A Framework for High Performance Image Analysis Pipelines over Cloud Resources

Raúl Ramos-Pollán, Ángel Cruz-Roa, Fabio González-Osorio
Bioingenium Research Group
Universidad Nacional de Colombia
Motivation

• Bioingenium research group
  www.bioingenium.unal.edu.co

• Large scale image processing pipelines
• Initially focused on medical imaging
• Support for machine learning processes
• Seldom availability of computing resources
• Limitation on collections size, applicability, algorithms design
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
Image Processing Pipelines

- **INPUT IMAGES**
- **PATCH EXTRACTION**
- **FEATURES EXTRACTION**
- **FEATURES CLUSTERING**
- **LATENT SEMANTICS**
- **AUTOMATIC ANNOTATION**

- **BOF representation Normalized by L1 norm**
- **Regular grid extraction**
- **Codebook k=700**
- **Block size 8x8 pixels**
- **DCT**
- **k-means**
- **Block descriptor d=192**
Image Processing Framework
Pipeline definition example

# FIRST STAGE: Patch Sampling
stage.01.task: ROIsFeatureExtractionTask
stage.01.roiAlgorithm: RandomPatchExtractor
stage.01.numberOfPartitions: 10
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
Algorithms patterns

DATA PARTITION
ITERATIVE ALGORITHMS
ONLINE ALGORITHMS
Goals for Experiments on the Cloud

• After using our framework in an opportunistic setting

• Understand how to deploy our framework on Amazon Cloud Services

• Validate transparency for the experimenter

• Understand cost patterns
Cloud deployment

A Framework for High Performance Image Analysis Over Cloud Resources
Cloud usage

```bash
# 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
```
Experiments setup

<table>
<thead>
<tr>
<th></th>
<th>100 imgs (1 to 10 workers)</th>
<th>1000 imgs (1 to 20 workers)</th>
</tr>
</thead>
<tbody>
<tr>
<td>fe-simple</td>
<td>20 splits</td>
<td>50 and 200 splits</td>
</tr>
<tr>
<td>fe-complex</td>
<td>20 splits</td>
<td>50 splits</td>
</tr>
</tbody>
</table>

Table I
Experimental scenarios

simple experiments before committing budget for Amazon 500KB images

Amazon EC2 t1.micro instances (613MB 1 virtual core)
fe-simple  simulated image feature extraction 0-2 secs
fe-complex  simulated image feature extraction 5-8 secs
Experiments result 100 imgs
Experiments results 1000 imgs

FE SIMPLE

FE COMPLEX
Experiments scalability

1000 imgs

<table>
<thead>
<tr>
<th>AWS Service</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC2 (workers)</td>
<td>9.28 USD</td>
</tr>
<tr>
<td>DynamoDB (tables)</td>
<td>11.05 USD</td>
</tr>
<tr>
<td>S3 (images)</td>
<td>0.23 USD</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td><strong>20.56 USD</strong></td>
</tr>
</tbody>
</table>
Additional experiments

Images from ImageCLEF2012 medical challenge.

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

306,530 medical images
from journal papers in biomedical sciences
x-rays, MRI, PET, diagrams, ultrasound, microscopy, gammagraphy, ...
mostly in JPEG format at medium resolution (avg width 512 pixels).
Additional experiments

CEDD EXTRACTION OF 3K ImageCLEF Medical

3 STAGE PIPELINE ON 30K ImageCLEF Medical

1 Patch extraction
2 Kmeans – codebook
3 BoF Representation
Additional experiments

• Submission to ImageCLEF2012 medical
• All 300K images processed and annotated
• 36 workers over 6 scattered servers
• One run (parameters configuration)
  • 116 mins elapsed time
  • 40 hours compute time

• Ad-hoc image retrieval task:
  • 1st out of 53 in textual classification
  • 3rd out of 36 in visual classification

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 → grasp whatever computer resources available
• Streamlined internal software process
• Unified software repository

• AGILE EXPERIMENTATION LIFECYCLE
• LITTLE A PRIORI 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