Big Data
Selected Computational Intelligence approaches

Francisco Herrera
Research Group on Soft Computing and Information Intelligent Systems (SCI²S)
Dept. of Computer Science and A.I.
University of Granada, Spain

Email: herrera@decsai.ugr.es
http://sci2s.ugr.es
Outline

- What Big Data?
- MapReduce Paradigm
- Hadoop and Mahout
- Computational Intelligence Approaches for Big Data
- Final Comments
Outline

- What Big Data?
- MapReduce Paradigm
- Hadoop and Mahout
- Computational Intelligence Approaches for Big Data
- Final Comments
What is Big Data?

Our world revolves around the data

- **Science**
  - Data bases from astronomy, genomics, environmental data, transportation data, ...

- **Humanities and Social Sciences**
  - Scanned books, historical documents, social interactions data, ...

- **Business & Commerce**
  - Corporate sales, stock market transactions, census, airline traffic, ...

- **Entertainment**
  - Internet images, Hollywood movies, MP3 files, ...

- **Medicine**
  - MRI & CT scans, patient records, ...

- **Industry, Energy, ...**
  - Sensors, ...
No single standard definition

**Big data** is a collection of data sets so large and complex that it becomes difficult to process using on-hand database management tools or traditional data processing applications.

“**Big Data**” is data whose scale, diversity, and complexity require new architecture, techniques, algorithms, and analytics to manage it and extract value and hidden knowledge from it...
What is Big Data? 3 Vs of Big Data

Big Data = Transactions + Interactions + Observations

Source: Contents of above graphic created in partnership with Teradata, Inc.
What is Big Data? 3 Vs of Big Data

Some Make it 4V’s

<table>
<thead>
<tr>
<th>Volume</th>
<th>Velocity</th>
<th>Variety</th>
<th>Veracity*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data at Rest</td>
<td>Data in Motion</td>
<td>Data in Many Forms</td>
<td>Data in Doubt</td>
</tr>
<tr>
<td>Terabytes to exabytes of existing data to process</td>
<td>Streaming data, milliseconds to seconds to respond</td>
<td>Structured, unstructured, text, multimedia</td>
<td>Uncertainty due to data inconsistency &amp; incompleteness, ambiguities, latency, deception, model approximations</td>
</tr>
</tbody>
</table>
What is Big Data?

5 V’s --> Value

Innovative new approaches and technologies

MapReduce

Hadoop

Mahout

Spark

Insight and Knowledge
What is Big Data?

Who’s Generating Big Data? Applications

Social media and networks (all of us are generating data)

Scientific instruments (collecting all sorts of data)

Mobile devices (tracking all objects all the time)

Sensor technology and networks (measuring all kinds of data)

- The progress and innovation is no longer hindered by the ability to collect data but, by the ability to manage, analyze, summarize, visualize, and discover knowledge from the collected data in a timely manner and in a scalable fashion.
What is Big Data?

Who’s Generating Big Data? Applications

Astronomy
- Astronomical sky surveys
- 120 Gigabytes/week
- 6.5 Terabytes/year

Genomics
- 25,000 genes in human genome
- 3 billion bases
- 3 Gigabytes of genetic data

Phone call billing records
- 250M calls/day
- 60G calls/year
- 40 bytes/call
- 2.5 Terabytes/year

Credit card transactions
- 47.5 billion transactions in 2005 worldwide
- 115 Terabytes of data transmitted to VisaNet data processing center in 2004

Internet traffic
- Traffic in a typical router:
  - 42 kB/second
  - 3.5 Gigabytes/day
  - 1.3 Terabytes/year

The World-Wide Web
- Google
  - 25 billion pages indexed
  - 10kB/Page
  - 250 Terabytes of indexed text data
  - "Deep web" is supposedly 100 times as large
What is Big Data?  Example

Evolutionary Computation for Big Data and Big Learning Workshop

Data Mining Competition 2014: Self-deployment track

Objective: Contact map prediction

Details:
- 32 million instances
- 631 attributes (539 real & 92 nominal values)
- 2 classes
- 98% of negative examples
- About 56.7GB of disk space

Evaluation:
True positive rate • True negative rate
TPR • TNR

http://cruncher.ncl.ac.uk/bdcomp/index.pl?action=data
Outline

- What Big Data?
- MapReduce Paradigm
- Hadoop and Mahout
- Computational Intelligence Approaches for Big Data
- Final Comments
MapReduce

- Scalability to large data volumes:
  - Scan 100 TB on 1 node @ 50 MB/sec = 23 days
  - Scan on 1000-node cluster = 33 minutes

⇒ Divide-And-Conquer (i.e., data partitioning)

A single machine cannot manage large volumes of data efficiently
MapReduce

- Scalability to large data volumes:
  - Scan 100 TB on 1 node @ 50 MB/sec = 23 days
  - Scan on 1000-node cluster = 33 minutes
  ➔ Divide-And-Conquer (i.e., data partitioning)

MapReduce

- Overview:
  - Data-parallel programming model
  - An associated parallel and distributed implementation for commodity clusters

- Pioneered by Google
  - Processes 20 PB of data per day

- Popularized by open-source Hadoop project
  - Used by Yahoo!, Facebook, Amazon, and the list is growing ...
MapReduce

- MapReduce is a popular approach to deal with Big Data
- Based on a **key-value pair** data structure
- Two key operations:
  1. **Map function**: Process independent data blocks and outputs summary information
  2. **Reduce function**: Further process previous independent results

MapReduce

MapReduce data flow

map \((k,v) \rightarrow \text{list}(k',v')\)

reduce \((k',\text{list}(v')) \rightarrow v''\)

Completely transparent to the programmer/user:
Handling machine failures
Managing inter-machine communication
MapReduce

Experience

- **Runs on large commodity clusters:**
  - 1000s to 10,000s of machines
- **Processes many terabytes of data**
- **Easy to use since run-time complexity hidden from the users**
- **Cost-efficiency:**
  - Commodity nodes (cheap, but unreliable)
  - Commodity network
  - Automatic fault-tolerance (fewer administrators)
  - Easy to use (fewer programmers)
MapReduce

The key of a MapReduce data partitioning approach is usually on the reduce phase.

MapReduce: Workflow
Advantage: MapReduce’s data-parallel programming model hides complexity of distribution and fault tolerance

Key philosophy:
- *Make it scale*, so you can throw hardware at problems
- *Make it cheap*, saving hardware, programmer and administration costs (but requiring fault tolerance)

MapReduce is not suitable for all problems, but when it works, it may save you a lot of time
The following malfunctions types of algorithms are examples where MapReduce:

Iterative Graph Algorithms. 
Gradient Descent. 
Expectation Maximization
Outline

- What Big Data?
- MapReduce Paradigm
- Hadoop and Mahout
- Computational Intelligence Approaches for Big Data
- Final Comments
Hadoop

Hadoop is an open source implementation of MapReduce computational paradigm

http://hadoop.apache.org/
Apache Hadoop is an open-source software framework that supports data-intensive distributed applications, licensed under the Apache v2 license.

Hadoop implements the computational paradigm named MapReduce.

Created by Doug Cutting (chairman of board of directors of the Apache Software Foundation, 2010)

http://hadoop.apache.org/
July 2008 - Hadoop Wins Terabyte Sort Benchmark
One of Yahoo's Hadoop clusters sorted 1 terabyte of data in 209 seconds, which beat the previous record of 297 seconds in the annual general purpose (Daytona) terabyte short benchmark. This is the first time that either a Java or an open source program has won.

What Is Apache Hadoop?

The Apache™ Hadoop® project develops open-source software for reliable, scalable, distributed computing.

http://hadoop.apache.org/
The project includes these modules:

- **Hadoop Common**: The common utilities that support the other Hadoop modules.
- **Hadoop Distributed File System (HDFS)**: A distributed file system that provides high-throughput access to application data.
- **Hadoop YARN**: A framework for job scheduling and cluster resource management.
- **Hadoop MapReduce**: A YARN-based system for parallel processing of large data sets.

Other Hadoop-related projects at Apache include:

- **Avro**™: A data serialization system.
- **Cassandra**™: A scalable multi-master database with no single points of failure.
- **Chukwa**™: A data collection system for managing large distributed systems.
- **HBase**™: A scalable, distributed database that supports structured data storage for large tables.
- **Hive**™: A data warehouse infrastructure that provides data summarization and ad hoc querying.
- **Mahout**™: A Scalable machine learning and data mining library.
- **Pig**™: A high-level data-flow language and execution framework for parallel computation.
- **ZooKeeper**™: A high-performance coordination service for distributed applications.

Recently: Apache Spark

http://hadoop.apache.org/
How do I access to a Hadoop platform?

Cloud Platform with Hadoop installation

Amazon Elastic Compute Cloud (Amazon EC2)
http://aws.amazon.com/es/ec2/

Windows Azure
http://www.windowsazure.com/

Cluster ATLAS: 4 super servers from Super Micro Computer Inc. (4 nodes per server)
The features of each node are:

- Microprocessors: 2 x Intel Xeon E5-2620 (6 cores/12 threads, 2 GHz, 15 MB Cache)
- RAM 64 GB DDR3 ECC 1600MHz, Registered
- 1 HDD SATA 1TB, 3Gb/s; (system)
- 1 HDD SATA 2TB, 3Gb/s; (distributed file system)
Mahout

Scalable machine learning and data mining

Apache Mahout has implementations of a wide range of machine learning and data mining algorithms: clustering, classification, collaborative filtering and frequent pattern mining

Mahout currently has
- Collaborative Filtering
- User and Item based recommenders
- K-Means, Fuzzy K-Means clustering
- Mean Shift clustering
- Dirichlet process clustering
- Latent Dirichlet Allocation
- Singular value decomposition

- Parallel Frequent Pattern mining
- Complementary Naive Bayes classifier
- Random forest decision tree based classifier
- High performance java collections (previously colt collections)
- A vibrant community
- and many more cool stuff to come by this summer thanks to Google summer of code

http://mahout.apache.org/
Mahout

Scalable machine learning and data mining

Apache Mahout has implementations of a wide range of machine learning and data mining algorithms: clustering, classification, collaborative filtering, and frequent pattern mining.

Four great application areas

Clustering

Recommendation Systems

Classification

Association
Case of Study: Random Forest for KddCup’99
The RF Mahout Partial implementation: is an algorithm that builds multiple trees for different portions of the data. Two phases:

**Building phase**

**Classification phase**
**Case of Study: Random Forest for KddCup’99**

<table>
<thead>
<tr>
<th>Class</th>
<th>Instance Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>normal</td>
<td>972.781</td>
</tr>
<tr>
<td>DOS</td>
<td>3.883.370</td>
</tr>
<tr>
<td>PRB</td>
<td>41.102</td>
</tr>
<tr>
<td>R2L</td>
<td>1.126</td>
</tr>
<tr>
<td>U2R</td>
<td>52</td>
</tr>
</tbody>
</table>

**Time elapsed (seconds) for sequential versions:**

<table>
<thead>
<tr>
<th>Datasets</th>
<th>10%</th>
<th>50%</th>
<th>full</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOS_vs_normal</td>
<td>6344.42</td>
<td>49134.78</td>
<td>NC</td>
</tr>
<tr>
<td>DOS_vs_PRB</td>
<td>4825.48</td>
<td>28819.03</td>
<td>NC</td>
</tr>
<tr>
<td>DOS_vs_R2L</td>
<td>4454.58</td>
<td>28073.79</td>
<td>NC</td>
</tr>
<tr>
<td>DOS_vs_U2R</td>
<td>3848.97</td>
<td>24774.03</td>
<td>NC</td>
</tr>
<tr>
<td>normal_vs_PRB</td>
<td>468.75</td>
<td>6011.70</td>
<td>NC</td>
</tr>
<tr>
<td>normal_vs_R2L</td>
<td>364.66</td>
<td>4773.09</td>
<td>14703.55</td>
</tr>
<tr>
<td>normal_vs_U2R</td>
<td>295.64</td>
<td>4785.66</td>
<td>14635.36</td>
</tr>
</tbody>
</table>
### Case of Study: Random Forest for KddCup’99

<table>
<thead>
<tr>
<th>Class</th>
<th>Instance Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>normal</td>
<td>972.781</td>
</tr>
<tr>
<td>DOS</td>
<td>3.883.370</td>
</tr>
<tr>
<td>PRB</td>
<td>41.102</td>
</tr>
<tr>
<td>R2L</td>
<td>1.126</td>
</tr>
<tr>
<td>U2R</td>
<td>52</td>
</tr>
</tbody>
</table>

#### Time elapsed (seconds) for Big data versions with 20 partitions:

<table>
<thead>
<tr>
<th>Datasets</th>
<th>RF-BigData</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10%</td>
</tr>
<tr>
<td>DOS_versus_normal</td>
<td>98</td>
</tr>
<tr>
<td>DOS_versus_PRB</td>
<td>100</td>
</tr>
<tr>
<td>DOS_versus_R2L</td>
<td>97</td>
</tr>
<tr>
<td>DOS_versus_U2R</td>
<td>93</td>
</tr>
<tr>
<td>normal_versus_PRB</td>
<td>94</td>
</tr>
<tr>
<td>normal_versus_R2L</td>
<td>92</td>
</tr>
<tr>
<td>normal_versus_U2R</td>
<td>93</td>
</tr>
</tbody>
</table>
Outline

- What Big Data?
- MapReduce Paradigm
- Hadoop and Mahout
- Computational Intelligence Approaches for Big Data
- Final Comments
Big Data: Selected Computational Intelligence approaches

The Field of Interest of the Society shall be the theory, design, application, and development of biologically and linguistically motivated computational paradigms emphasizing neural networks, connectionist systems, genetic algorithms, evolutionary programming, fuzzy systems, and hybrid intelligent systems in which these paradigms are contained.

Computational Intelligence
Big Data: Selected Computational Intelligence approaches

Fuzzy Sets - 1965 Lotfi Zadeh, Berkely

Fuzzy sets are sets whose elements have degrees of membership, as an extension of the classical notion of set.

![Diagram showing crisp sets and fuzzy sets for height classification](image)
Big Data: Selected Computational Intelligence approaches

Genetic algorithms

They are optimization algorithms, search and learning inspired in the process of

Natural and Genetic Evolution
Analizing 3 approaches

1. Evolutionary Computation for Big Data and Big Learning Workshop: Evolutionary Feature Weighting

2. On the use of MapReduce to build Linguistic Fuzzy Rule Based Classification Systems for Big Data

3. A Combined MapReduce-Windowing Two-Level Parallel Scheme for Evolutionary Prototype Generation
Big Data: Selected Computational Intelligence approaches

1. Evolutionary Computation for Big Data and Big Learning Workshop: Evolutionary Feature Weighting (+ preprocessing + Random Forest)

1. On the use of MapReduce to build Linguistic Fuzzy Rule Based Classification Systems for Big Data

2. A Combined MapReduce-Windowing Two-Level Parallel Scheme for Evolutionary Prototype Generation
Evolutionary Computation for Big Data and Big Learning Workshop

Data Mining Competition 2014: Self-deployment track

Objective: Contact map prediction

Details:
- 32 million instances
- 631 attributes (539 real & 92 nominal values)
- 2 classes
- 98% of negative examples
- About 56.7GB of disk space

Evaluation:
True positive rate · True negative rate
TPR · TNR

http://cruncher.ncl.ac.uk/bdcomp/index.pl?action=data

J. Bacardit et al, Contact map prediction using a large-scale ensemble of rule sets and the fusion of multiple predicted structural features, Bioinformatics 28 (19) (2012) 2441-2448
The challenge:

- Very large size of the training set
  - Does not fit all together in memory.
- Even large for the test set (5.1GB, 2.9 million instances)
- Relatively high dimensional data.
- Low ratio (<2%) of true contacts. Imbalance rate: > 49
  - Unbalanced problem!
Big Data: Selected Computational Intelligence approaches

Evolutionary Computation for Big Data and Big Learning Workshop

Our approach:

1. Balance the original training data
   (As first idea, it was extended)
   - Random Oversampling

2. Detect relevant features.
   1. Evolutionary Feature Weighting

3. Learning a model.
   - RandomForest

Classifying test set.
Big Data: Selected Computational Intelligence approaches

A MapReduce Approach for Random Oversampling

Low ratio of true contacts (<2%).
Imbalance rate: > 49. Unbalanced problem!

Over-Sampling
Random
Focused

Under-Sampling
Random
Focused

Cost Modifying (cost-sensitive)
Boosting/Bagging approaches (with preprocessing)
Big Data: Selected Computational Intelligence approaches

A MapReduce Approach for Random Oversampling

Low ratio of true contacts (<2%). Imbalance rate: > 49. **Unbalanced problem!**

Building a model with Random Forest

Training dataset

Bootstrap sample

Building Random trees
- Select $m$ variables at random for each node decision
- Compute best split using the $m$ variables as CART
- Fully grow the trees and do not prune them

Majority Voting

Predicted class
Big Data: Selected Computational Intelligence approaches

We initially focused on

- Oversampling rate: 100%

RandomForest:
- Number of used features: \(10 \log n + 1\); Number of trees: 100
- Number of maps: \{64, 190, 1024, 2048\}

<table>
<thead>
<tr>
<th>Nº mappers</th>
<th>TPR_tst</th>
<th>TNR_tst</th>
<th>TNR*TPR Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>0.601723</td>
<td>0.806269</td>
<td>0.485151</td>
</tr>
<tr>
<td>190</td>
<td>0.635175</td>
<td>0.773308</td>
<td>0.491186</td>
</tr>
<tr>
<td>1024</td>
<td>0.627896</td>
<td>0.756297</td>
<td>0.474876</td>
</tr>
<tr>
<td>2048</td>
<td>0.624648</td>
<td>0.759753</td>
<td>0.474578</td>
</tr>
</tbody>
</table>

Very low TPR (relevant!)

How to increase the TPR rate?

Idea: To increase the ROS percentage
Big Data: Selected Computational Intelligence approaches

How to increase the TPR rate?

**Idea:** To increase the ROS percentage

- Oversampling rate: \{100, 105, 110, 115, 130\}

RandomForest:
- Number of used features: 10; Number of trees: 100

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>TPR</th>
<th>TNR</th>
<th>TNR*TPR Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROS+RF (RS: 100%)</td>
<td>0.6351</td>
<td>0.7733</td>
<td>0.491186</td>
</tr>
<tr>
<td>ROS+RF (RS: 105%)</td>
<td>0.6568</td>
<td>0.7555</td>
<td>0.496286</td>
</tr>
<tr>
<td>ROS+RF (RS: 110%)</td>
<td>0.6759</td>
<td>0.7337</td>
<td>0.495941</td>
</tr>
<tr>
<td>ROS+RF (RS: 115%)</td>
<td>0.7041</td>
<td>0.7103</td>
<td>0.500175</td>
</tr>
<tr>
<td>ROS+RF (RS: 130%)</td>
<td>0.7472</td>
<td>0.6609</td>
<td>0.493913</td>
</tr>
</tbody>
</table>

Why we consider 190 maps?

Our cluster Atlas manages 192 cores (4 serves x 4 nodes x (2 x 6 cores) = 192 cores)

190 mappers

The higher ROS percentage, the higher TPR and the lower TNR
Big Data: Selected Computational Intelligence approaches

We analyze again the number of mappers with ROS (130%)

- Oversampling rate: {130}  Number of mappers: 128 to 2048

RandomForest:
- Number of used features: 10; Number of trees: 100

ROS+RF (130%) and mappers

<table>
<thead>
<tr>
<th>Num Mappers</th>
<th>TPR</th>
<th>TNR</th>
<th>TNR*TPR Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>0.7450</td>
<td>0.6678</td>
<td>0.497624</td>
</tr>
<tr>
<td>190</td>
<td>0.7472</td>
<td>0.6609</td>
<td>0.493913</td>
</tr>
<tr>
<td>256</td>
<td>0.7518</td>
<td>0.6531</td>
<td>0.491064</td>
</tr>
<tr>
<td>1024</td>
<td>0.7644</td>
<td>0.6121</td>
<td>0.467969</td>
</tr>
<tr>
<td>2048</td>
<td>0.7755</td>
<td>0.5860</td>
<td>0.454528</td>
</tr>
</tbody>
</table>

The equilibrium with the number of mappers is very important

The less number of maps, the less TPR and the high TNR
Third component: MapReduce Approach for Feature Weighting
for getting a major equilibrium between classes

Map Side
- Each map reads one block from the dataset.
- Perform an Evolutionary Feature Weighting step.
- **Output:** A real vector that represents the degree of importance of each feature.
- Number of maps: 32768 (less than 1000 original data per map)

Reduce Side
- Aggregate the feature’s weights
- A feature is finally selected if it overcomes a given threshold.
- **Output:** A binary vector that represents the final selection

Big Data: Selected Computational Intelligence approaches

**Experimental study**

Random Oversampling:
- Oversampling ratio. Analyzed values: {100 to 130}

**Feature Weigthing:**
- Threshold --> number of selected features.
- Set of features: \{19, 63, 90, 146\}
- Number of maps: 32768

RandomForest:
- Number of used features: \{\log \text{NumFeatures}, 2 \times \log +1\}
- Number of trees: \{100\}
- Number of maps: \{32, 64, 128, 190, 256, 512\}
Big Data: Selected Computational Intelligence approaches

**We investigate: The use of Evolutionary Feature Weighting. It allows us to construct several subset of features (changing the threshold).**

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>128 mappers</th>
<th>64 mappers</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TNR*TPR Training</td>
<td>TPR</td>
</tr>
<tr>
<td>ROS+RF (130% - Feature Weighting 19)</td>
<td>0.621638</td>
<td>0.684726</td>
</tr>
<tr>
<td>ROS+RF (115% - Feature Weighting 19)</td>
<td>0.628225</td>
<td>0.674569</td>
</tr>
<tr>
<td>ROS+RF (100% - Feature Weighting 19)</td>
<td>0.635029</td>
<td>0.629397</td>
</tr>
<tr>
<td>ROS+RF (130% - Feature Weighting 63)</td>
<td>0.634843</td>
<td>0.683800</td>
</tr>
<tr>
<td>ROS+RF (115% - Feature Weighting 63)</td>
<td>0.639319</td>
<td>0.677015</td>
</tr>
<tr>
<td>ROS+RF (100% - Feature Weighting 63)</td>
<td>0.648723</td>
<td>0.638567</td>
</tr>
</tbody>
</table>
**On the Random Forest:** We decided to investigate the influence of the Random Forest’s parameters (internal features and number of trees) as well as a higher number of features (90).

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>TNR*TPR Training</th>
<th>TPR</th>
<th>TNR</th>
<th>TNR*TPR Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROS+ RF (130%+ FW 63+6f+100t)</td>
<td>0.604687</td>
<td>0.698152</td>
<td>0.742462</td>
<td>0.518351</td>
</tr>
<tr>
<td>ROS+ RF (130%+ FW 63+6f+200t)</td>
<td>0.632078</td>
<td>0.700064</td>
<td>0.745225</td>
<td>0.521705</td>
</tr>
<tr>
<td>ROS+ RF (140%+ FW 63+15f+200t)</td>
<td>0.627409</td>
<td>0.719678</td>
<td>0.728912</td>
<td>0.524582</td>
</tr>
<tr>
<td>ROS+ RF (140%+ FW 90+15f+200t)</td>
<td>0.635855</td>
<td>0.722639</td>
<td>0.726397</td>
<td>0.524923</td>
</tr>
<tr>
<td>ROS+ RF (140%+ FW 90+25f+200t)</td>
<td>0.629273</td>
<td>0.721652</td>
<td>0.729740</td>
<td>0.526618</td>
</tr>
<tr>
<td>ROS+ RF (140%+ FW 90+30f+200t)</td>
<td>0.630104</td>
<td>0.721138</td>
<td>0.730323</td>
<td>0.526664</td>
</tr>
</tbody>
</table>

Correct decisions with FW 90 and RF with 25/30f and 200 trees. Good trade off between TPR and TNR
Big Data: Selected Computational Intelligence approaches

Last decision: We investigated to increase ROS until 180% reducing the number of mappers (64 mappers)

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>TNR*TPR Training</th>
<th>TPR</th>
<th>TNR</th>
<th>TNR*TPR Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROS+ RF (130%+ FW 90+25f+200t)</td>
<td>0.736987</td>
<td>0.671279</td>
<td>0.783911</td>
<td>0.526223</td>
</tr>
<tr>
<td>ROS+ RF (140%+ FW 90+25f+200t)</td>
<td>0.717048</td>
<td>0.695109</td>
<td>0.763951</td>
<td>0.531029</td>
</tr>
<tr>
<td>ROS+ RF (150%+ FW 90+25f+200t)</td>
<td>0.706934</td>
<td>0.705882</td>
<td>0.753625</td>
<td>0.531971</td>
</tr>
<tr>
<td>ROS+ RF (160%+ FW 90+25f+200t)</td>
<td>0.698769</td>
<td>0.718692</td>
<td>0.741976</td>
<td>0.533252</td>
</tr>
<tr>
<td>ROS+ RF (170%+ FW 90+25f+200t)</td>
<td>0.682910</td>
<td>0.730432</td>
<td>0.730183</td>
<td>0.533349</td>
</tr>
<tr>
<td>ROS+ RF (180%+ FW 90+25f+200t)</td>
<td>0.678986</td>
<td>0.737381</td>
<td>0.722583</td>
<td>0.532819</td>
</tr>
</tbody>
</table>

To increase ROS and reduce the mappers number lead us to increase the TPR maintaining a good TNR rate. An equilibrium and good results
Big Data: Selected Computational Intelligence approaches

**Evolutionary Computation for Big Data and Big Learning Workshop**

**Results of the competition:** Contact map prediction

<table>
<thead>
<tr>
<th>Team Name</th>
<th>TPR</th>
<th>TNR</th>
<th>Acc</th>
<th>TPR * TNR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Efdamis</td>
<td>0.730432</td>
<td>0.730183</td>
<td>0.730188</td>
<td>0.533349</td>
</tr>
<tr>
<td>ICOS</td>
<td>0.703210</td>
<td>0.730155</td>
<td>0.729703</td>
<td>0.513452</td>
</tr>
<tr>
<td>UNSW</td>
<td>0.699159</td>
<td>0.727631</td>
<td>0.727153</td>
<td>0.508730</td>
</tr>
<tr>
<td>HyperEns</td>
<td>0.640027</td>
<td>0.763378</td>
<td>0.761308</td>
<td>0.488583</td>
</tr>
<tr>
<td>PUC-Rio_ICA</td>
<td>0.657092</td>
<td>0.714599</td>
<td>0.713634</td>
<td>0.469558</td>
</tr>
<tr>
<td>Test2</td>
<td>0.632009</td>
<td>0.735545</td>
<td>0.733808</td>
<td>0.464871</td>
</tr>
<tr>
<td>EmeraldLogic</td>
<td>0.686926</td>
<td>0.669737</td>
<td>0.670025</td>
<td>0.460059</td>
</tr>
<tr>
<td>LidiaGroup</td>
<td>0.653042</td>
<td>0.695753</td>
<td>0.695036</td>
<td>0.454356</td>
</tr>
</tbody>
</table>

[http://cruncher.ncl.ac.uk/bdcomp/index.pl?action=ranking](http://cruncher.ncl.ac.uk/bdcomp/index.pl?action=ranking)
Big Data: Selected Computational Intelligence approaches

**Evolutionary Computation for Big Data and Big Learning Workshop**

**Results of the competition:** Contact map prediction

<table>
<thead>
<tr>
<th>Team Name</th>
<th>TPR</th>
<th>TNR</th>
<th>Acc</th>
<th>TPR * TNR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Efdamis</td>
<td>0.730432</td>
<td>0.730183</td>
<td>0.730188</td>
<td>0.533349</td>
</tr>
<tr>
<td>ICOS</td>
<td>0.703210</td>
<td>0.730155</td>
<td>0.729703</td>
<td>0.513452</td>
</tr>
<tr>
<td>UNSW</td>
<td>0.699159</td>
<td>0.727631</td>
<td>0.727153</td>
<td>0.508730</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>TNR*TPR Training</th>
<th>TPR</th>
<th>TNR</th>
<th>TNR*TPR Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROS+RF (130% - Feature Weighting 63)</td>
<td>0.726350</td>
<td>0.66949</td>
<td>0.775652</td>
<td>0.519292</td>
</tr>
<tr>
<td>ROS+RF (115% - Feature Weighting 63)</td>
<td>0.736596</td>
<td>0.652692</td>
<td>0.790822</td>
<td>0.516163</td>
</tr>
<tr>
<td>ROS+RF (100% - Feature Weighting 63)</td>
<td>0.752824</td>
<td>0.626190</td>
<td>0.811176</td>
<td>0.507950</td>
</tr>
</tbody>
</table>

To increase ROS and to use Evolutionary feature weighting were two good decisions
A feature selection over 631 attributes is a good approach.

Evolutionary feature weight allows us to manage the threshold for choosing the number of features.

A high ROS with a reduced number of MAPs increase the TPR.

The final model: ROS(170%) + Evol. FW (90 features) + RF (25f+ 200t) − 64 MAPERS \( (TPR \cdot TNR = 0.533349) \)
Big Data: Selected Computational Intelligence approaches

1. A Combined MapReduce-Windowing Two-Level Parallel Scheme for Evolutionary Prototype Generation

2. On the use of MapReduce to build Linguistic Fuzzy Rule Based Classification Systems for Big Data

3. Evolutionary Computation for Big Data and Big Learning Workshop: Evolutionary Feature Weighting
Uncertainty and Big Data

- Uncertainty is inherent to Big Data due to
  - Heterogeneous sources
  - Variety in data
  - Incomplete data
  - Veracity in question
- Fuzzy Rule Based Classification Systems can manage
  - Uncertainty
  - Ambiguity
  - Vagueness
Chi-FRBCS-BigData

- MapReduce design based on the FRBCS algorithm (Chi et al).
- Uses two different MapReduce processes
  - Phase 1: Building the Fuzzy Rule Base
  - Phase 2: Estimating the class of samples belonging to big data sample sets
- Two versions which differ in the *Reduce function* of the building of the FRB have been produced
  - Chi-FRBCS-BigData-Max
  - Chi-FRBCS-BigData-Average

Chi-FRBCS

- Produces rules like “Rule $R_j$: IF $x_1$ IS $A_{1j}$ AND ... AND $x_n$ IS $A_{nj}$ THEN Class = $C_j$ with $RW_j$”
- Builds the fuzzy partition using equally distributed triangular membership functions
- Builds the FRB creating a fuzzy rule associated to each example
- Rules with the same antecedent may be created:
  - Same consequent $\rightarrow$ Delete duplicated rules
  - Different consequent $\rightarrow$ Preserve highest weight rule

Z. Chi, H. Yan and T. Pham, Fuzzy algorithms with applications to image processing and pattern recognition, World Scientific, 1996.
Big Data: Selected Computational Intelligence approaches

Building the RB with Chi-FRBCS-BigData

The key of a MapReduce data partitioning approach is usually on the reduce phase: Two alternative reducers (Max vs average weights)
Big Data: Selected Computational Intelligence approaches

Building the FRB with Chi-FRBCS-BigData-Max

Final RB generation

RB1, R1, C1, RW = 0.8743
RB2, R1, C2, RW = 0.9254
RB3, R2, C1, RW = 0.7142
RB4, R1, C2, RW = 0.2143
RB5, R2, C1, RW = 0.8215
Big Data: Selected Computational Intelligence approaches

Building the FRB with Chi-FRBCS-BigData-Ave

REDUCE

Final RB generation

RB1, R1, C1, RW = 0.8743
RB2, R1, C2, RW = 0.9254
RB3, R2, C1, RW = 0.7142
RB4, R1, C2, RW = 0.2143
RB5, R2, C1, RW = 0.8215
RBn, R1, C1, RW = 0.7784
RBn, R2, C2, RW = 0.8215
RC1, C1, RWave = 0.8033
RC2, C2, RWave = 0.5699
Big Data: Selected Computational Intelligence approaches

Estimating the class of a Big dataset with Chi-FRBCS-BigData

INITIAL

Original classification set → Classification set map_1 → Classification set map_2 → ... → Classification set map_n

MAP

Sample_11: Actual class C1; Predicted class C1
Sample_12: Actual class C2; Predicted class C2
Sample_13: Actual class C1; Predicted class C2
...

Sample_21: Actual class C1; Predicted class C1
Sample_22: Actual class C2; Predicted class C2
Sample_23: Actual class C2; Predicted class C2
...

Sample_n1: Actual class C1; Predicted class C2
Sample_n2: Actual class C2; Predicted class C2
Sample_n3: Actual class C1; Predicted class C2
...

Predictions set_1 → Predictions set_2 → ... → Predictions set_n

FINAL

Sample_11: Actual class C1; Predicted class C1
Sample_12: Actual class C2; Predicted class C2
Sample_13: Actual class C1; Predicted class C2
...

Sample_21: Actual class C1; Predicted class C1
Sample_22: Actual class C2; Predicted class C2
Sample_23: Actual class C2; Predicted class C2
...

Sample_n1: Actual class C1; Predicted class C2
Sample_n2: Actual class C2; Predicted class C2
Sample_n3: Actual class C1; Predicted class C2
...

Final predictions file

Mappers classification sets prediction
### Experimental Framework

**Chi-FRBCS-BigData-Max vs. Chi-FRBCS-BigData-Ave**

- 6 Datasets with two classes to alleviate the small sample size problem
- Stratified 10 fold cross-validation
- Parameters:
  - Conjunction Operator: Product T-norm
  - Rule Weight: Penalized Certainty Factor
  - Fuzzy Reasoning Method: Winning Rule
  - Number of fuzzy labels per variable: 3 labels
  - Number of mappers: 16, 32, 64

---

<table>
<thead>
<tr>
<th>Datasets</th>
<th>#Ex.</th>
<th>#Atts.</th>
<th>Selected classes</th>
<th>#Samples per class</th>
</tr>
</thead>
<tbody>
<tr>
<td>RLCP</td>
<td>5749132</td>
<td>2</td>
<td>(FALSE; TRUE)</td>
<td>(5728201; 20931)</td>
</tr>
<tr>
<td>Kddeup_DOS_vs_normal</td>
<td>4856151</td>
<td>41</td>
<td>(DOS; normal)</td>
<td>(3883370; 972781)</td>
</tr>
<tr>
<td>Poker_0_vs_1</td>
<td>946799</td>
<td>10</td>
<td>(0; 1)</td>
<td>(513702; 433097)</td>
</tr>
<tr>
<td>Covtype_2_vs_1</td>
<td>495141</td>
<td>54</td>
<td>(2; 1)</td>
<td>(283301; 211840)</td>
</tr>
<tr>
<td>Census</td>
<td>141544</td>
<td>41</td>
<td>(&lt;=50000.; 50000+)</td>
<td>(133430; 8114)</td>
</tr>
<tr>
<td>Fars Fatal_Inj_vs_No_Inj</td>
<td>62123</td>
<td>29</td>
<td>(Fatal_Inj; No_Inj)</td>
<td>(42116; 20007)</td>
</tr>
</tbody>
</table>
Big Data: Selected Computational Intelligence approaches

### Analysis of the Precision

<table>
<thead>
<tr>
<th>Datasets</th>
<th>16 mappers</th>
<th></th>
<th></th>
<th></th>
<th>32 mappers</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Chi-BigData-Max</td>
<td>Chi-BigData-Ave</td>
<td></td>
<td></td>
<td>Chi-BigData-Max</td>
<td>Chi-BigData-Ave</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Acc_</td>
<td>Acc_H</td>
<td>Acc_R</td>
<td>Acc_M</td>
<td></td>
<td>Acc_</td>
<td>Acc_H</td>
<td>Acc_R</td>
</tr>
<tr>
<td>Poker_0_vs_1</td>
<td>62.18</td>
<td>59.88</td>
<td>62.58</td>
<td>60.35</td>
<td></td>
<td>61.27</td>
<td>58.93</td>
<td>61.82</td>
</tr>
<tr>
<td>Covtype_2_vs_1</td>
<td>74.77</td>
<td>74.72</td>
<td>74.77</td>
<td>74.69</td>
<td></td>
<td>74.69</td>
<td>74.62</td>
<td>74.88</td>
</tr>
<tr>
<td>Census</td>
<td>97.14</td>
<td>93.75</td>
<td>97.15</td>
<td>93.52</td>
<td></td>
<td>97.11</td>
<td>93.48</td>
<td>97.12</td>
</tr>
<tr>
<td>Fars Fatal_Inj vs No_Inj</td>
<td>96.69</td>
<td>94.75</td>
<td>97.06</td>
<td>95.01</td>
<td></td>
<td>96.49</td>
<td>94.26</td>
<td>96.87</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td>88.39</td>
<td>87.11</td>
<td>88.52</td>
<td>87.19</td>
<td></td>
<td>88.49</td>
<td>86.81</td>
<td>88.37</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Datasets</th>
<th>64 mappers</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Chi-BigData-Max</td>
<td>Chi-BigData-Ave</td>
<td></td>
<td></td>
<td>Chi-BigData-Max</td>
<td>Chi-BigData-Ave</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Acc_</td>
<td>Acc_H</td>
<td>Acc_R</td>
<td>Acc_M</td>
<td></td>
<td>Acc_</td>
<td>Acc_H</td>
<td>Acc_R</td>
</tr>
<tr>
<td>Poker_0_vs_1</td>
<td>60.45</td>
<td>57.95</td>
<td>60.88</td>
<td>58.12</td>
<td></td>
<td>61.27</td>
<td>58.93</td>
<td>61.82</td>
</tr>
<tr>
<td>Covtype_2_vs_1</td>
<td>74.67</td>
<td>74.52</td>
<td>75.05</td>
<td>74.96</td>
<td></td>
<td>74.69</td>
<td>74.62</td>
<td>74.88</td>
</tr>
<tr>
<td>Census</td>
<td>97.07</td>
<td>93.30</td>
<td>97.13</td>
<td>93.11</td>
<td></td>
<td>97.11</td>
<td>93.48</td>
<td>97.12</td>
</tr>
<tr>
<td>Fars Fatal_Inj vs No_Inj</td>
<td>96.27</td>
<td>93.98</td>
<td>96.76</td>
<td>94.56</td>
<td></td>
<td>96.49</td>
<td>94.26</td>
<td>96.87</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td>88.06</td>
<td>86.55</td>
<td>88.23</td>
<td>86.72</td>
<td></td>
<td>88.49</td>
<td>86.81</td>
<td>88.37</td>
</tr>
</tbody>
</table>

- Performance improves with less mappers
- Chi-BigData-Ave obtains slightly better classification results
FRBCS for Big Data: Final Comments

- We introduced a **linguistic FRBCS for Big Data** named Chi-FRBCS-BigData:
  - Manages big datasets
  - Without damaging the classification accuracy
  - Fast response times
- Designed under the MapReduce framework with different rule fusion approaches. Presenting good results
- **Two versions** – differ on the Reduce step:
  - Chi-FRBCS-BigData-Max
  - Chi-FRBCS-BigData-Ave
Big Data: Selected Computational Intelligence approaches

1. On the use of MapReduce to build Linguistic Fuzzy Rule Based Classification Systems for Big Data

2. Evolutionary Computation for Big Data and Big Learning Workshop: Evolutionary Feature Weighting

3. A Combined MapReduce-Windowing Two-Level Parallel Scheme for Evolutionary Prototype Generation

Prototype Generation: properties

- The NN classifier is one of the most used algorithms in machine learning.

- **Prototype Generation** (PG) processes learn new representative examples if needed. It results in more accurate results.

- **Advantages:**
  - PG reduces the computational costs and high storage requirements of NN.
  - Evolutionary PG algorithms highlighted as the best performing approaches.

- **Main issues:**
  - Dealing with big data becomes impractical in terms of *Runtime* and *Memory consumption*. Especially for EPG.
Evolutionary Prototype Generation

- Evolutionary PG algorithms are typically based on adjustment of the positioning of the prototypes.
- Each individual encodes a single prototype or a complete generated set with real codification.
- The fitness function is computed as the classification performance in the training set using the Generated Set.
- Currently, best performing approaches use differential evolution.


More information about Prototype Reduction can be found in the SCI2S thematic website: http://sci2s.ugr.es/pr
Parallelizing PG with MapReduce

Map phase:
- Each map constitutes a subset of the original training data.
- It applies a Prototype Generation step.
- For evaluation, it uses **Windowing: Incremental Learning with Alternating Strata (ILAS)**
- As output, it returns a Generated Set of prototypes.

Reduce phase:
- We established a single reducer.
- It consists of an iterative aggregation of all the resulting generated sets.
- As output, it returns the final Generated Set.
Parallelizing PG with MapReduce

The key of a MapReduce data partitioning approach is usually on the reduce phase.

Two alternative reducers:

- **Join**: Concatenates all the resulting generated sets.
  - This process does not guarantee that the final generated set does not contain irrelevant or even harmful instances

- **Fusion**: This variant eliminates redundant prototypes by fusion of prototypes. Centroid-based PG methods: ICPL.

Big Data: Selected Computational Intelligence approaches

Windowing: Incremental Learning with Alternating Strata (ILAS)

- Training set is divided into strata, each iteration just uses one of the stratum.

Main properties:
- Avoids a (potentially biased) static prototype selection
- This mechanism also introduces some generalization pressure

Big Data: Selected Computational Intelligence approaches

The MRW-EPG scheme

Windowing: Incremental Learning with Alternating Strata (ILAS)
Experimental Study

- **PokerHand data set.** 1 million of instances, 3x5 fcv.
- **Performance measures:** Accuracy, reduction rate, runtime, test classification time and speed up.
- PG technique tested: IPADECS.

### TABLE I: Parameter specification for all the methods

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>MRW-EPW</td>
<td>Mappers = 16/32/64/128, Reducers= 1 Window = [1-7], ReduceType = Join/Fusion.</td>
</tr>
<tr>
<td>IPADECS</td>
<td>PopulationSize = 10, iterations of Basic DE = 500 iterSFGSS =8, iterSFHC=20, Fl=0.1, Fu=0.9 Filtering method = RT2</td>
</tr>
<tr>
<td>ICLP2 (Fusion)</td>
<td></td>
</tr>
<tr>
<td>NN</td>
<td>Number of neighbors = 1, Euclidean distance.</td>
</tr>
</tbody>
</table>
Big Data: Selected Computational Intelligence approaches

Results

PokerHand: Accuracy Test vs. Runtime results obtained by MRW-EPG
Big Data: Selected Computational Intelligence approaches

Results

TABLE III: Results obtained incorporating the windowing scheme with MRW-EPG and fusion reducer.

<table>
<thead>
<tr>
<th>#Windows nw</th>
<th>#Mappers</th>
<th>Training</th>
<th>Test</th>
<th>Runtime</th>
<th>Reduction rate</th>
<th>Classification time (TS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>16</td>
<td>0.5121</td>
<td>0.0028</td>
<td>0.5120</td>
<td>0.0031</td>
<td>15058.4740</td>
</tr>
<tr>
<td>2</td>
<td>16</td>
<td>0.5115</td>
<td>0.0035</td>
<td>0.5113</td>
<td>0.0036</td>
<td>8813.7134</td>
</tr>
<tr>
<td>3</td>
<td>16</td>
<td>0.5038</td>
<td>0.0032</td>
<td>0.5039</td>
<td>0.0033</td>
<td>4666.5424</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>0.5052</td>
<td>0.0060</td>
<td>0.5055</td>
<td>0.0057</td>
<td>4095.8610</td>
</tr>
<tr>
<td>5</td>
<td>16</td>
<td>0.5041</td>
<td>0.0024</td>
<td>0.5034</td>
<td>0.0022</td>
<td>3244.0716</td>
</tr>
<tr>
<td>6</td>
<td>16</td>
<td>0.5031</td>
<td>0.0042</td>
<td>0.5028</td>
<td>0.0041</td>
<td>2639.4266</td>
</tr>
<tr>
<td>7</td>
<td>16</td>
<td>0.5000</td>
<td>0.0067</td>
<td>0.4998</td>
<td>0.0069</td>
<td>2099.5182</td>
</tr>
<tr>
<td>1</td>
<td>32</td>
<td>0.5089</td>
<td>0.0031</td>
<td>0.5086</td>
<td>0.0029</td>
<td>6963.5734</td>
</tr>
<tr>
<td>2</td>
<td>32</td>
<td>0.5084</td>
<td>0.0045</td>
<td>0.5080</td>
<td>0.0041</td>
<td>4092.5484</td>
</tr>
<tr>
<td>3</td>
<td>32</td>
<td>0.5067</td>
<td>0.0025</td>
<td>0.5065</td>
<td>0.0024</td>
<td>2343.1542</td>
</tr>
<tr>
<td>4</td>
<td>32</td>
<td>0.5012</td>
<td>0.0045</td>
<td>0.5012</td>
<td>0.0039</td>
<td>1639.0032</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>0.5012</td>
<td>0.0045</td>
<td>0.5012</td>
<td>0.0039</td>
<td>1639.0032</td>
</tr>
<tr>
<td>6</td>
<td>32</td>
<td>0.4824</td>
<td>0.0104</td>
<td>0.4820</td>
<td>0.0101</td>
<td>1083.1116</td>
</tr>
<tr>
<td>7</td>
<td>32</td>
<td>0.4838</td>
<td>0.0072</td>
<td>0.4835</td>
<td>0.0065</td>
<td>1129.8838</td>
</tr>
</tbody>
</table>
Big Data: Selected Computational Intelligence approaches

Results: Speed-up

![Graph showing speed-up of PokerHand (16 mappers) with number of windows and speedup on the y-axis and number of windows on the x-axis. The graph compares speed-up for different ReduceTypes: Join and IterativeFusion.]
There is a good synergy between the windowing and MapReduce approaches. They complement themselves in the proposed two-level scheme.

Without windowing, evolutionary prototype generation could not be applied to data sets larger than approximately ten thousands instances.

The application of this model has resulted in a very big reduction of storage requirements and classification time for the NN rule.
Outline

- What Big Data?
- MapReduce Paradigm
- Hadoop and Mahout
- Computational Intelligence Approaches for Big Data
- Final Comments
Final Comments

- Parallelization of machine learning algorithms with data-partitioning approaches can be successfully performed with MapReduce.
- Partitioning and applying the machine learning algorithm to each part.
- Focus on the combination phase (reduce). The combination of models is the challenge for each algorithm.
- Computational Intelligence based algorithms may provide good tools for managing imprecision, process optimization ....
Final Comments

The Challenges of Big Data

- Efficiency requirements for Algorithm
  - Traditionally, “efficient” algorithms
    - Run in (small) polynomial time: $O(n \log n)$
    - Use linear space: $O(n)$
  - For large data sets, efficient algorithms
    - Must run in linear or even sub-linear time: $o(n)$
    - Must use up to poly-logarithmic space: $(\log n)^2$

- Clean Big Data
  - Noise in data distorts
    - Computation results
    - Search results
  - Need automatic methods for “cleaning” the data
    - Duplicate elimination
    - Quality evaluation

- Computing Model
  - Accuracy and Approximation
  - Efficiency
**Final Comments: On the limitations of Hadoop. New frameworks/platforms**

<table>
<thead>
<tr>
<th>Framework/Platform</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Procesamiento iterativo de grafos</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Twister (Indiana University)</strong></td>
<td><a href="http://www.iterativemapreduce.org/">http://www.iterativemapreduce.org/</a></td>
</tr>
<tr>
<td><strong>Distributed GraphLab</strong></td>
<td><a href="https://github.com/graphlab-code/graphlab">https://github.com/graphlab-code/graphlab</a></td>
</tr>
<tr>
<td><strong>HaLoop (University of Washington)</strong></td>
<td><a href="http://clue.cs.washington.edu/node/14">http://clue.cs.washington.edu/node/14</a></td>
</tr>
<tr>
<td><strong>Twister (Indiana University)</strong></td>
<td><a href="http://www.iterativemapreduce.org/">http://www.iterativemapreduce.org/</a></td>
</tr>
<tr>
<td><strong>Cluster propios y Amazon EC2 cloud</strong></td>
<td></td>
</tr>
<tr>
<td><strong>GPU based platforms</strong></td>
<td><strong>Mars Grex</strong></td>
</tr>
<tr>
<td><strong>Spark (UC Berkeley)</strong></td>
<td>(100 times more efficient than Hadoop, including iterative algorithms, according to creators)</td>
</tr>
</tbody>
</table>
In many new applications - face recognition, speech understanding, recommendations, or fraud detection - bigger data does produce better results.

To help clarify the different meanings of "Big Data", Dr. Piatetsky proposes to consider 3 stages of Big Data.
Final Comments

Vignette in a Spanish newspaper

the next floods will be data floods ...
Big Data
Selected Computational Intelligence approaches

Thanks!!!