BIGS: A Framework for Large-Scale Image Processing and Analysis Over Distributed and Heterogeneous Computing Resources

Raúl Ramos-Pollán, Fabio González, Juan C. Caicedo, Angel Cruz-Roa, Jorge E. Camargo, Jorge A. Vanegas, Santiago A. Pérez, Jose D. Bermeo, Juan S. Otálora, Paola K. Rozo, John E. Arévalo

Bioingenium Research Group
Universidad Nacional de Colombia
Motivation
Motivation

• State of the art computer vision algorithms may be computationally expensive
Size of Image Databases

1 icon = 2,000 images

2,600  25,000  230,000

Less computing Resources  More computing Resources
Feature Extraction Pipelines

INPUT IMAGES -> PATCH EXTRACTION -> FEATURES EXTRACTION -> FEATURES CLUSTERING

- BOF representation normalized by L1 norm
- Codebook k=700
- Regular grid extraction
- Block size 8x8 pixels
- DCT
- Block descriptor d=192
- k-means
Motivation

• How to take advantage of seldom availability of computing resources?
Strategy

• Build on experiences with Hadoop / Grid, etc.

• Profit from whatever resources available
• Decouple algorithm design from deployment
• Adopt Big Data principles and technologies
• Streamline software development process
• Unify coherent algorithms repository
BIGS Framework

EXPERIMENTER

PIPELINE DEFINITION
STAGE 1 → STAGE 2
STAGE 4 ← STAGE 3

SCHEDULE GENERATOR
CLIENT at EXPERIMENTER’S DESKTOP

INPUT OUTPUT DATA
SCHEDULE
NOSQL DATABASE

WORKERS
WORKER
WORKER
WORKER
(AMAZON, DESKTOPs)

WORKER
ADDED LATER
Pipeline definition example

#######################################
# FIRST STAGE: Patch Sampling
#######################################
stage.01.task: ROIsFeatureExtractionTask
stage.01.numberOfPartitions: 10
stage.01.roiAlgorithm: RandomPatchExtractor
stage.01.feAlgorithm: GrayHistogram

stage.01.RandomPatchExtractor.blockSize: 18
stage.01.RandomPatchExtractor.Size: 256

stage.01.input.table: corel5k.imgs
stage.01.output.table: corel5k.sample

#######################################
# SECOND STAGE: CodeBook Construction
#######################################
stage.02.task: KMeans

stage.02.KMeans.kValue: 10
stage.02.KMeans.maxNumberOfIterations: 5
stage.02.KMeans.minChangePercentage: 0.1
stage.02.KMeans.numberOfPartitions: 10

stage.02.input.table: corel5k.sample
stage.02.output.table: corel5k.codebook

#######################################
# THIRD STAGE: Bag of Features Histograms
#######################################
stage.03.task: BagOfFeaturesExtractionTask

stage.03.numberOfPartitions: 10
stage.03.maxImageSize: 256
stage.03.minImageSize: 20
stage.03.indexCodeBook: true
stage.03.featureExtractor: GrayHistogram
stage.03.patchExtractor: RegularGridPatchExtractor
stage.03.spatialLayoutExtractor: IdentityROIExtractor
stage.03.codeBookID: STAGE-OUTPUT 2

stage.03.RegularGridPatchExtractor.blockSize: 18
stage.03.RegularGridPatchExtractor.stepSize: 9

stage.03.input.table: corel5k.imgs
stage.03.output.table: corel5k.histograms
Framework architecture

BIGS: A Framework for Large-Scale Image Processing

ичество

KMEANS

ITERATIVE

DATA PARTITION

CORE

schedule generator
data management
jobs wrapper

SCHEDULES

DYNAMO DB IMPL

HBASE IMPL

FILE SYSTEM IMPL

STORAGE API

TASKS API

ITERATIVE

DATA PARTITION

KMEANS

BAG of FEATURES

SCHEDULES

INPUT/OUTPUT DATASETS

UNDERLYING NoSQL STORAGE

WEB CONSOLE

CMD LINE TOOL

WORKER

AMZON VM LAUNCHER

JWS LAUNCHER

CMD LINE LAUNCHER

WORKER API

CLIENT API

EXPERIMENTER

WEB CONSOLE

CMD LINE TOOL

WORKER

AMZON VM LAUNCHER

JWS LAUNCHER

CMD LINE LAUNCHER

WORKER API

CLIENT API

EXPERIMENTER
Algorithms patterns

DATA PARTITION
ITERATIVE ALGORITHMS
ONLINE ALGORITHMS

BEFORE PROCESSING ALL PARTITIONS

PARTITION 1
START PARTITION
PROCESS DATA ITEM
FINALIZE PARTITION
AFTER PROCESSING ALL PARTITIONS

SAME PROCESS

PARTITION 2
START PARTITION
PROCESS DATA ITEM
FINALIZE PARTITION
AFTER PROCESSING ALL PARTITIONS

SAME PROCESS

PARTITION N
START PARTITION
PROCESS DATA ITEM
FINALIZE PARTITION
AFTER PROCESSING ALL PARTITIONS

SAME PROCESS

START ITERATION

BEFORE PROCESSING ALL PARTITIONS

PARTITION 1
PARTITION 2
PARTITION N
AFTER PROCESSING ALL PARTITIONS
FINALIZE ITERATION

INPUT DATA PARTITION
PROCESS DATA ITEM
OUTPUT DATA PARTITION

eScience, Chicago 2012
Experimental Evaluation

- ImageCLEFmed 2012 Database
- Split in **three** subsets:
  - 3K: 1%
  - 30k: 10%
  - 300k: 100%
First Experiment

- University Computer Room for Students

- 30K dataset
- Java Web Start
- Up to 40 Workers
Results

• Elapsed and compute times (min)

![Graph showing elapsed and compute times](image)
Second Experiment

- Amazon EC2

- 3K dataset
- EC2 VM
- Up to 20 Workers
Cloud usage

# prepare files with AWS credentials and desired pipeline
> load.imgs images /home/me/clef/images
306,530 files loaded into table images
> pipeline.load bof.pipeline
pipeline successfully loaded. pipeline number is 4
> pipeline.prepare 4
pipeline generated 56 jobs
> aws.launchvms ami-30495 15
launching 15 instances of ami-30495
> pipeline.info 4
.....
> aws.termvms ami-30495 5
terminating 5 instances of ami-30495
> pipeline.prepare 4
Results

- Elapsed and compute times

![Graphs comparing elapsed and compute times for 3K Amazon Hadoop and 3K Amazon BIGS](image-url)
Results

• Empirical and theoretical speedup
Third Experiment

- Servers and desktops at our lab

- 300K dataset
- Command line
- 40 Workers
Results

- Times required to process the 300k dataset

<table>
<thead>
<tr>
<th>Measure</th>
<th>Bag of Features</th>
<th>Spatial Pyramid CEDD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workers</td>
<td>36</td>
<td>40</td>
</tr>
<tr>
<td>Elapsed Time</td>
<td>116.05 min</td>
<td>36.67 min</td>
</tr>
<tr>
<td>Total Compute Time</td>
<td>39.59 hours</td>
<td>110.32 min</td>
</tr>
<tr>
<td>Speedup</td>
<td>20.5X</td>
<td>3.0X</td>
</tr>
<tr>
<td>Avg Compute Time per Worker</td>
<td>65.98 min. +/- 18.7</td>
<td>2.76 min. +/- 0.93</td>
</tr>
</tbody>
</table>
Results

- Adhoc image retrieval task
  - 1st out of 53 experiments (textual modality)
  - 3rd out of 36 experiments (visual modality)
- Label: UNAL

http://www.imageclef.org/2012/medical
Conclusions

• Tool for supporting the full image processing lifecycle
• First scalability behavior on Amazon Cloud
• Decoupled algorithm design from deployment
• Adapted to our reality \(\rightarrow\) grasp whatever computer resources available
Conclusions

• Streamlined internal software process
• Unified software repository

• AGILE EXPERIMENTATION LIFECYCLE
• LITTLE PRIOR KNOWLEDGE ON COMPUTING RESOURCES AVAILABLE
Future work

• Extend support to additional machine learning processes
• Larger scalability experiments (# workers)
• Optimize Amazon usage (DynamoDB)
• Better understand Amazon costs patterns
• Experience with NoSQL scalability (HBase)

• Better understand limitations of each deployment model (opportunistic, cluster, cloud)
• Move to >1M image datasets
BIGS: A Framework for Large-Scale Image Processing and Analysis Over Distributed and Heterogeneous Computing Resources

Raúl Ramos-Pollán, Fabio González, Juan C. Caicedo, Angel Cruz-Roa, Jorge E. Camargo, Jorge A. Vanegas, Santiago A. Pérez, Jose D. Bermeo, Juan S. Otálora, Paola K. Rozo, John E. Arévalo

Bioingenium Research Group
Universidad Nacional de Colombia