A Simple Approach Towards Visualizing and Evaluating Complexity of Textual Case Bases

Sutanu Chakraborti, Ulises Cervino Beresi, Nirmalie Wiratunga, Stewart Massie, Robert Lothian and Stuart Watt

THE ROBERT GORDON UNIVERSITY ABERDEEN

School of Computing

Overview

- Motivation
- Case Base Visualisation
  - “Case Base as Image” Metaphor for Visualization
  - Algorithm for Case Base “Stacking”
- Image Compression to Evaluate Complexity
  - Unsupervised Tasks - GAME
  - Extension for Supervised Classification
- Experimental Results

Motivation

Visualization is useful in the Textual CBR (TCBR) for:

1. Easing knowledge acquisition from human experts
2. Visually evaluating goodness of the underlying representation
3. Aiding case base maintenance, by revealing redundant or noisy features and cases
4. Presenting and explaining retrieved results to end users

OFFLINE

ONLINE

Complexity - alignment between problem & solution space

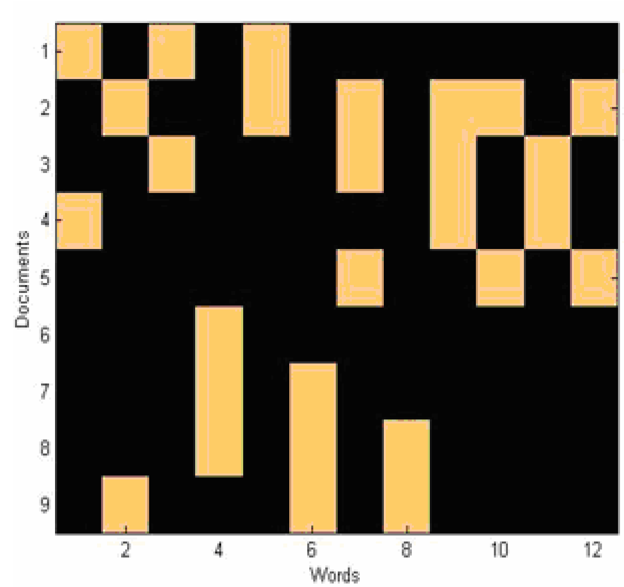
- Important for all offline tasks mentioned above, especially tasks 2 and 3
- Offers a quantitative as opposed to qualitative insight into the characteristics of the case base
- Shares similar goals to visualisation

The Case Base as an Image

c1: *Human* machine *interface* for Lab ABC *computer* applications
c2: A *survey* of *user* opinion of *computer system response time*
c3: The *EPS user interface* management system
c4: *System* and *human system* engineering testing of *EPS*
c5: Relation of *user*-perceived *response time* to error measurement

m1: The generation of random, binary, unordered *trees*
m2: The intersection *graphs* of paths in *trees*
m3: *Graph minors* IV : Widths of *trees* and well-quasi-ordering
m4: *Graph minors*: A *survey*

	human	survey	interface	trees	computer	graph	user	minors	system	time	EPS	time
c1 :	1	0	1	0	1	0	0	0	0	0	0	0
c2:	0	1	0	0	1	0	1	0	1	1	0	1
c3:	0	0	1	0	0	0	1	0	1	0	1	0
c4:	1	0	0	0	0	0	0	0	1	0	1	0
c5:	0	0	0	0	0	0	1	0	0	1	0	1
m1:	0	0	0	1	0	0	0	0	0	0	0	0
m2:	0	0	0	1	0	1	0	0	0	0	0	0
m3:	0	0	0	1	0	1	0	1	0	0	0	0
m4:	0	1	0	0	0	1	0	1	0	0	0	0



Is this picture useful ?

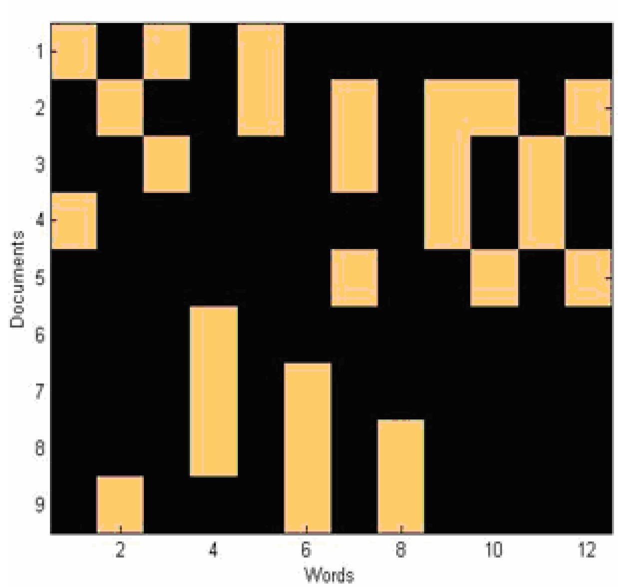
Yes *and* **No**

The Case Base as an Image

c1: *Human* machine *interface* for Lab ABC *computer* applications
c2: A *survey* of *user* opinion of *computer system response time*
c3: The *EPS user interface* management system
c4: *System* and *human system* engineering testing of *EPS*
c5: Relation of *user*-perceived *response time* to error measurement

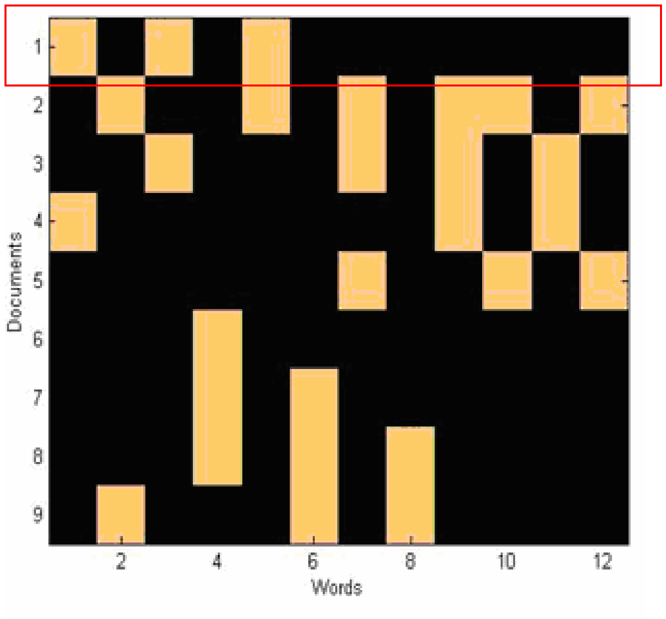
m1: The generation of random, binary, unordered *trees*
m2: The intersection *graphs* of paths in *trees*
m3: *Graph minors* IV : Widths of *trees* and well-quasi-ordering
m4: *Graph minors*: A *survey*

	human	survey	interface	trees	computer	graph	user	minors	system	time	EPS	time
c1 :	1	0	1	0	1	0	0	0	0	0	0	0
c2:	0	1	0	0	1	0	1	0	1	1	0	1
c3:	0	0	1	0	0	0	1	0	1	0	1	0
c4:	1	0	0	0	0	0	0	0	1	0	1	0
c5:	0	0	0	0	0	0	1	0	0	1	0	1
m1:	0	0	0	1	0	0	0	0	0	0	0	0
m2:	0	0	0	1	0	1	0	0	0	0	0	0
m3:	0	0	0	1	0	1	0	1	0	0	0	0
m4:	0	1	0	0	0	1	0	1	0	0	0	0



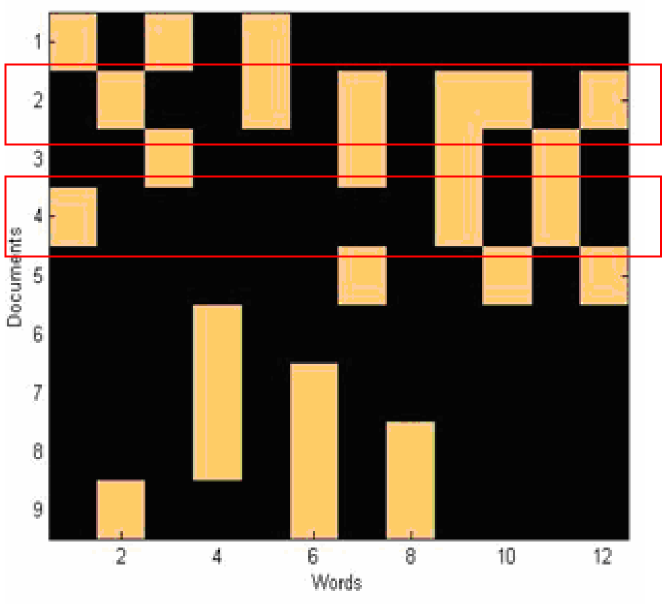
- Conveys little information about underlying patterns in terms of word or document clusters
- Sensitive to the ordering of words and documents in the matrix
- tells us little about the complexity of the underlying case base.

Stacking : Step 1



- The first case row in the original matrix is retained as it is
- Compute similarity of all other cases to the first case

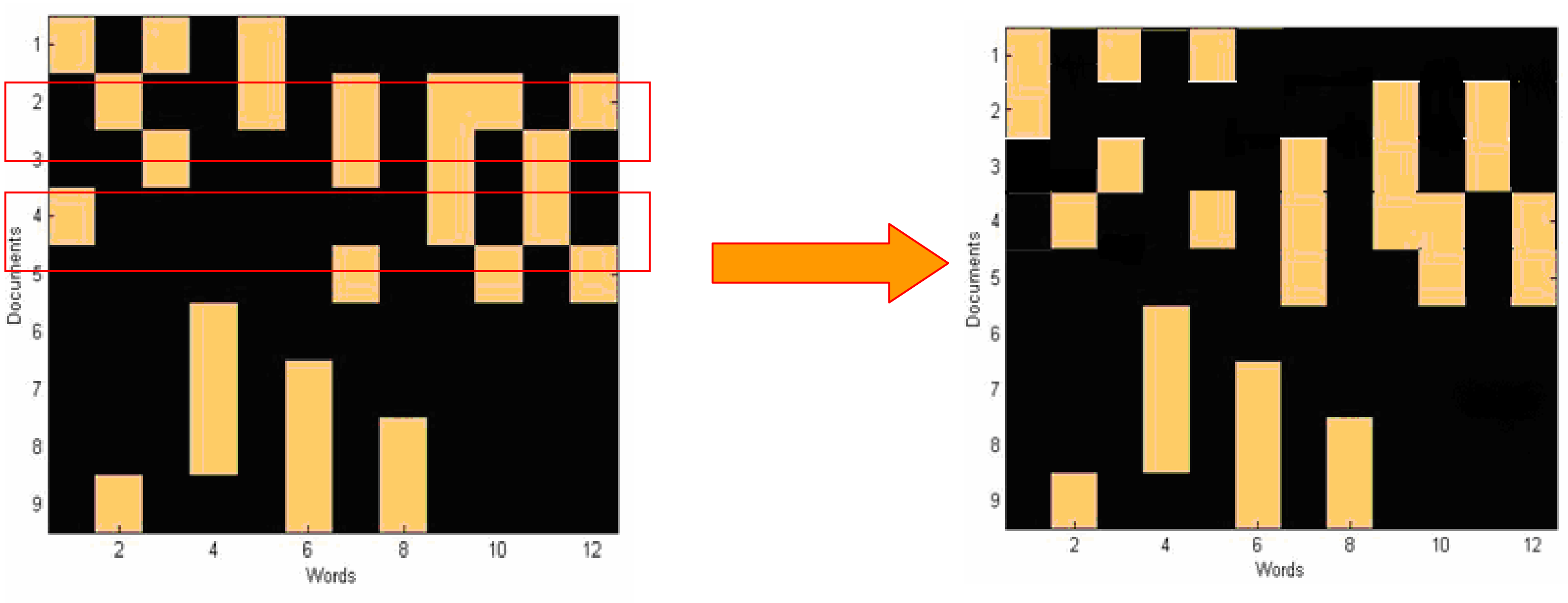
Stacking : Step 2



Rows 2 and 4 are swapped since case 4 is more similar to case 1 than case 2

- The case most similar to the first case is stacked next to it, by swapping positions with the existing second row.
- If more than one case is found to be equally similar, one of them is chosen randomly.

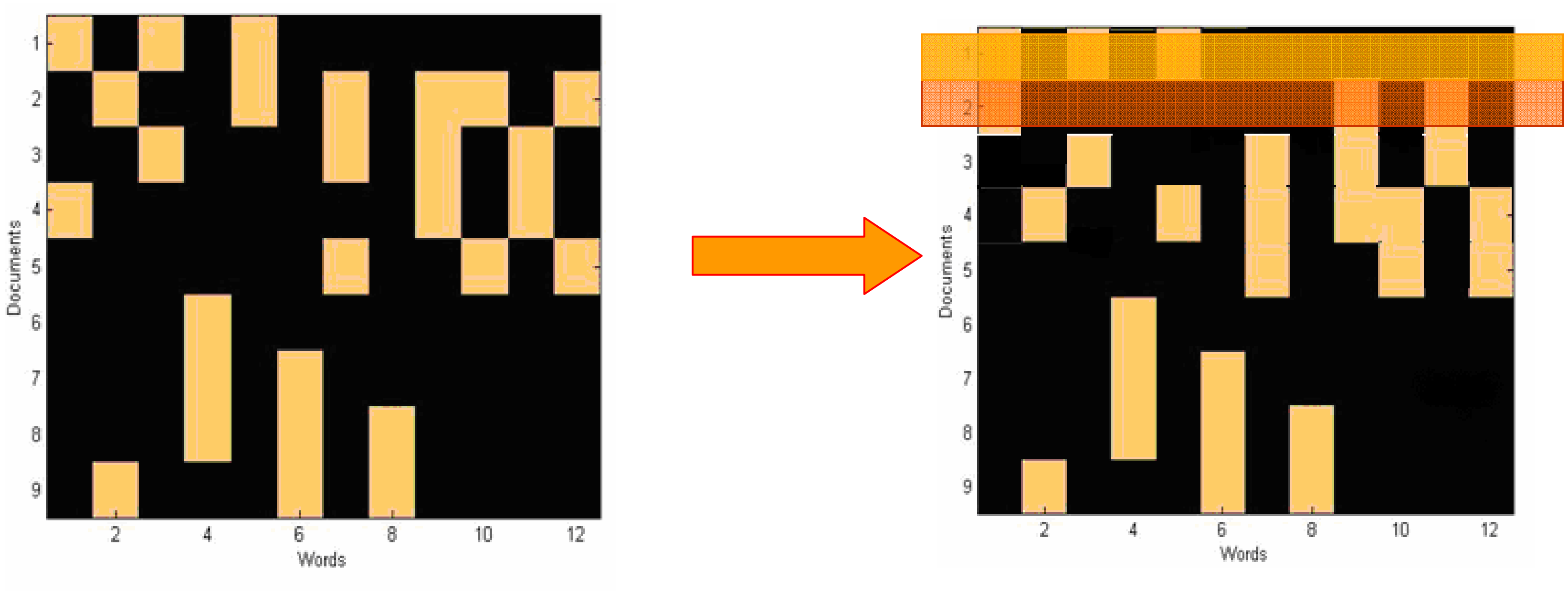
Stacking : Step 2



After swapping rows
2 and 4

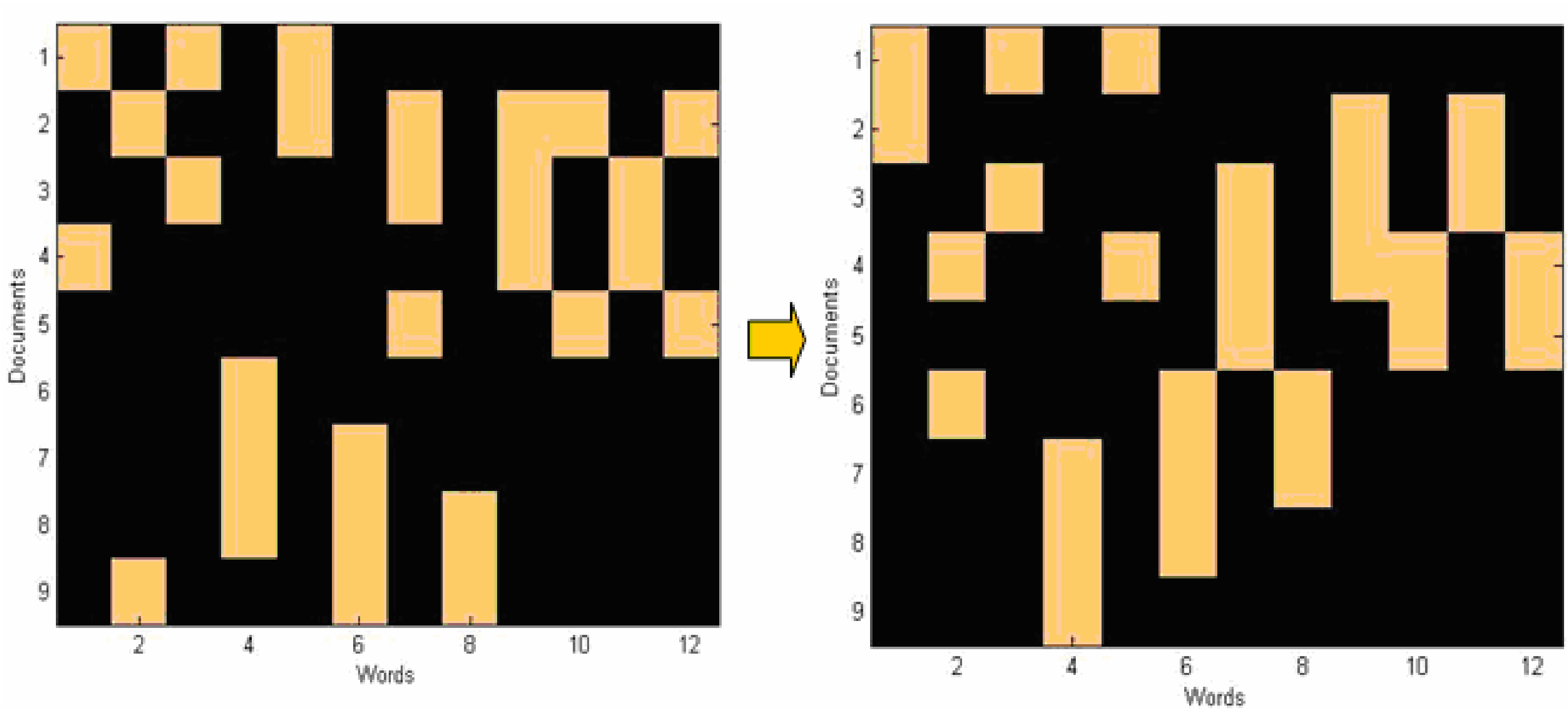
- The case most similar to the first case is stacked next to it, by swapping positions with the existing second row.
- If more than one case is found to be equally similar, one of them is chosen randomly.

Stacking : Step 3



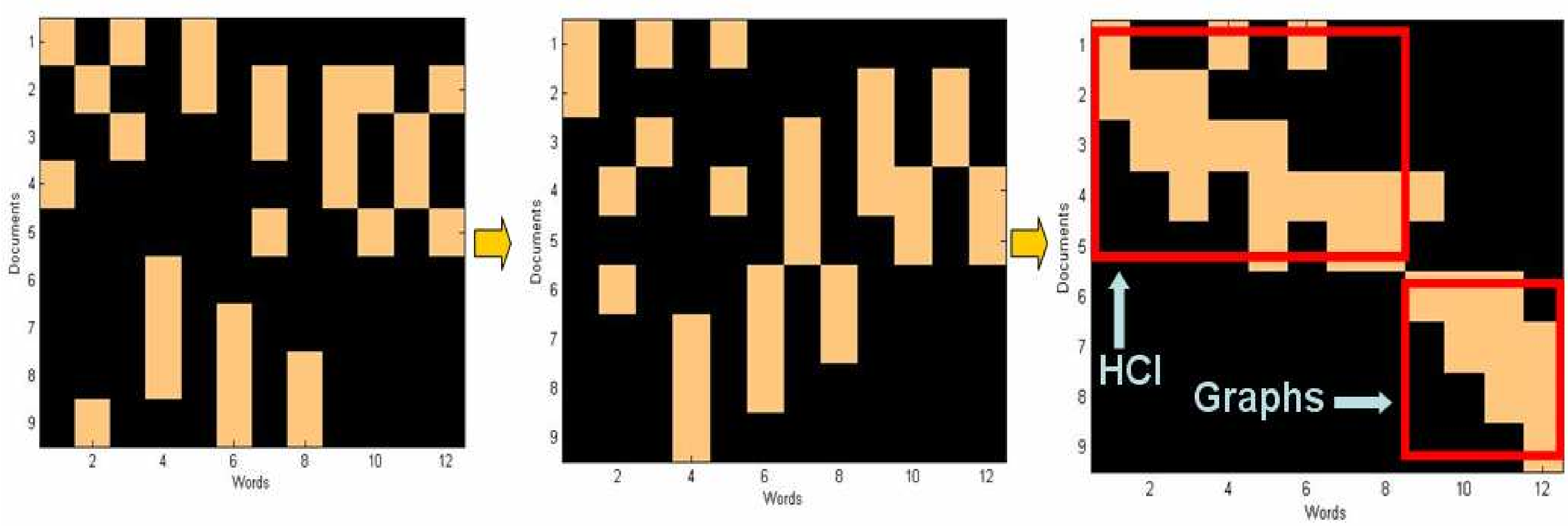
- The similarity of all non-stacked cases are calculated with respect to second case.
- The case that maximizes a weighted combination of similarities to the first and second case (higher weight assigned to the second case) is stacked next to the second row.

Stacking : Step 4



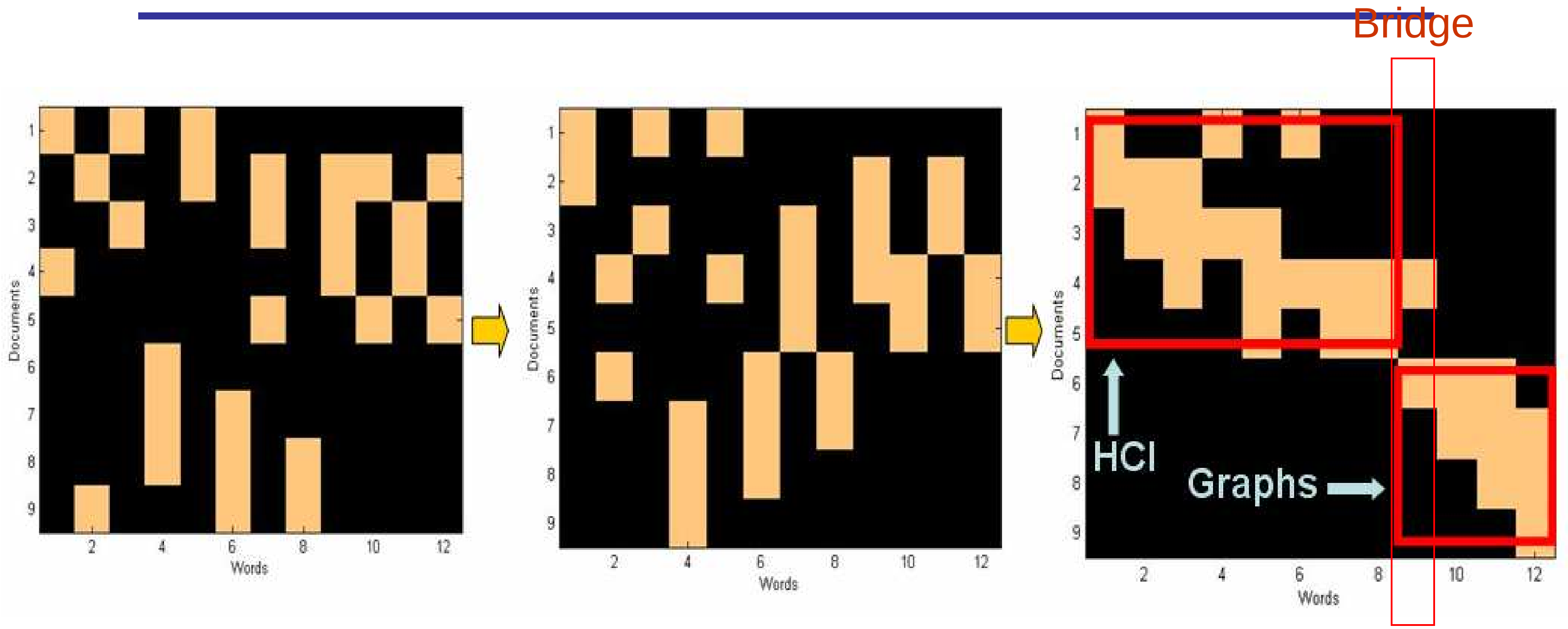
- The process is repeated till all rows are stacked.

Stacking : Step 5



- The Steps 1-4 process are repeated, this time over the columns of the row-stacked matrix generated by Step 4.

Stacking : Step 5



- Topics HCI and Graphs revealed as “chunks”
- Bridge terms shared by adjacent topics are easily identified
  - Similarly bridge cases can be identified
- Redundant features and noisy cases may also be identified
- Clustering patterns
  - Not derived by considering cases & features in isolation
  - Rather they emerge from the inter-relationship between them

Weighted Similarity Computation

- Basic Intuition: We want to ensure a gradual change in the way cases and features are grouped and displayed.
- Select the (*k*+1) row (case) that maximizes :

$$\sum_{i=1}^{k} w_i \, sim(c_i, c)$$

k = number of already stacked rows,
c_i = *i* th stacked case,
c = case being evaluated for (*k*+*1*) th position,
$sim(c_i, c)$ = cosine similarity between cases c_i and *c*
w_i = weight attached to $sim(c_i, c)$. We used: $w_i = 1/(k - i + 1)$

- Same approach applied for weighted similarity between columns
- Efficiency refinement:
  - consider only the previous 10 stacked rows or columns
  - weights associated with very distant cases are negligible

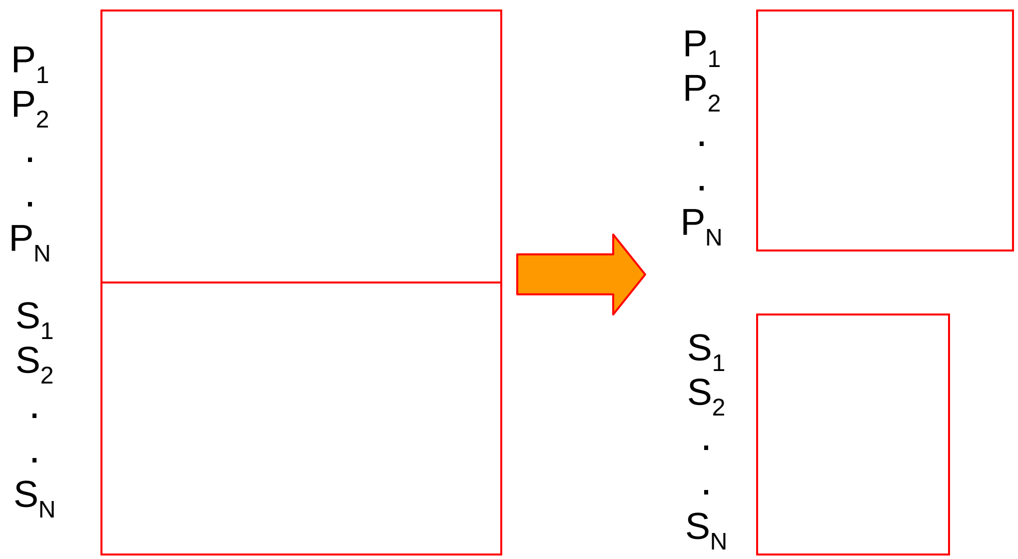
Measuring Complexity

Complexity in the TCBR context can have two interpretations :

- Collection Complexity :
  - Measures clustering tendency of the case base
  - Case base with well defined clusters has lower complexity
  - Various approaches from Text Mining & IR (e.g. Vinay ECIR06)
  - Distinction between problem and solution components is ignored

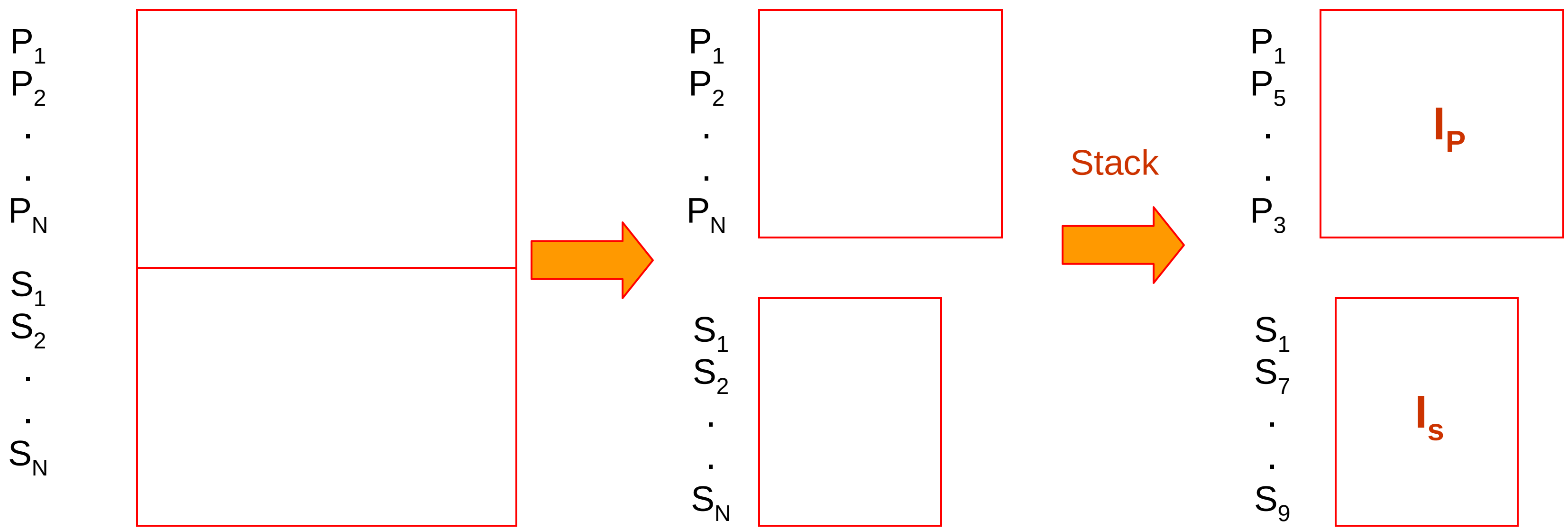
- Alignment Complexity :
  - Measures degree of alignment between problem & solution components
  - “Do similar problems have similar solutions?”
    - Local measures
      - Each case is evaluated individually (e.g. Lamontagne TCBR06)
    - Global Measures
      - Measures how well clusters derived from problem representation corresponds to clusters formed from solution representation
      - Global Alignment MEasure (GAME)

GAME : Step 1



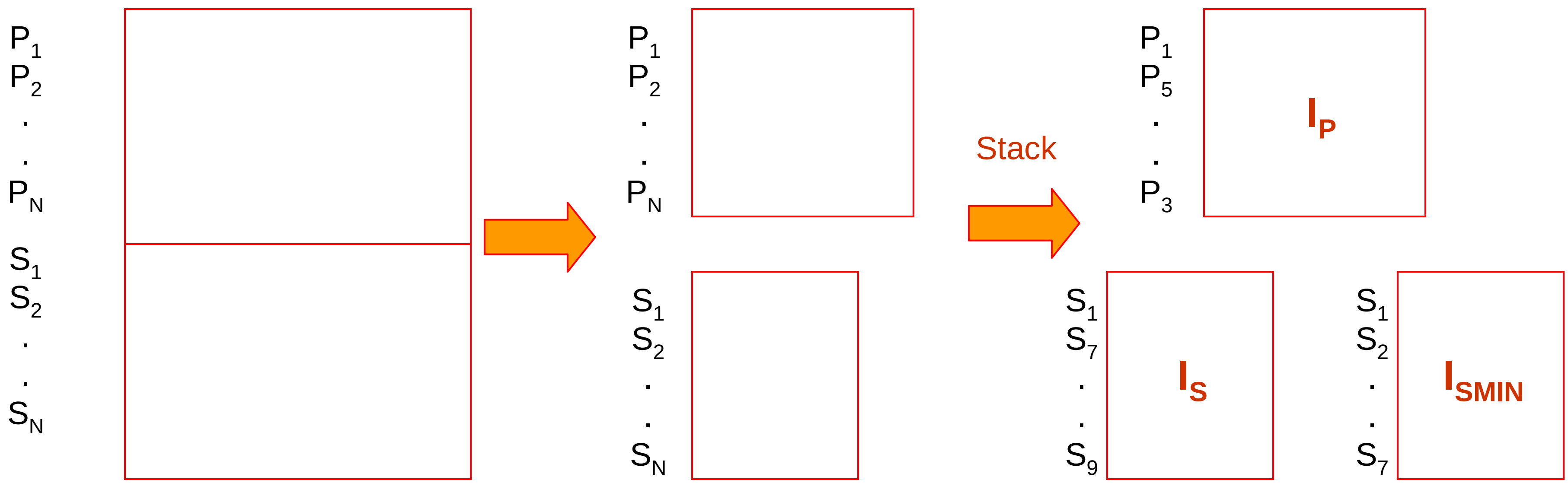
- Split representation to give separate problem and solution side case bases

GAME : Step 2



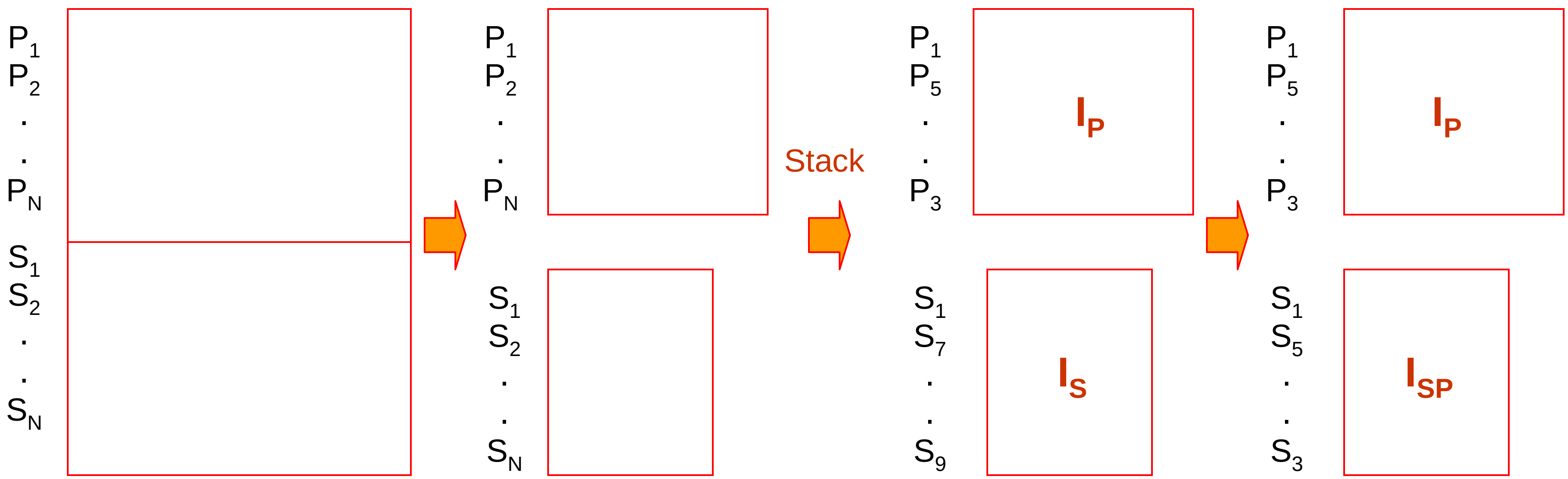
- Stack problem and solution case bases independently
  - obtain .bmp images - I_P and I_s respectively

GAME : Step 3



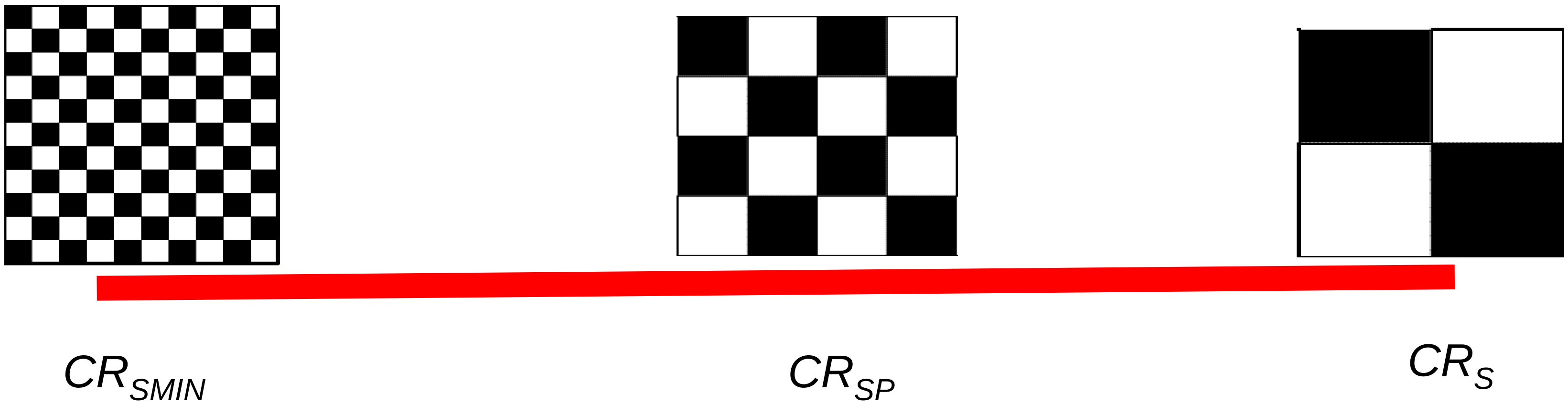
- Create image I_{SMIN} as worst layout of solution side
- Compress images I_S and I_{SMIN} by creating .png image
  - Let compression ratios be CR_S and CR_{SMIN}

GAME : Step 4



- Impose problem side ordering on solution to obtain image I_{SP}
  - ➢ Compress I_{SP} to give compression ratio CR_{SP}
- Expect $CR_{SMIN} \leq CR_{SP} \leq CR_S$

GAME : Step 5



$$GAME = \frac{CR_S - CR_{SMIN}}{CR_S - CR_{SP}}$$

- Comparing ordering of problem & solution side case bases
- High value for GAME indicates better alignment
- Low value for GAME indicates poor alignment

Extending GAME to Classification

- In Classification Datasets
  - each training case is associated with a class label
  - task is to predict the class label of an unlabelled test case
  - class labels regarded as solution vocabulary
  - simpler string based compression replaces image compression
  - do neighbours in problem side ordering belong to same class?
- Run Length Encoding
  - compression algorithm that exploits contiguous blocks
  - does not consider repeating patterns
- Adopt Similar String Compression Measure
  - count number of *flips* in solution class for a given ordering

An Example

- Binary classification problem -10 cases in the email domain
  - cases C_1 through C_5 belong to class S (for SPAM)
  - cases C_6 through C_{10} belong to class L (for LEGITIMATE)

- Assume problem side ordering of cases after stacking is $C_1C_2C_6C_4C_5C_7C_3C_9C_{10}C_8$

- Replace each identifier with class label gives string SSLSSLSLLL
  - most easily classifiable a string would be SSSSSLLLLL
  - most complex string would be SLSLSLSLSL

- Using our string compression measure
  - number of flips for problem side ordering, *flips* = 5
  - min. number of flips, $flips_{min} = 1$
  - max. number of flips, $flips_{max} = 9$

$GAME_{class}$

$$GAME_{class} = \log\left(\frac{flips_{max} - flips_{min}}{flips - flips_{min}}\right) = \log\left(\frac{(n-1) - (k-1)}{flips - (k-1)}\right)$$ **= log((9-1)/(5-1)) = 0.3**

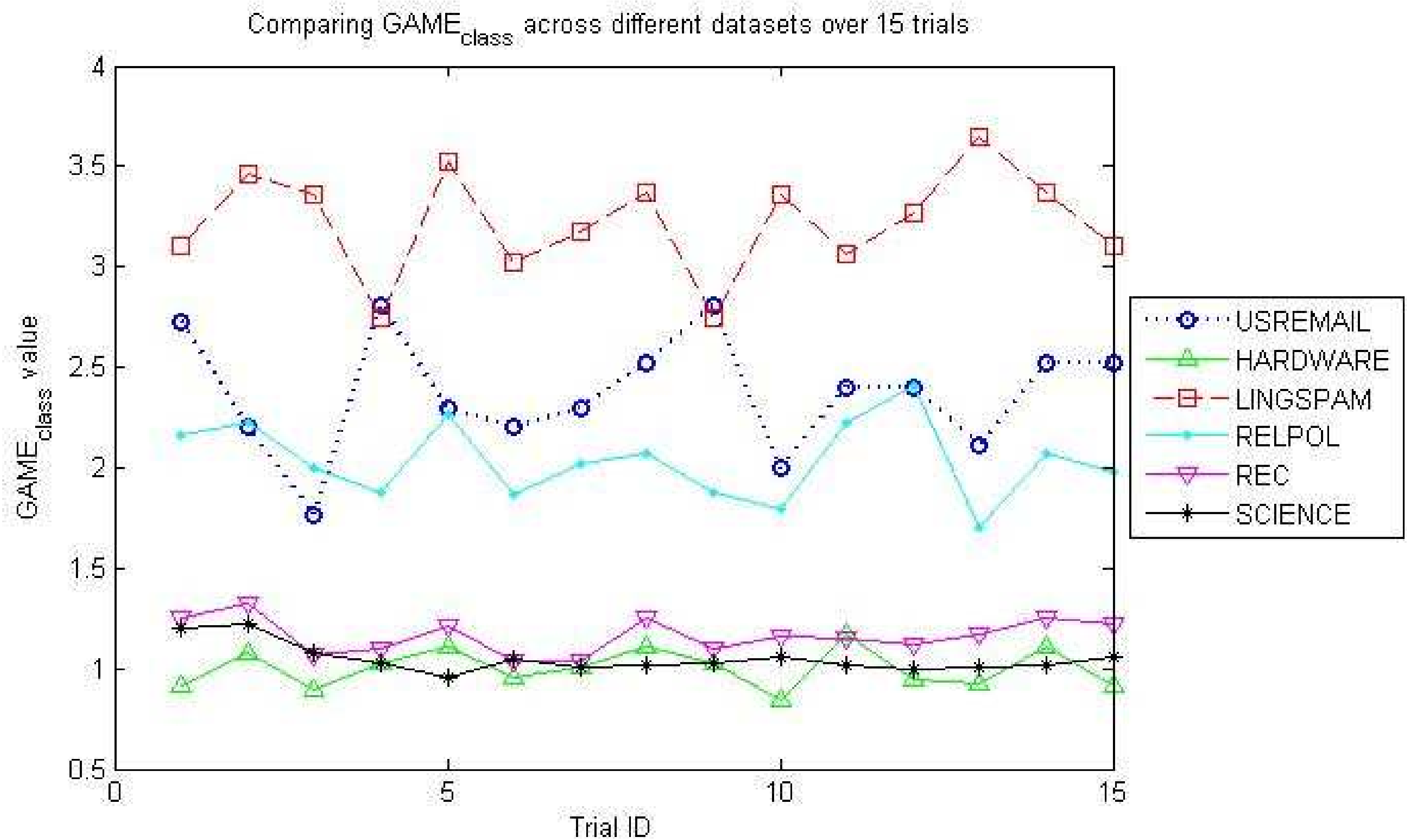
- *k* = number of classes,
- *n* = number of cases (*n* > *k*),
- *flips* = number of class transitions in problem side ordering
- $flips_{min}$ = minimum number of *flips* possible (k-1)
- $flips_{max}$ = number of *flips* for most complex case base (n-1)

- High values to well aligned dataset
- Low value equates to complex dataset
- Log introduced to reduce range

Experimental Set-Up

- Datasets were created from the 20 Newsgroups corpus
  - 1000 messages from each of the 20 newsgroups were chosen at random and partitioned by the newsgroup name
  - Four sub corpuses were formed:
    - SCIENCE which has 4 science related groups
    - REC which has 4 recreation related groups
    - HARDWARE which has 2 discussion groups on PC and MAC
    - RELPOL which has 2 groups on religion and politics
- Two datasets were used for evaluating spam filtering:
  - USREMAIL which contains 1000 emails of which 50% are spam
  - LINGSPAM which contains 2893 emails of which 83% are non-spam
- Equal sized disjoint training and test sets were created
  - Each set contains 20% of documents randomly selected from the original corpus
  - 15 such training/test splits were formed for repeated trials.

Experimental Results



Experimental Results

- Classifiers used: LSI, LSISPR, SVM, LogitBoost
- $GAME_{class}$ scores from six classification datasets
- Accuracy figures recorded by four classifiers

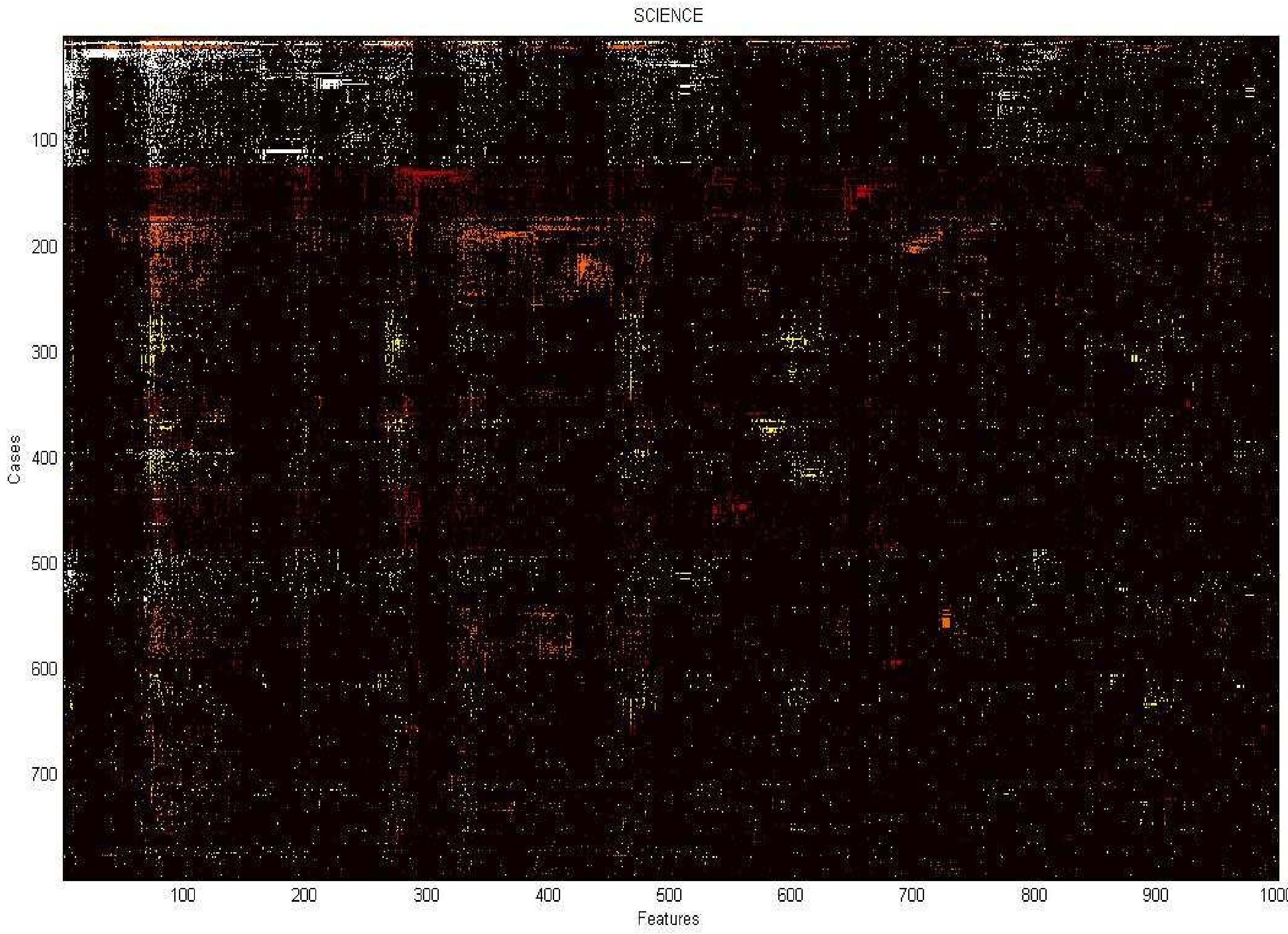
Table 1. $GAME_{class}$ and Accuracies obtained by different classifiers

	HARDWARE	RELPOL	USREMAIL	LINGSPAM	REC	SCIENCE
GAME measure	1.0028	2.0358	2.3728	3.2222	1.1629	1.0492
LSI + kNN-3	66.30	91.17	94.67	97.37	79.32	72.55
LSISPR + kNN-3	80.42	93.89	96.13	98.34	86.99	80.60
SVM	78.82	91.86	95.83	95.63	--	--
LogitBoost	77.99	79.67	92.67	95.80	87.15	73.77

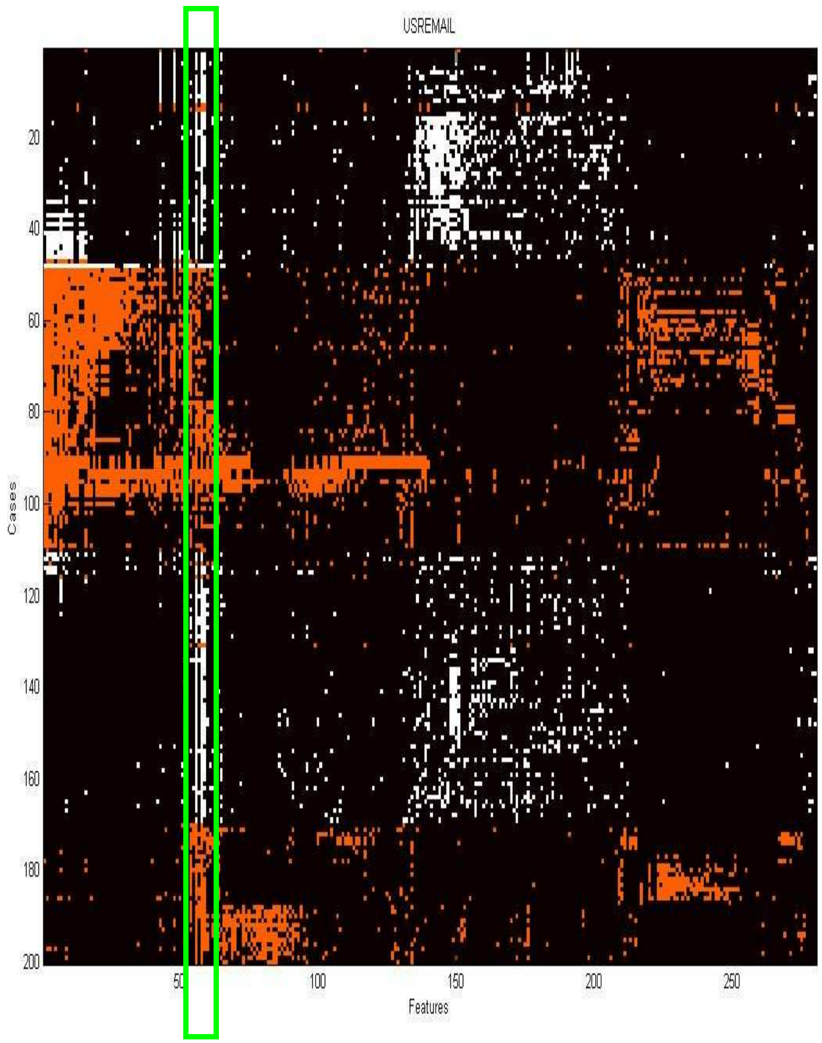
Table 2. Correlation of $GAME_{class}$ with classifier accuracies over 4 binary classification problems

	LSI + kNN-3	LSISPR + kNN-3	SVM	LogitBoost
ρ	0.9176	0.9365	0.9023	0.8820

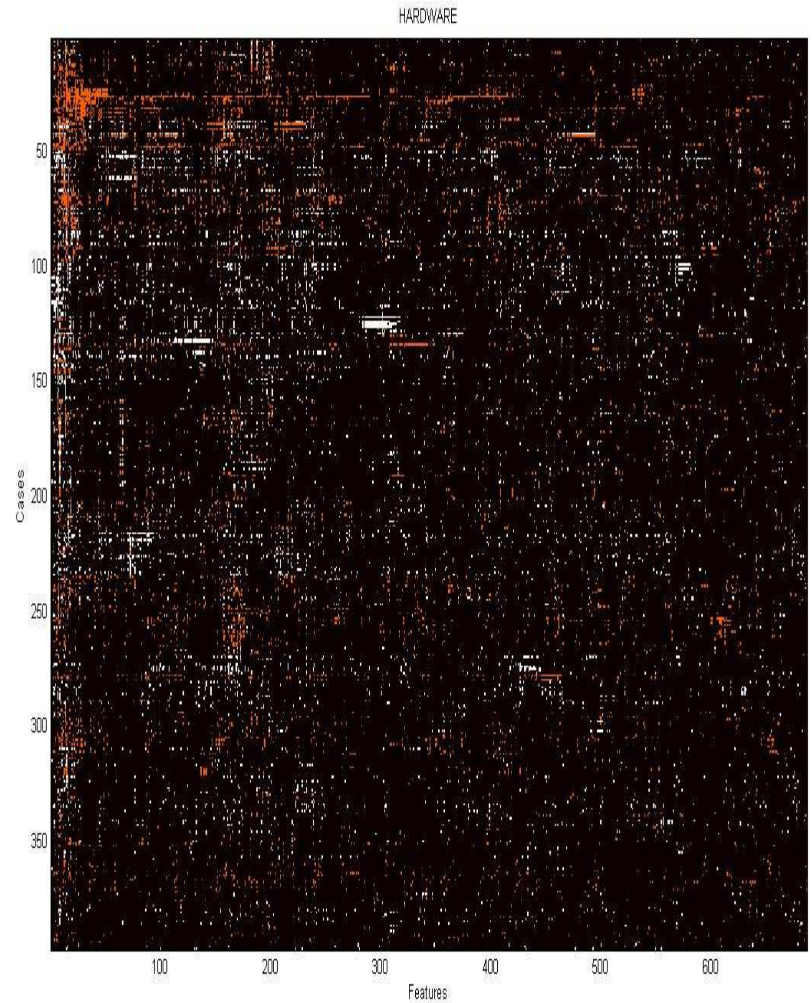
Multi-class Datasets



Comparing Datasets



USREMAIL



HARDWARE

Conclusion & Future Work

- Simple approach to visualising textual case bases
  - ➢ Shows case and feature clusters in relation to one another
  - ➢ No abstraction - helps in spotting redundant/noisy features or cases
  - ➢ Fast & simple to implement with no convergence issues and largely parameter-free
- GAME - a global complexity measure for textual case bases
  - ➢ Compares alignment of problem and solution space clusters
  - ➢ $\text{GAME}_{\text{CLASS}}$ extends the approach to supervised problems
  - ➢ Initial evaluation confirms correlation to test set accuracies
- Future Work
  - ➢ Evaluate GAME on unsupervised domains
  - ➢ Make the visualisation more interactive
  - ➢ Show word associations

QUESTIONS