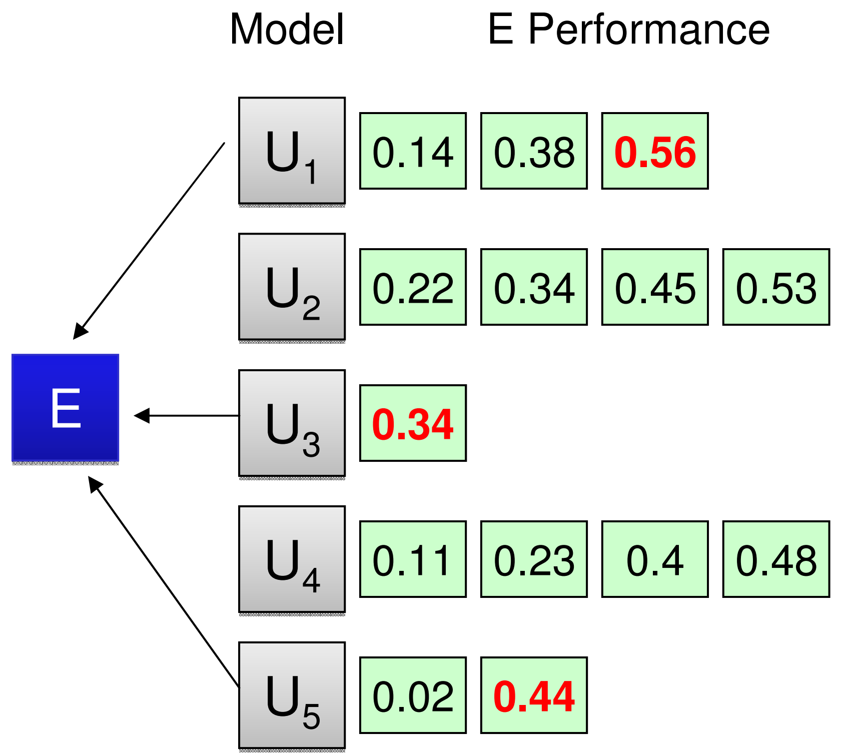
- **Input**
  - Collection of *N* Unit Models U_i each with weight w_i (either 1/N or training cross validation score s_i)
  - Metric m to optimize (AP, Accuracy, Precision, Recall, F1)
- **Step 1** – *scoring*
  - Score Unit Models against Validation Dataset
- **Step 2** – *fusion loop*
  - Initialize Ensemble Model with top performing Unit Model
  - For each Unit Model $M_i(x)$, evaluate performance $m(i)$ when added to Ensemble Model with weight w_i
  - If no Unit Model produces increase in Ensemble $m(i)$ **break** loop
  - Add to Ensemble the Unit Model that maximizes $m(i)$
- **Output**
  - Final Ensemble (weighted combination) of Unit Models

$$E = \sum_{i=1}^{T} w_i U_i$$

Demo: Forward model selection chooses best unit models to boost Ensemble model performance

- Step 1: Select Unit Model U_1 with best performance on a held-out validation dataset, and add to Ensemble Model E, which is initially empty. Weight and judge ensemble performance according to a user selected metric at a specific threshold (AP, Recall, Precision, etc).

- Step 2: Loop: while still models remaining,
  - Select Unit Model U_i, weighted by selected metric, such that $(E + U_i)$ has best performance, in terms of chosen metric m (AP, Precision, Recall, etc).
  - If performance less than previous iteration, **break** from loop.
  - Remove U_i from remaining models,
  - Repeat Step 2



$$E = w_3 U_3 + w_5 U_5 + w_1 U_1$$

w_i = cross-validation training score s_i (AP, or Precision, or Recall, etc.)

or 1/3 (equal weight)

IBM

Columbia University in the City of New York

Approach

Features

Global Visual Features - Spatial Granularities

	Center	Cross	Global	Grid	Horizontal	Horiz. Parts	Layout	Vertical
Color Correlogram	X	X	X		x		X	X
Color Histogram	X	X	X		X	X	X	X
Color Moments	X		X			x		X
Color Wavelet		x	X					
Color Wavelet Texture	X		X		X	x	X	X
Fourier Polar Pyramid	X		X					
Edge Histogram	X		X		X	X	X	x
GIST			X					
Image Stats			X	X				
Image Type	X		x	X	X	x		x
LBP histogram			X					
Maxi Thumbnail Vector			X					
Mini Thumbnail Vector	X		X					
Siftogram			X					
Size Vector			X					
Thumbnail Vector	X		X					
Wavelet Texture	X		X					
Curvelet Texture			x	x				

Local SIFT-based Descriptors

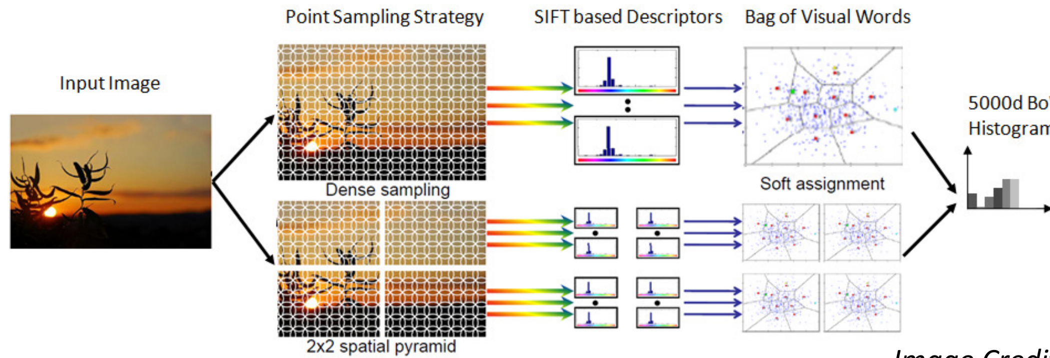


Image Credit: Koen Van De Sande

DETECTOR

- Dense sampling, with offset = 6 pixels

DESCRIPTORS

- SIFT [Lowe 04]: 128 dimensions
- SIFT color variants [Van de Sande et al. PAMI10]:
  - RGB-SIFT : SIFT computed for every RGB channel independently: 128x3 = 384 dimensions
  - HSV-SIFT : SIFT computed for every HSV channel independently: 128x3 = 384 dimensions
- C-SIFT : SIFT computed for every $O_1O_2O_3$ opponent channel. O_1 and O_2 are normalized with the C-invariant to eliminates intensity information: : 128x3 = 384 dimensions

$$\begin{pmatrix} o_1 \\ o_2 \\ o_3 \end{pmatrix} = \begin{pmatrix} \frac{R-G}{\sqrt{2}} \\ \frac{R-G-2B}{\sqrt{6}} \\ \frac{R+G+B}{\sqrt{3}} \end{pmatrix}$$

BOW MODEL + SPATIAL POOLING

- For each descriptor, we computed 2 separate codebooks (vocabularies *V*) with 1000 elements w
  K-means clustering, starting from ~250K descriptors randomly sampled from the Training set
- Soft BoW assignment using codeword uncertainty: $UNC(w) = \frac{1}{n}\sum_{i=1}^{N} \frac{K_\sigma(D(w, r_i))}{\sum_{j=1}^{|V|} K_\sigma(D(v_j, r_i))}$ $D(w, r_i)$ distance between a codeword w and descriptor r_i

  $K_\sigma = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{1}{2}\frac{x^2}{\sigma^2}\right)$ Gaussian shaped kernel with σ=90
- Spatial Pyramid Matching, 1x2x2 (one global level + a 2x2 grid)
- For each descriptor, 2 versions of 5000 dimensions each (one per codebook): 1000x(1+2x2)

[Van Gemert et al. PAMI10] Jan C. van Gemert, , Cor J. Veenman, Arnold W. M. Smeulders and Jan-Mark Geusebroek, **Visual Word Ambiguity**, in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 32 (7), pages 1271-1283, 2010.

[Van De Sande et al. PAMI10] Koen E. A. van de Sande, Theo Gevers and Cees G. M. Snoek, **Evaluating Color Descriptors for Object and Scene Recognition**, in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 32 (9), pages 1582-1596, 2010.

Code available at http://koen.me/research/colordescriptors/

FourierPolarPyramid Feature Vector Description

- Pyramid constructed of Fourier space in polar coordinates.
- Radial levels of 1, 2, and 4 segments.
- Angular levels of 1, 2, 4, 8, and 16 segments.
- 4 color channels: R, G, B, Gray
- Circular pre-filter to improve consistency of rotational effects across domains.

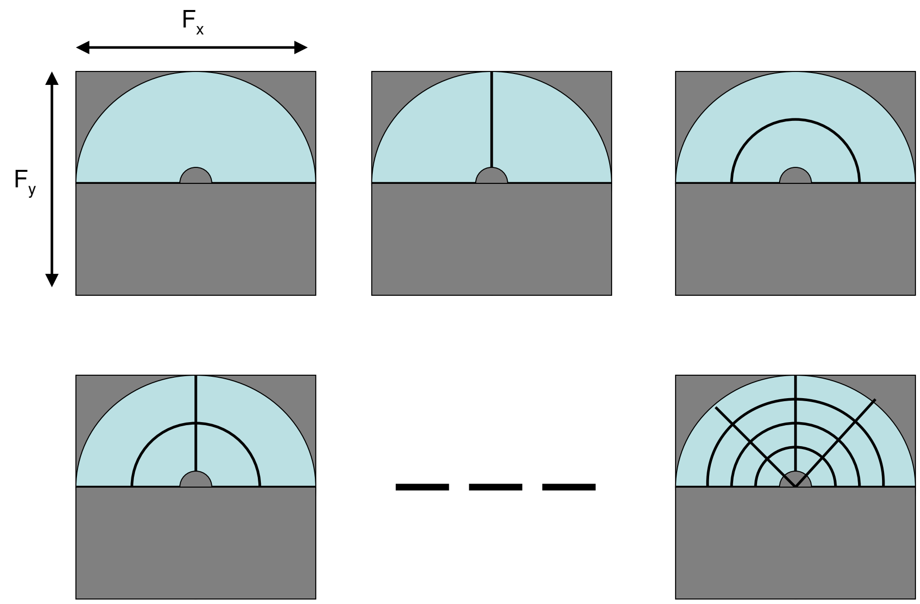
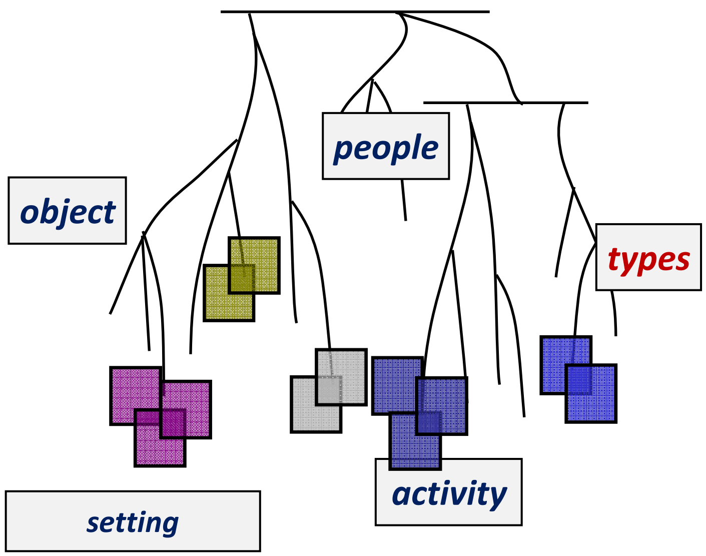


Fig. 1: Segmentation of polar coordinates in Fourier-Mellin space. The average value of each blue region in a single color channel becomes a feature vector element. This it iterated for all regions and all color channels.

Learning Semantics with Semantics



- IBM Semantic Taxonomy
  - Over 600 concepts with training data crawled from web
  - Categories cover objects, settings, activities, etc.
- By extracting semantic information from SIN training data, this can be used as another feature from which to learn the new SIN concepts.

IMARS Visual Semantic Taxonomy

Key

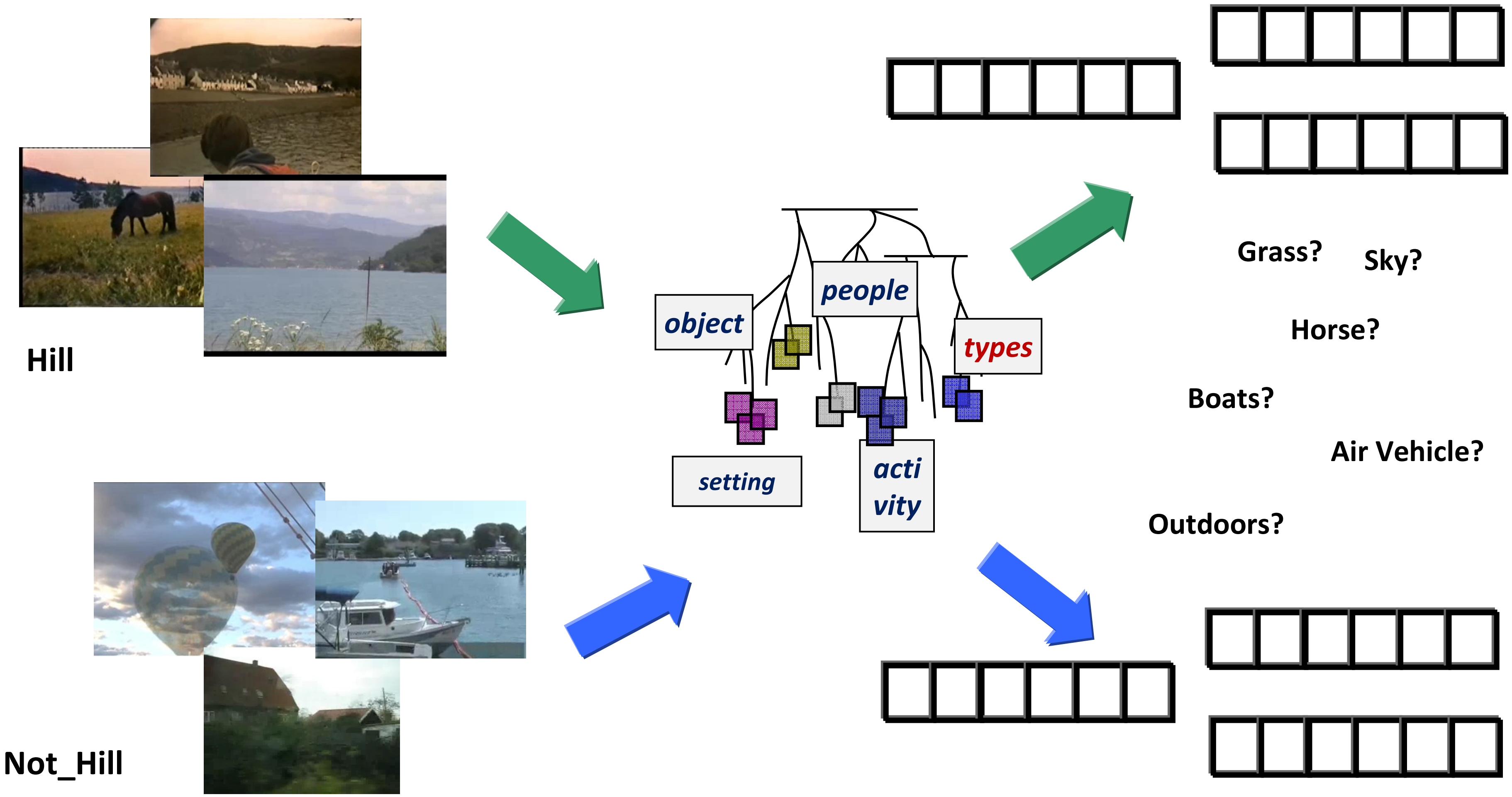
Facet

examples
(data size)
categories

Activity **25.5k** **(4GB)** **29**	**Animal** **62.4k** **(11.5GB)** **85**	**People** **35.8k** **(2.5GB)** **16**	**General Setting** **95.9k** **(26.3GB)** **217**	**Dominant Color** **11.5k** **(1.75GB)** **13**	**Object** **93.2k** **(14.2GB)** **124**
Sports **11.3k** **(2.4GB)** **9**	**Disaster Scene** **9k** **(2.1GB)** **7**	**Sky Scene** **11.3k** **(2.4GB)** **9**	**Building View** **42k** **(5.75GB)** **20**	**People w/Affil.** **22.1k** **(1.8GB)** **16**	**2011:** **300k images** **780 classes** **2012:** **500k images** **630 classes**

- Adding training examples to strengthen classifiers (500K images/630 classes)
- Designing **attributes** that further express properties and relationships among concepts

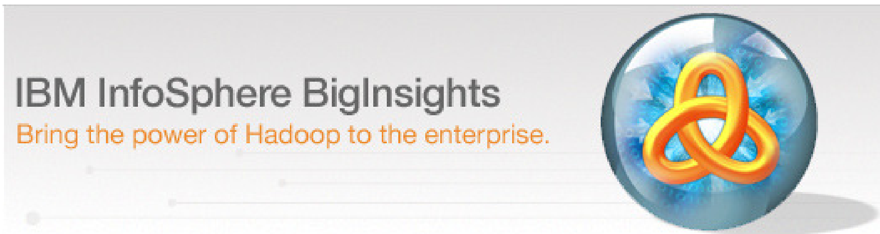
Semantic Model Vectors as Feature Representation



Approach

Implementation

IMARS Hadoop Implementation



- **Proprietary distributed implementation of IMARS**.
  - Not reliant on any previous package machine learning environment for Hadoop; i.e. Apache Mahoot.

- **Large-scale feature extraction, classifier training, and image scoring.**
  - Ability to add concepts, examples, and features, without throughput concerns.

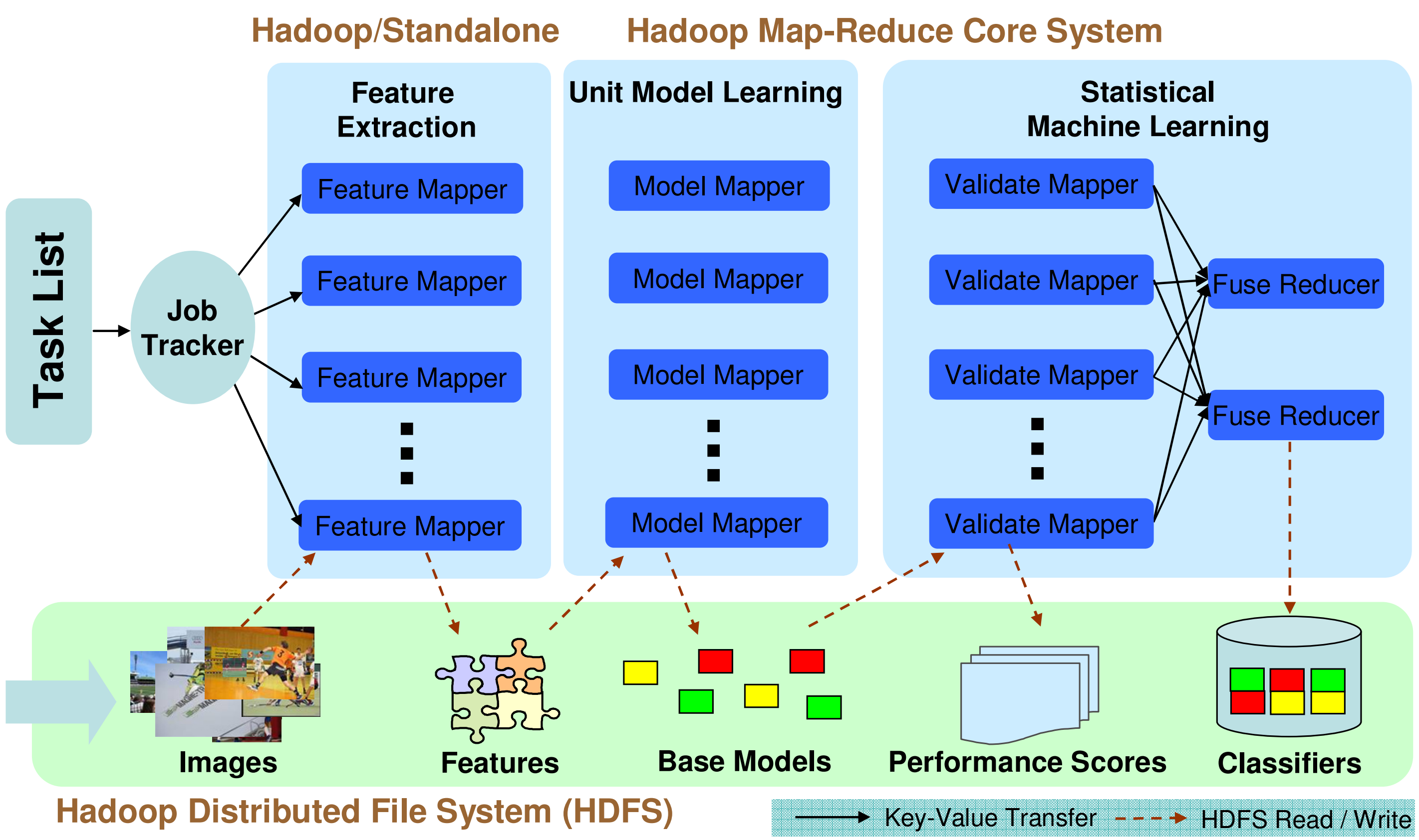
- **Feature Extraction:**
  - Map Step: parallelized over concepts

- **Ensemble Model Learning**:
  - Mapper: parallelized SVM training over bags of feature types, granularity, randomly sampled data during unit model training Map job.
  - Reducer: parallelized over concept classifiers during model fusion

- **Image scoring:**
  - Mapper: unit model evaluation on image features
  - Reducer: sum of unit models into fusion models

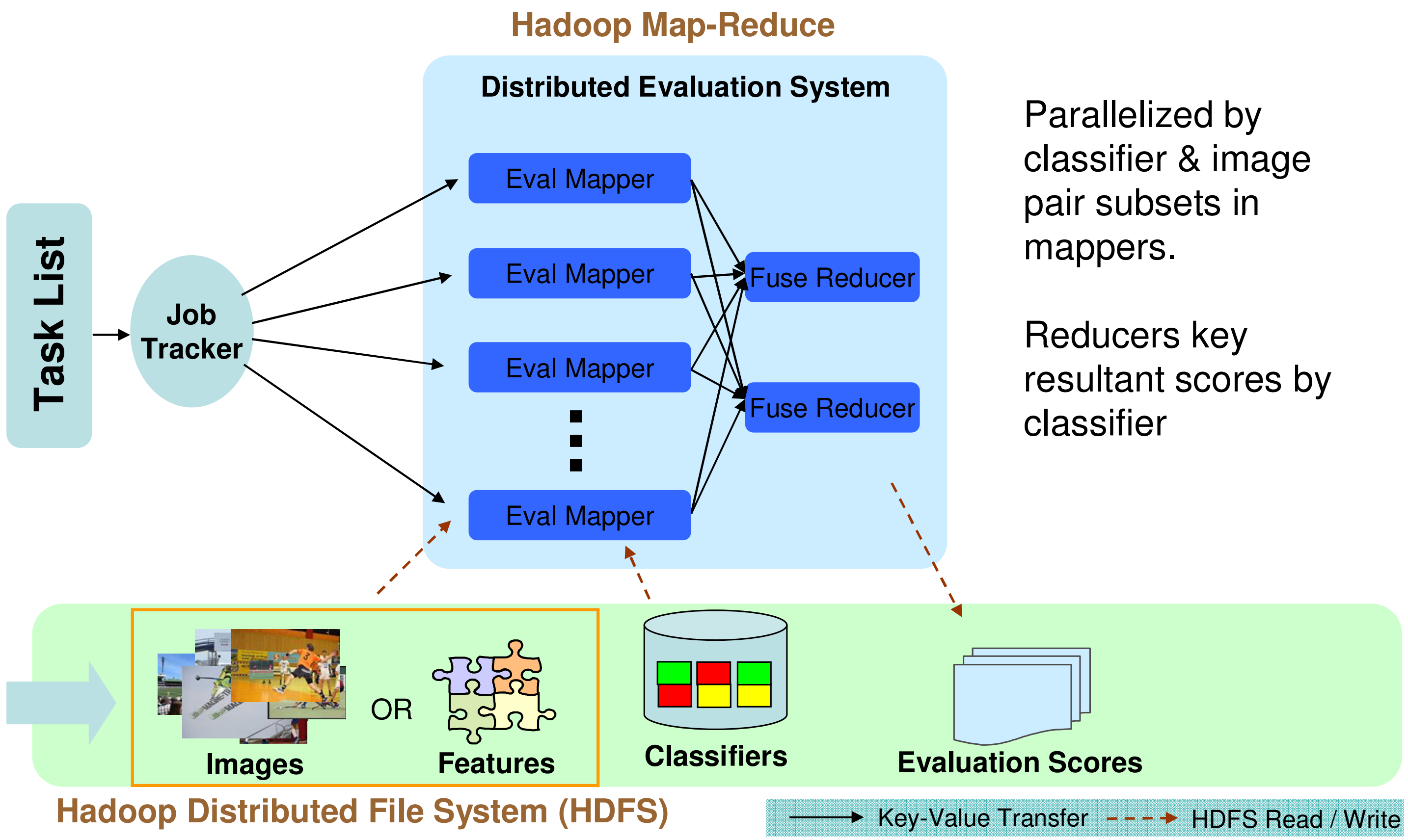
IMARS Hadoop Implementation

Overall Architecture of Hadoop Distributed Learning System



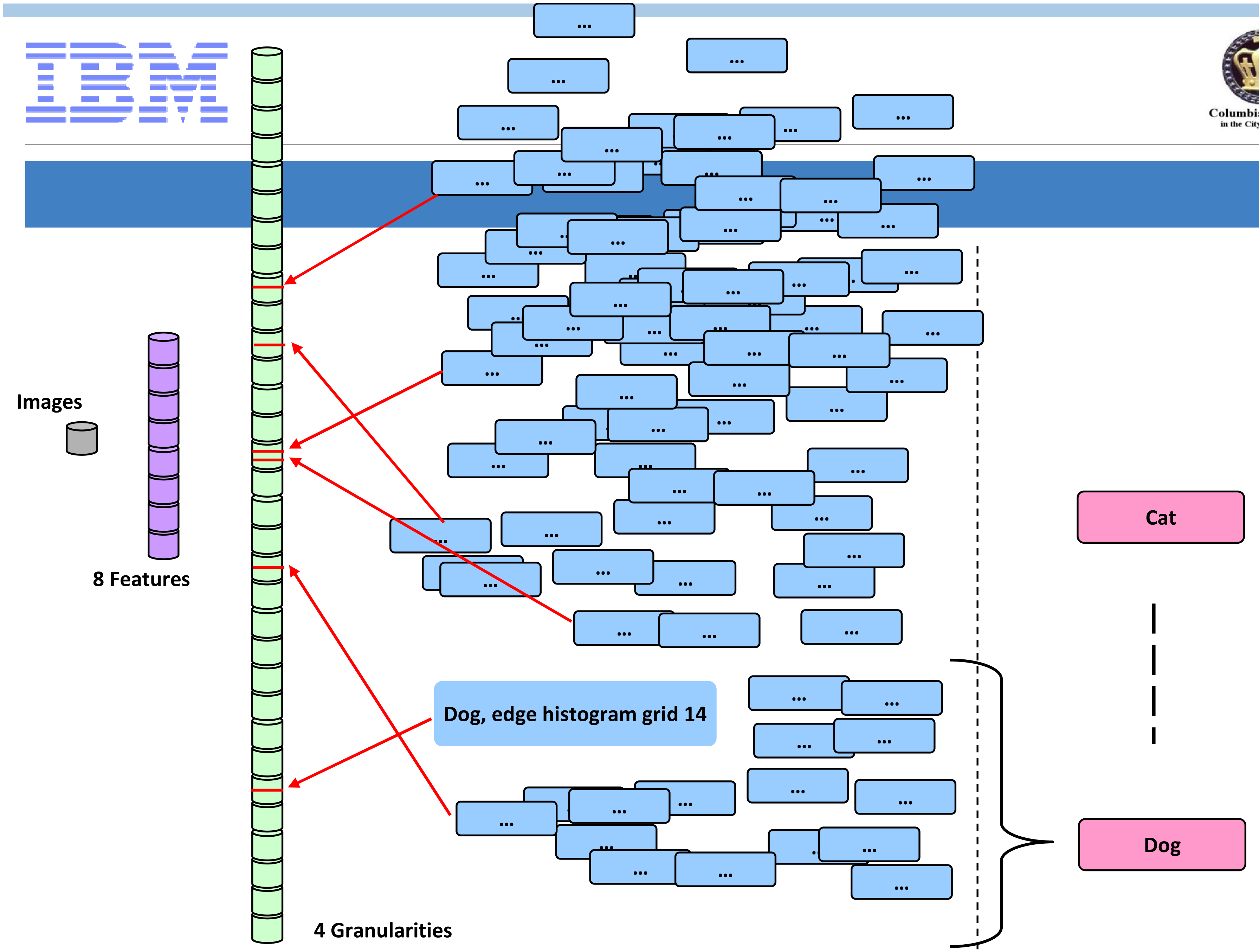
IMARS Hadoop Implementation

Overall Architecture of Hadoop Distributed Evaluation System



Parallelized by classifier & image pair subsets in mappers.

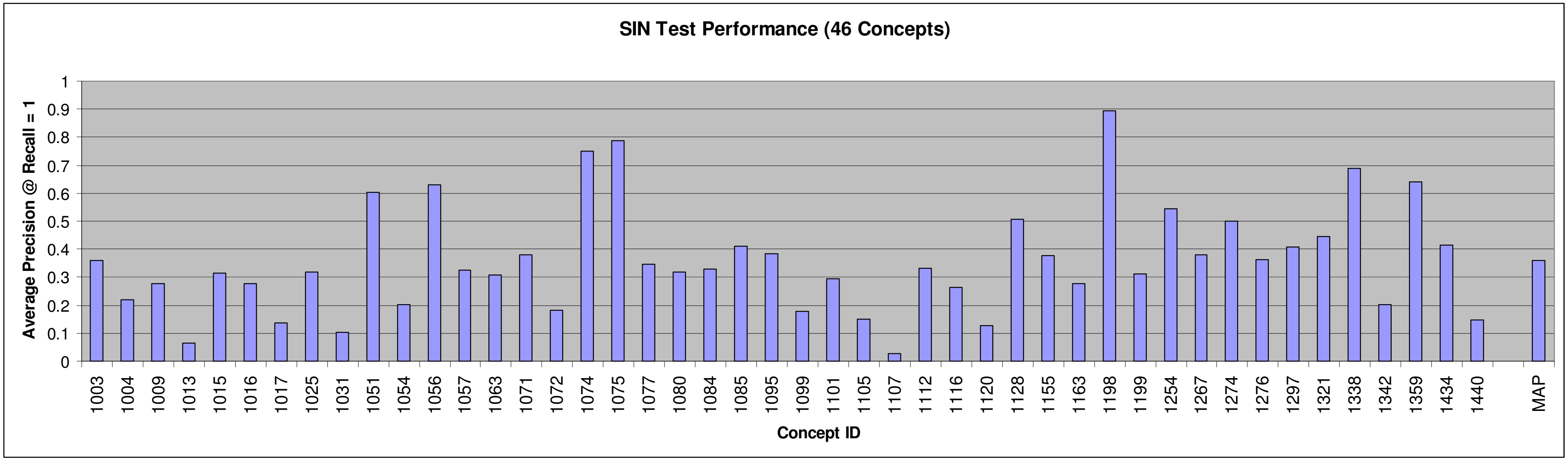
Reducers key resultant scores by classifier


Images
8 Features
4 Granularities
Dog, edge histogram grid 14
Cat
Dog

Results

Results

Sampling: Up to 3000 positive examples, and 3000 negative examples per category. Including BRNO annotations.



Official submission compromised by incorrect ground truth data and scoring system bug.

MAP = 0.3588

Test set re-scored and compared to ground truth **once** to avoid possibility of overfitting.

Best SIN Run: 0.321
Our Previous: 0.14

Top 20 Feature MAP Performance

MAP evaluated from a single feature "bag" performance on 20% validation data across all 46 categories

FEATURE TYPE	MAP	FEATURE TYPE	MAP
IBM_ModelVector.sig	0.71782	lbp_histogram_grid	0.65906
IBM_ModelVector	0.71194	color_correlogram_horizontalparts	0.65873
csiftVLFEAT	0.69013	color_correlogram_layout	0.65795
csift	0.68636	lbp_histogram_grid7	0.65791
siftVLFEAT	0.67694	lbp_histogram_layout	0.65678
rgbsiftVLFEAT	0.66576	color_correlogram_grid	0.65502
rgbsift	0.66431	lbp_histogram_horizontalparts	0.64486
hsvsiftVLFEAT	0.66371	gist_layout	0.63726
sift	0.66284	color_correlogram_cross	0.63417
hsvsift	0.661	color_histogram_grid	0.63341

Future Directions

- Current Ensemble Model Fusion strategies are a **retrospective** approach:
  - Unit Models learning decoupled from ensemble fusion
  - Once all unit models are trained, go select which combination is the best

- A **prospective** approach intricately tied with Unit Model training would be preferred and has potential to improve performance
  - **Method 1:** Train one batch of unit models for all features and granularities on a subsample of data. For the next round of unit model training, use not the ground truth labels, but the residual (incorrect classifications) from the first round of models, or other similar methods to reduce error correlation between unit models. [Levy, Wolf, ECCV 2012]
  - **Method 2:** Train one batch of unit models for all features and granularities on a subsample of data. For the next round of unit model training, use only data misclassified by first round of models (but use original ground truth labels). [Khoshgoftaar et. al., IEEE TSMC 2011]

- Potential downsides:
  - May be more difficult to parallelize in a computationally balanced fashion.

Thank you!

- Questions?