Using Progressive Filtering to Deal with Information Overload

A. Addis, G. Armano, and E. Vargiu

Intelligent Agents and Soft-Computing Group

Dept. of Electrical and Electronic Engineering

University of Cagliari, Italy

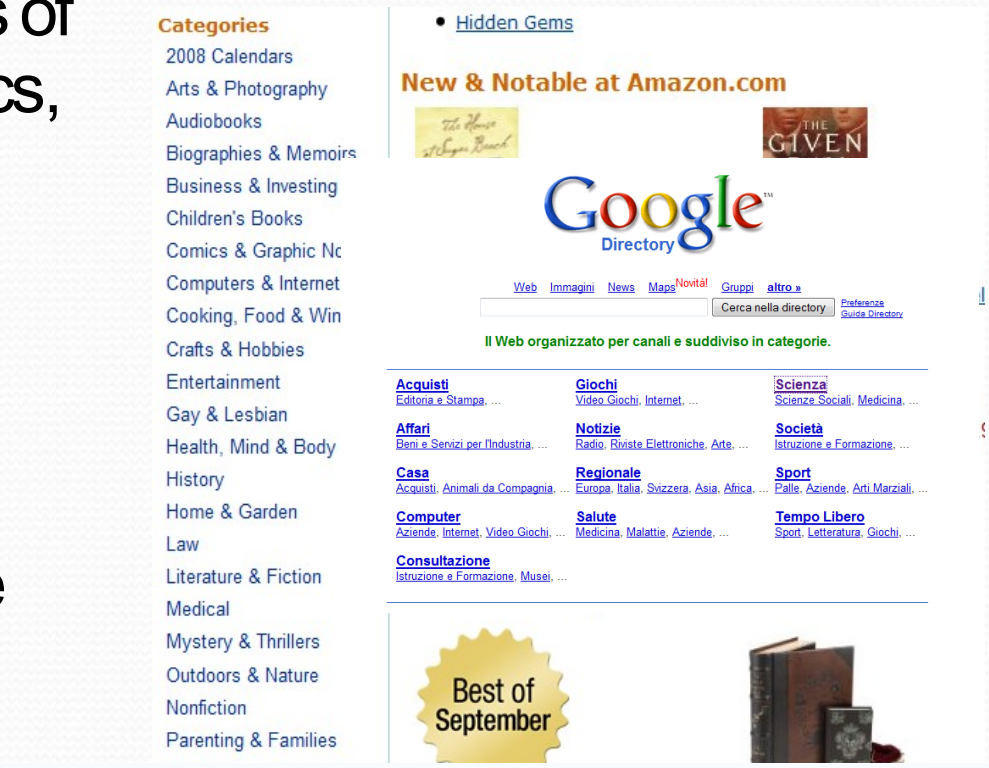
email: vargiu@diee.unica.it

Outline

- Motivations
- Progressive Filtering
- A Threshold Selection Algorithm
- Experiments and Results
- Conclusions and Future Directions

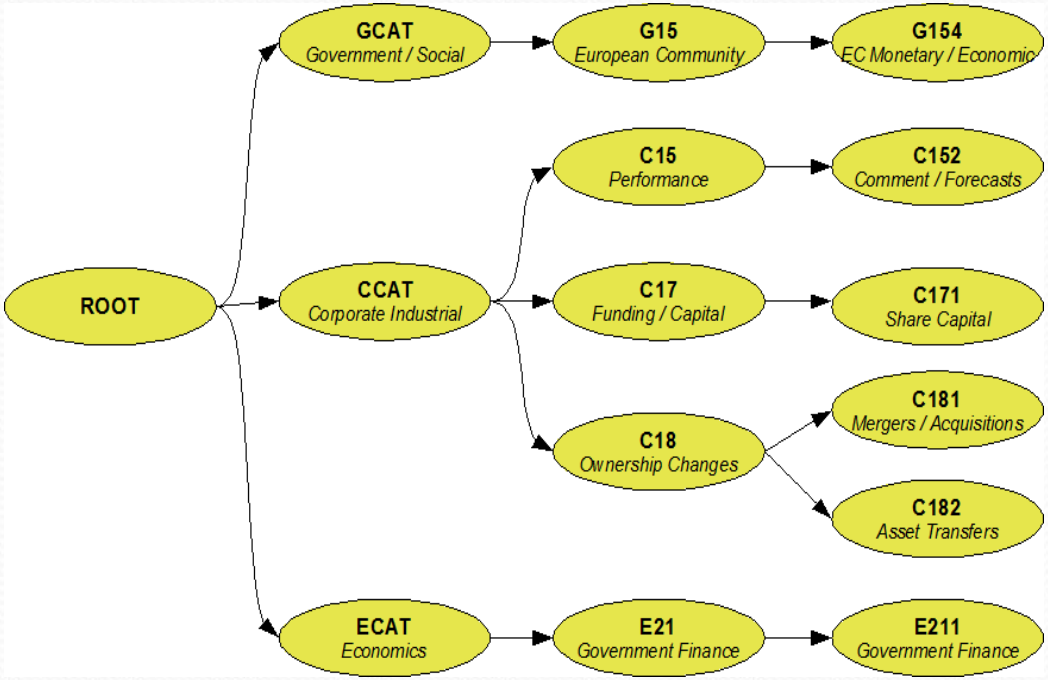
Motivations: Introduction

- People organize large collections of documents in hierarchies of topics, or arrange a large body of knowledge in ontologies
- The main goal of automatic text categorization is to deal with underlying taxonomies
- A hierarchical approach can give benefits in real-world scenarios, characterized by information overload and imbalanced data



Motivations: HTC

- Hierarchical Text Categorization (HTC) studies how to improve the performances provided by classical text categorization techniques by exploiting the knowledge of the taxonomic relationships among classes



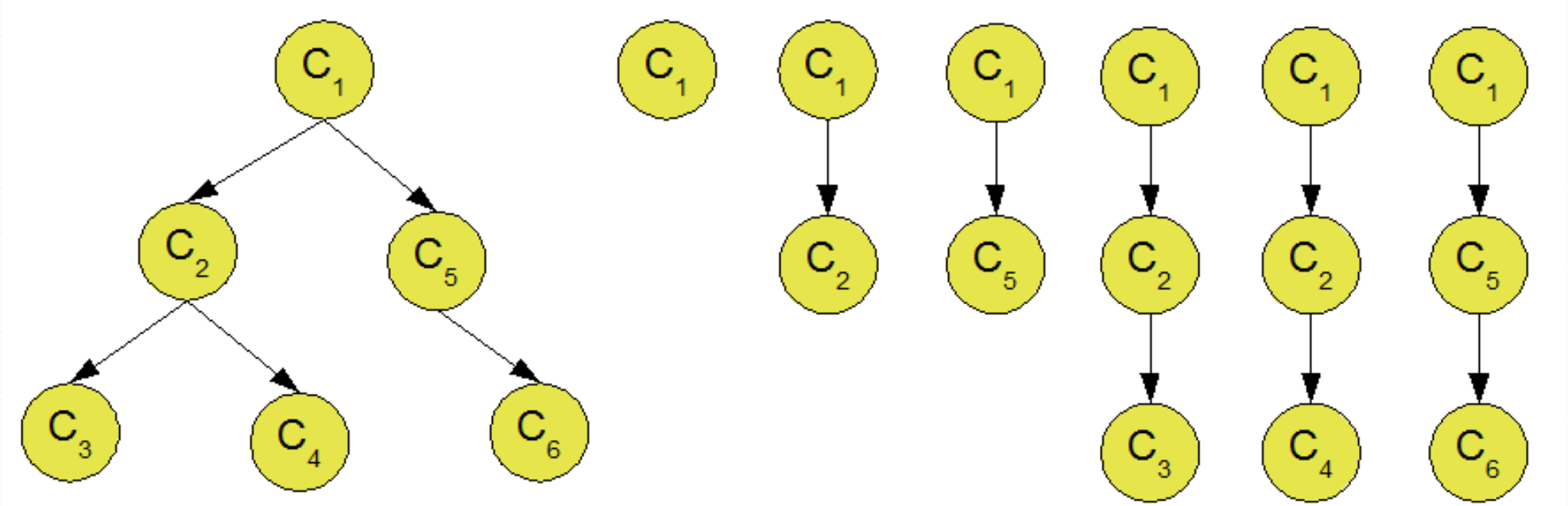
Motivations: Our Goal

- Studying how to cope with input imbalance in a hierarchical text categorization setting
- In fact, in real-world applications, an imbalance between item of interest (positive examples) vs. uninteresting items (negative examples) typically occurs according to user queries

Progressive Filtering (PF)

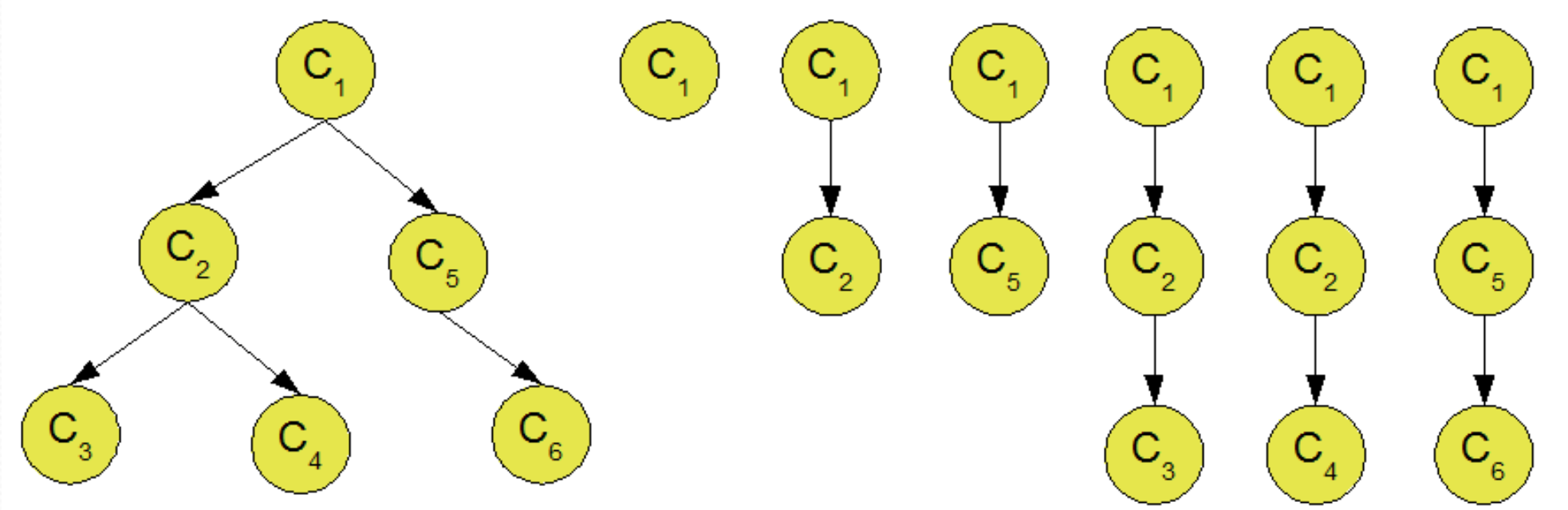
- PF decomposes a given rooted taxonomy into pipelines, one for of each path that exists between the root and each node of the taxonomy
- A threshold selection algorithm (TSA) can be run to identify an optimal, or sub-optimal, combination of thresholds for each pipeline
- Each node is a binary classifier able to recognize whether or not an input belongs to the corresponding class

Progressive Filtering (PF)



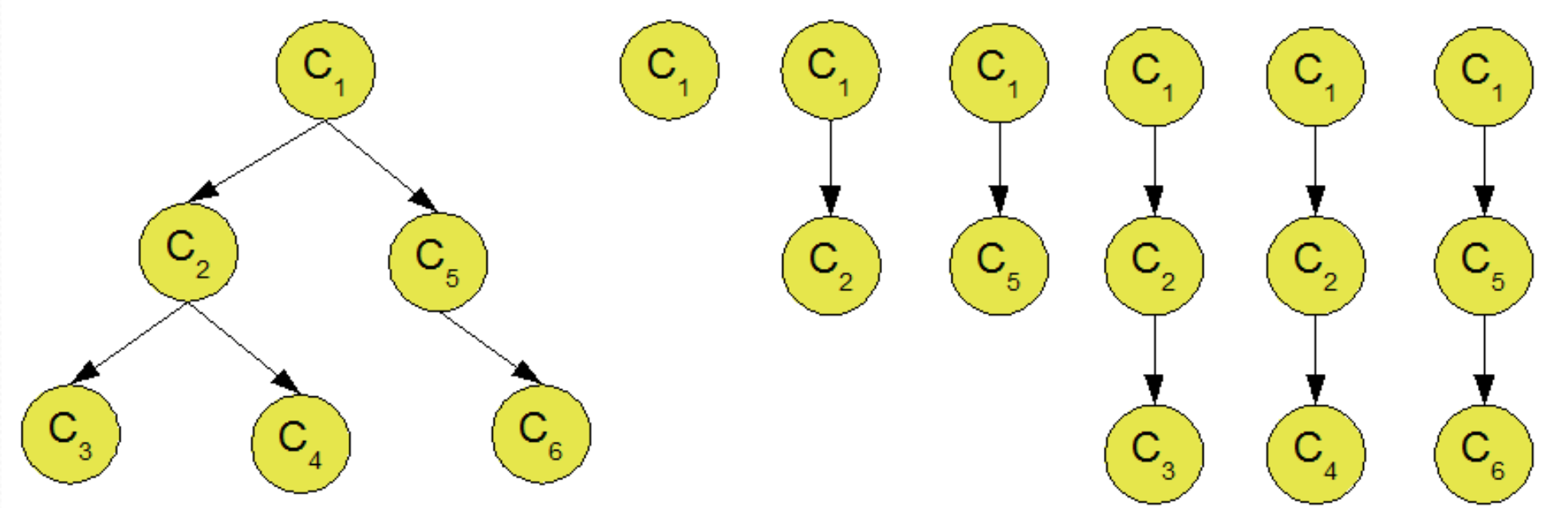
- Partitioning the taxonomy in pipelines gives rise to a set of new classifiers, each represented by a pipeline

Progressive Filtering (PF)



- Each input traverses the taxonomy as a "token", starting from the root
- A typical result consists of activating one or more branches within the taxonomy

Progressive Filtering (PF)



- The same classifier may have different behaviours, depending on which pipeline it is embedded
- Each pipeline can be considered in isolation from the others

Progressive Filtering (PF)

- A relevant problem is how to calibrate the threshold of the binary classifiers embedded by each pipeline in order to optimize the pipeline behaviour
- Searching for a optimal or sub-optimal combination of thresholds in a pipeline can be actually viewed as the problem of finding a maximum in a utility function F that depends on the corresponding threshold vector θ

The Threshold Selection Algorithm (TSA)

- For each pipeline the best combination of thresholds is calculated according to a bottom up algorithm that uses two functions
  - Repair which increases/decreases (↑/ ↓) the threshold until the utility function reaches a maximum
  - Calibrate which recursively operates downward from the given classifier by repeatedly calling repair (↑/ ↓)

The Threshold Selection Algorithm (TSA)

```
function TSA(p:pipeline):
  for k:=1 to p.length
    do p.thresholds[i] = 0
  for k:=p.length downto 1
    do Calibrate(up,p,k)
  return p.thresholds
end TSA
```

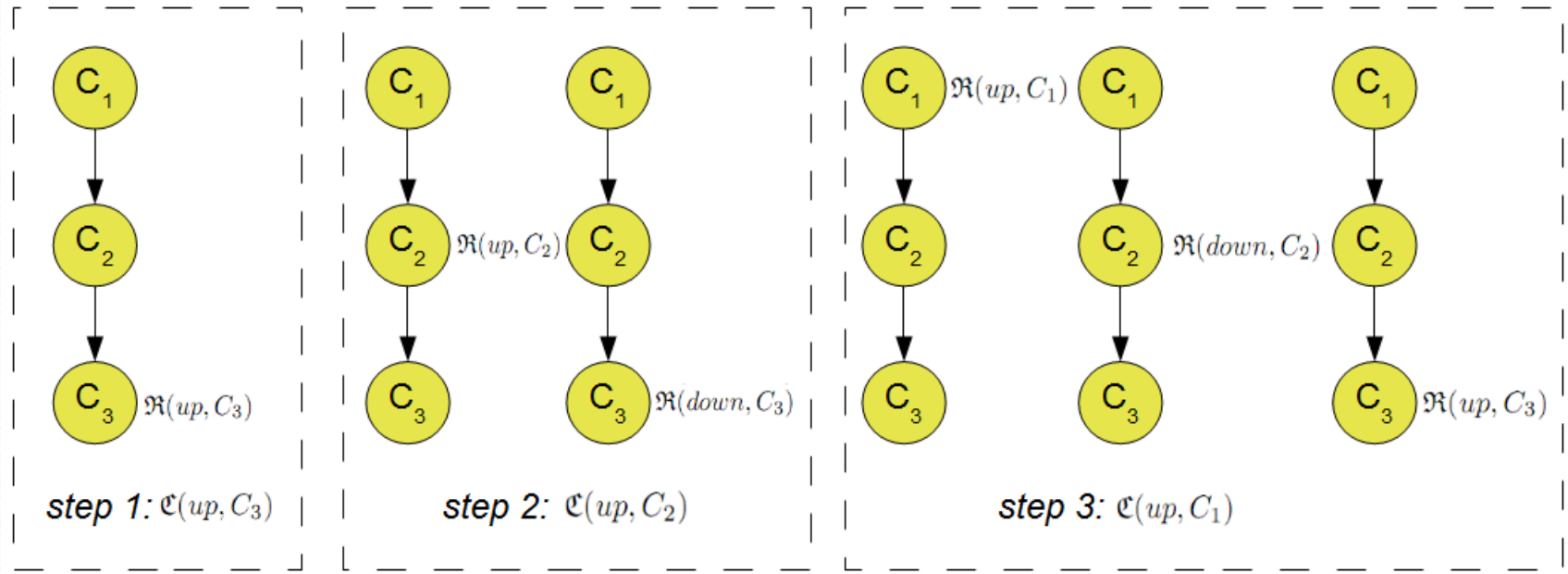
The Threshold Selection Algorithm (TSA)

```
function Calibrate(dir:{up,down}, p:pipeline,
                   level:integer):
  Repair(dir,p,level)
  if level < p.length
  then Calibrate(toggle(dir),p,level+1)
end Calibrate
```

The Threshold Selection Algorithm (TSA)

```
function Repair(dir:{up,down}, p:pipeline,
                level:integer):
  delta := (dir = up) ? p.delta : -p.delta
  best_threshold := p.thresholds[level]
  max_uf := p.utility_function()
  uf := max_uf
  while uf >= max_uf * 0.8 and
        p.thresholds[level] in [0,1]
    do p.thresholds[level] :=
       p.thresholds[level] + delta
       uf := p.utility_function()
       if uf < max_uf then continue
       max_uf := uf
       best_threshold := p.thresholds[level]
  p.thresholds[level] := best_threshold
end Repair
```

The Threshold Selection Algorithm (TSA)



R(up;C_3);

R(up;C_2) + **C**(down;C_3) =
R(up;C_2) + **R**(down;C_3);

R(up;C_1) + **C**(down;C_2) =
R(up;C_1) + **R**(down;C_2) + **C**(up;C_3) =
R(up;C_1) + **R**(down;C_2) + **R**(up;C_3);

Experiments and Results

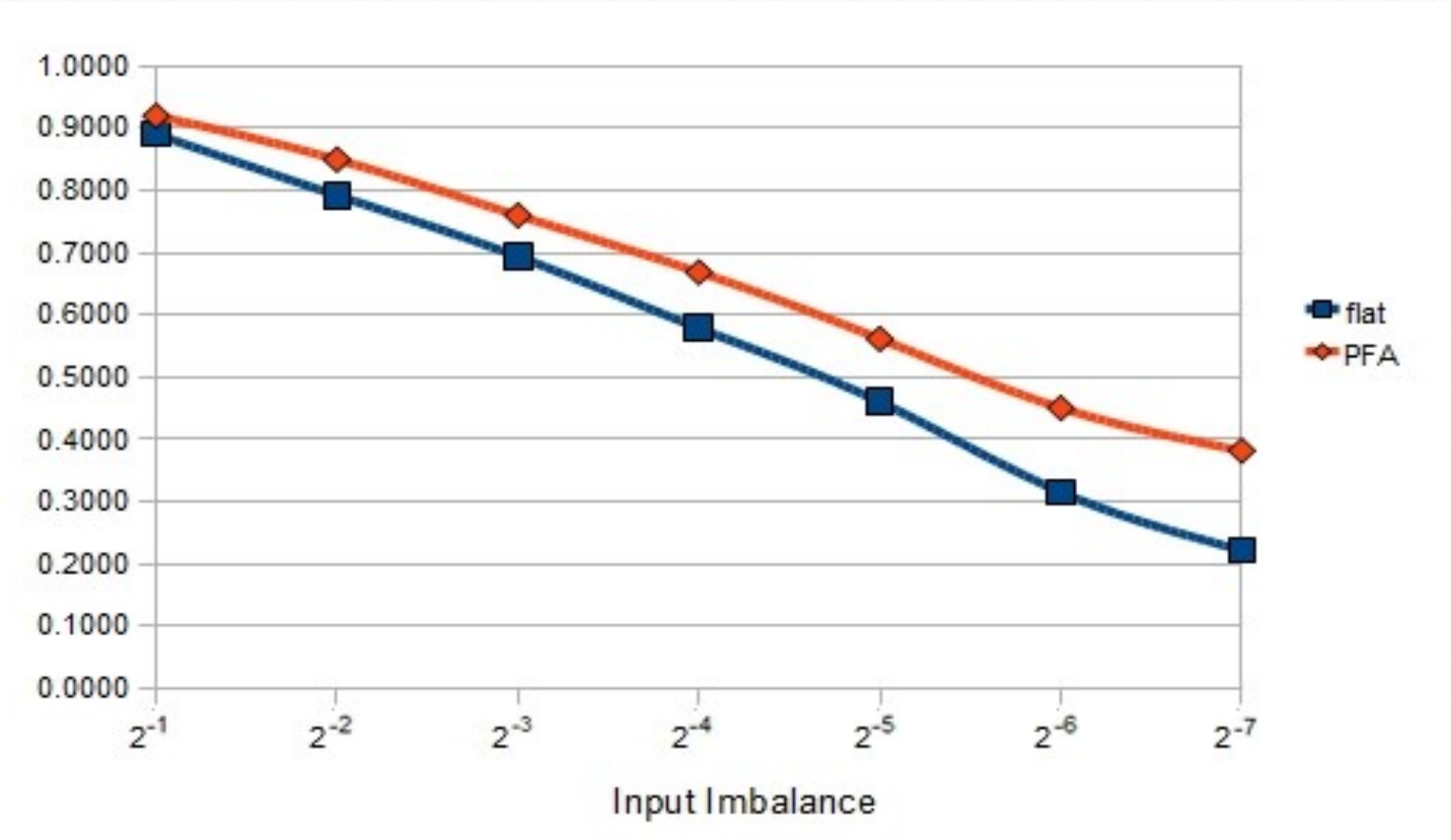
- Experiments have been performed by customizing to this specific task X.MAS a generic multiagent architecture devised to make it easier the implementation of information retrieval and information filtering applications
- Benchmark datasets
  - Reuters Corpus Volume I (RCV1-v2)
  - DMOZ
- Baseline
  - To calculate the effectiveness of the proposed approach with respect to flat classification

Experiments and Results

- Each classifier is trained with a balanced data set of 1000 documents (for Reuters) and 100 (for DMOZ) by using 200 (TFIDF) features selected resorting to information gain
- The best thresholds are selected by using F1 as utility function
- Different percentages of positive examples vs. negative examples (i.e., from 2^{-1} to 2^{-7}) have been considered
- Only pipelines that end with a leaf node of the taxonomy have been selected
- For the flat approach, only classifiers that correspond to a leaf have been selected

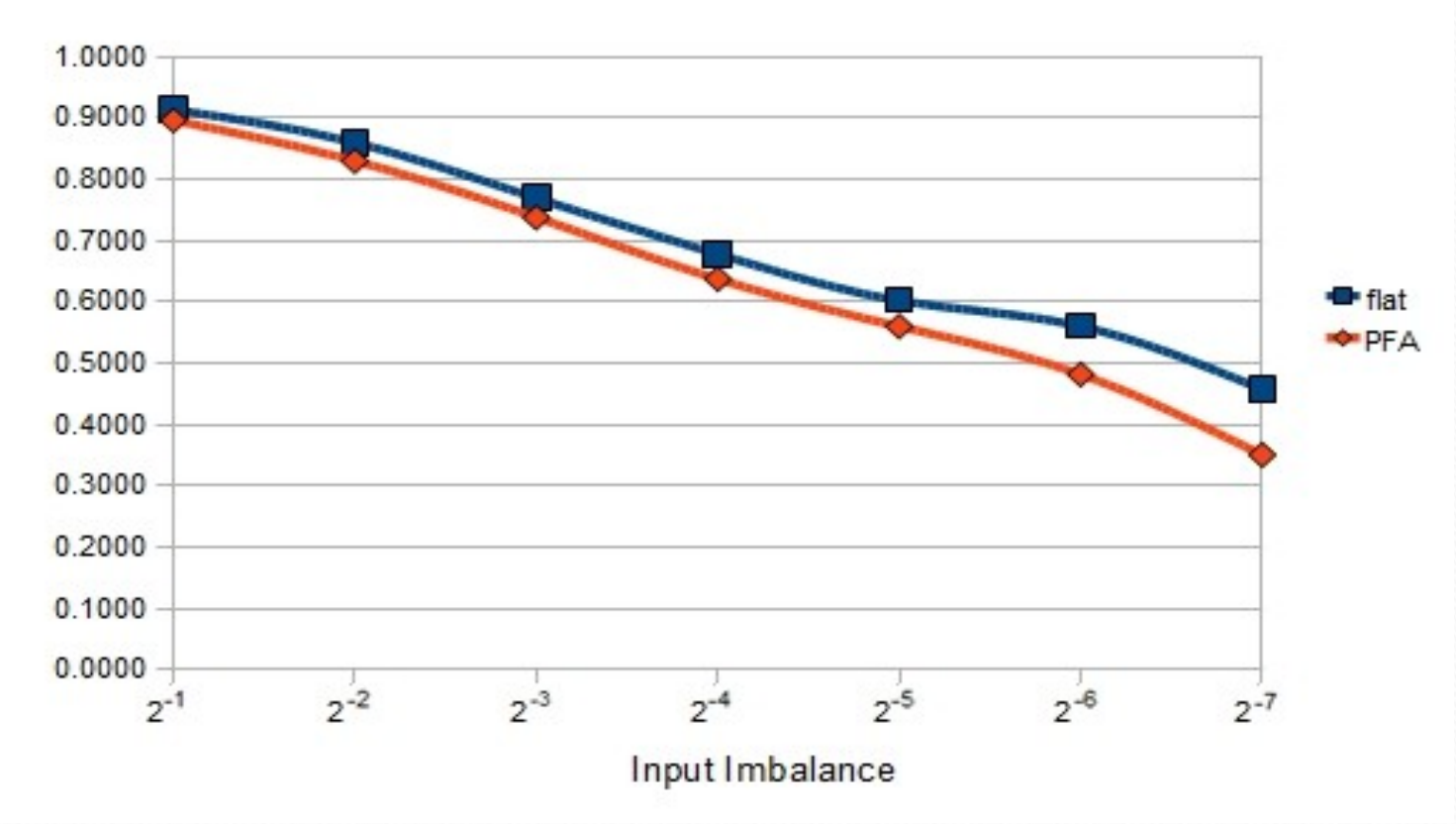
Experiments and Results

- PF vs. Flat Classification: Reuters – precision



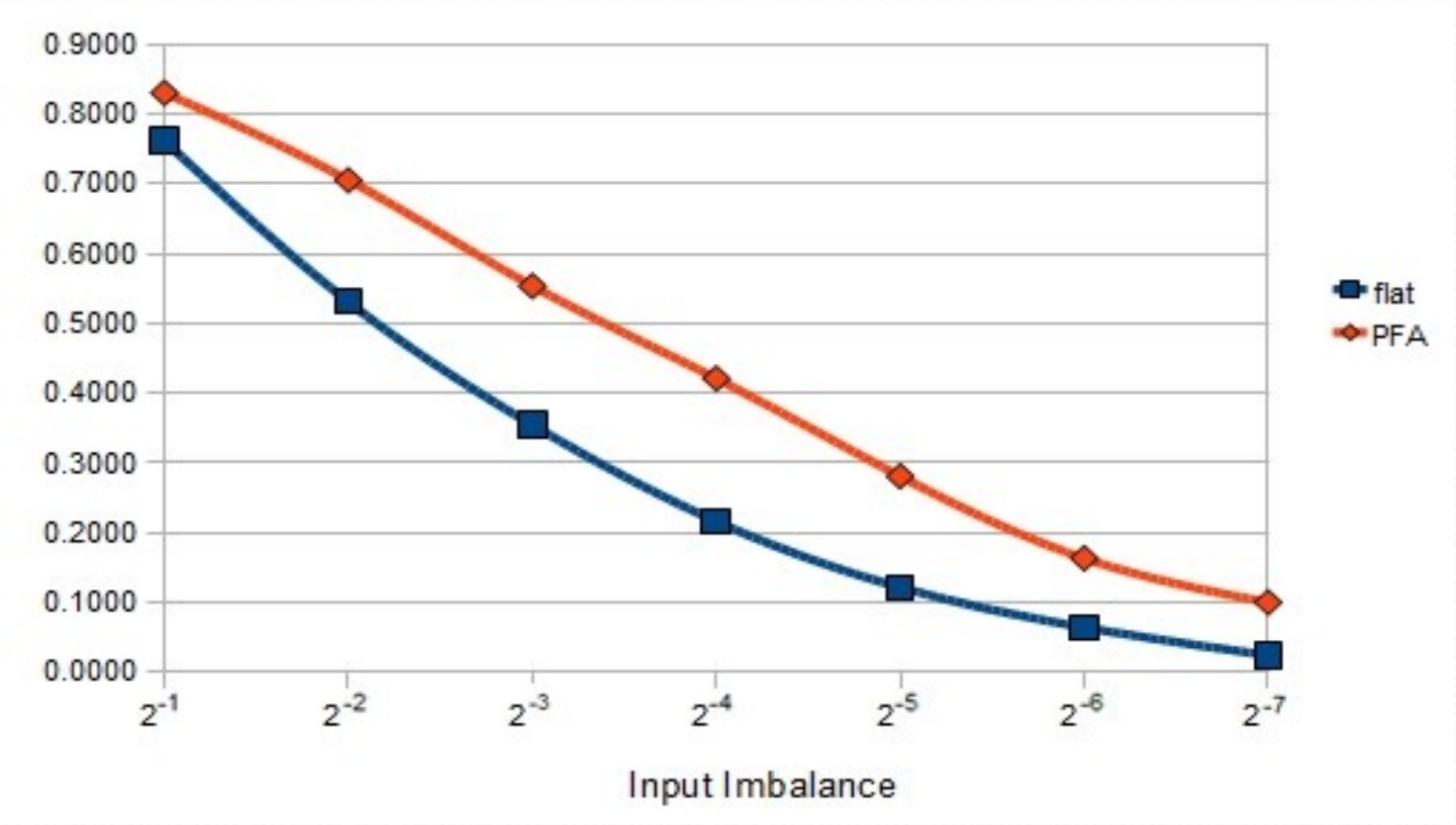
Experiments and Results

- PF vs. Flat Classification: Reuters – recall



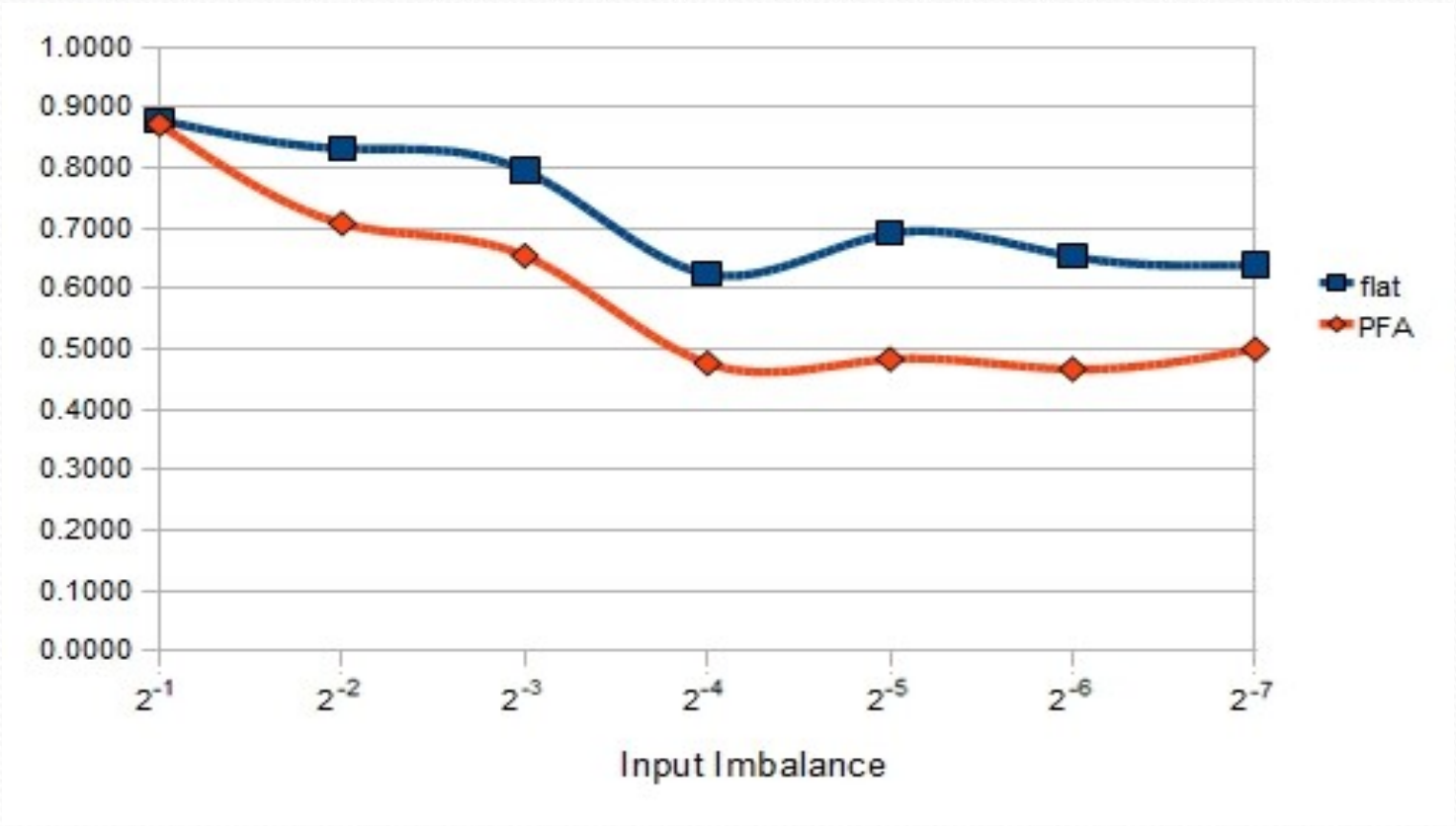
Experiments and Results

- PF vs. Flat Classification: DMOZ – precision



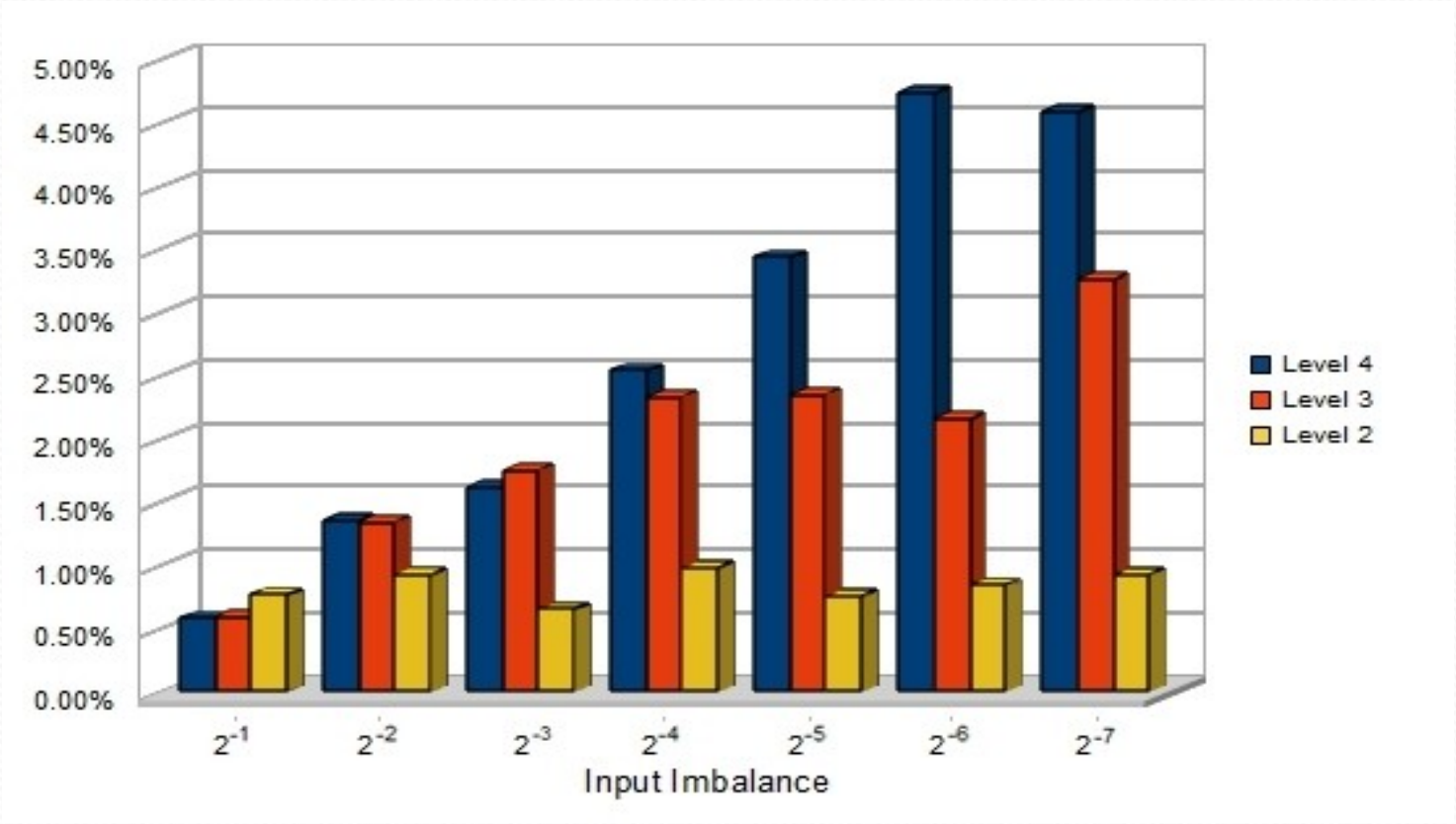
Experiments and Results

- PF vs. Flat Classification: DMOZ – recall



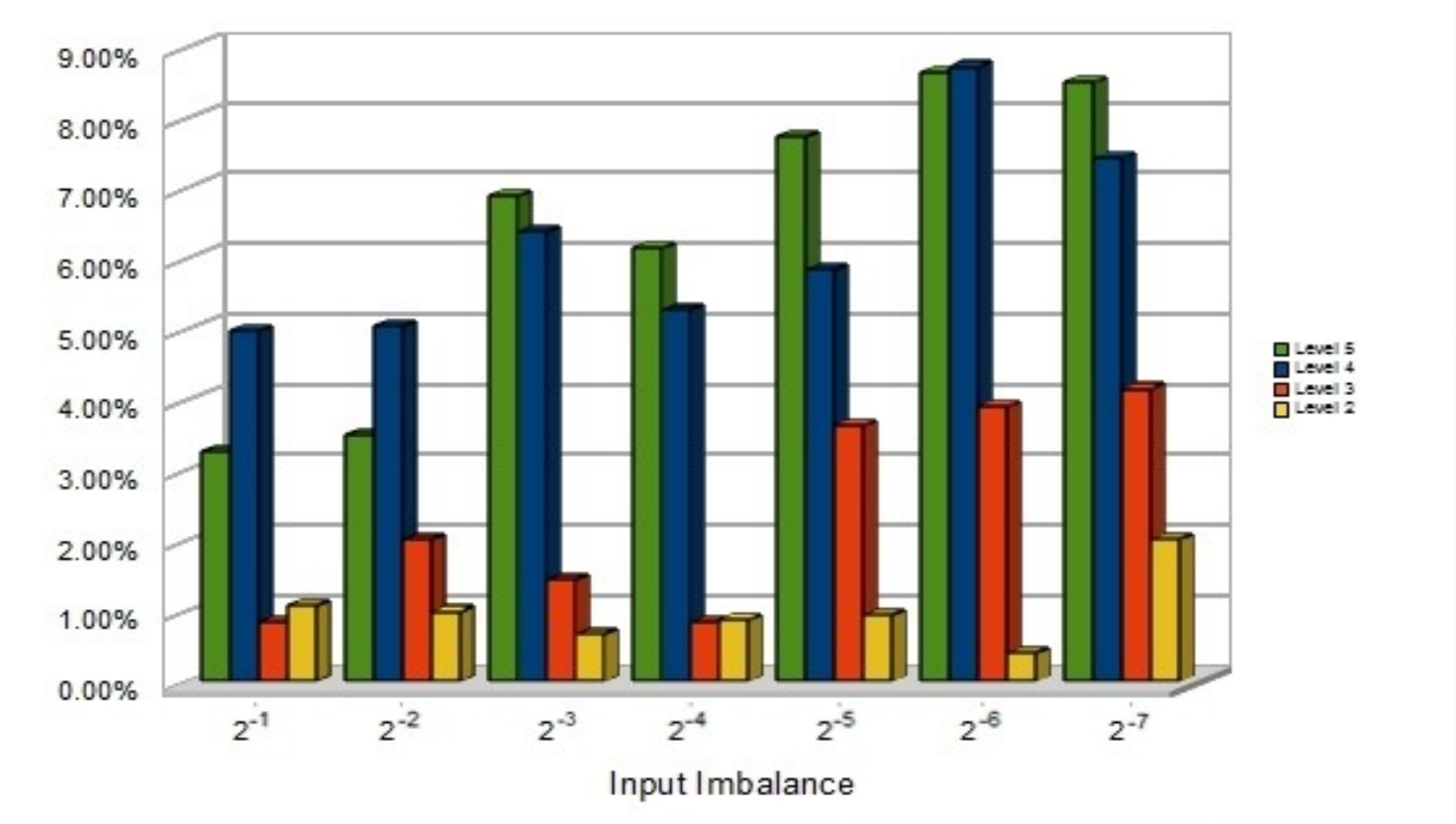
Experiments and Results

- Improving performance along the pipeline: Reuters



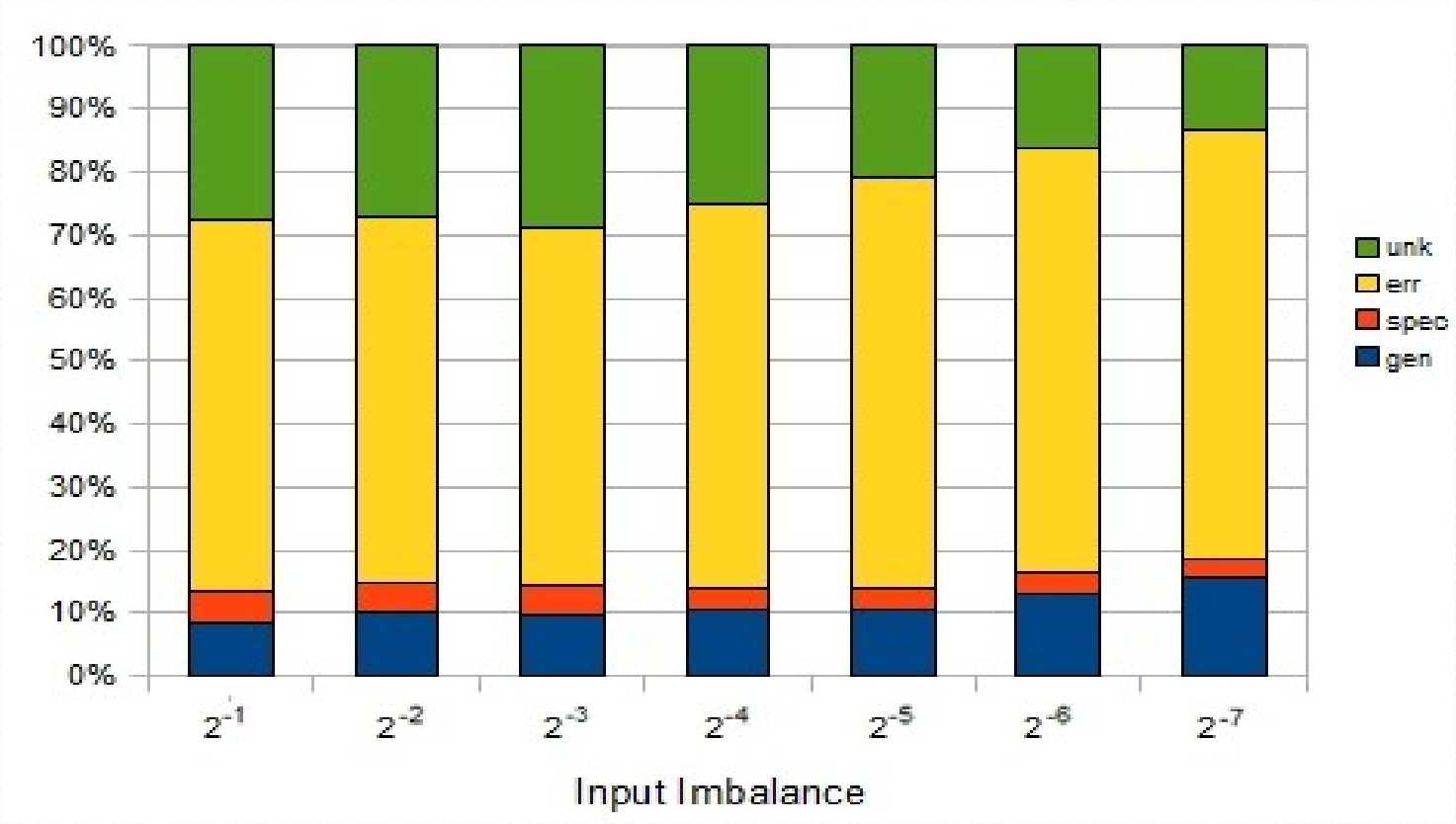
Experiments and Results

- Improving performance along the pipeline: DMOZ



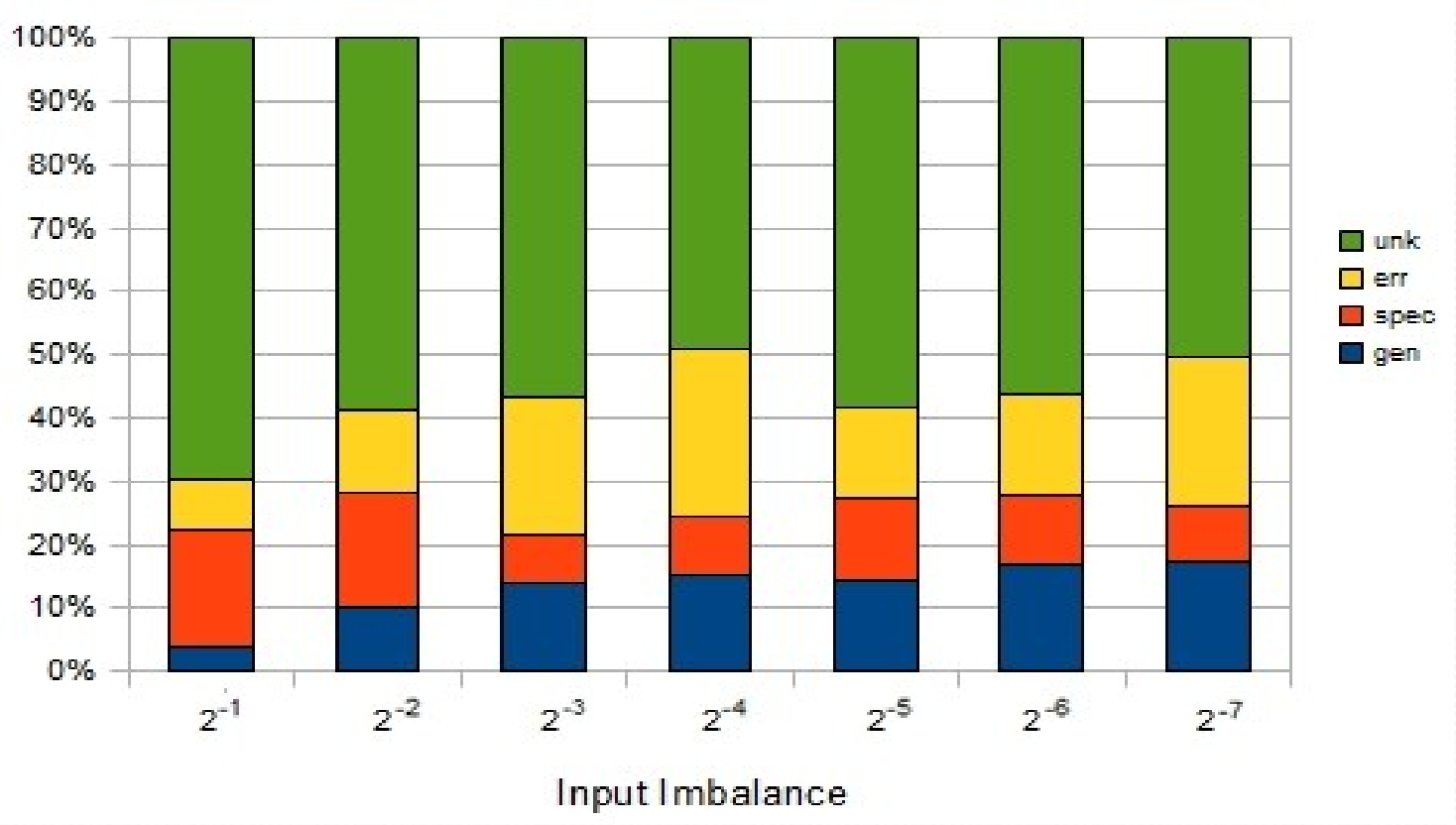
Experiments and Results

- Hierarchical Metrics: Reuters



Experiments and Results

- Hierarchical Metrics: DMOZ



Conclusions

- We studied the impact of the input imbalance that typically occurs in real-world scenarios
- PF decomposes a given rooted taxonomy into pipelines, one for each path that exists between the root and each node of the taxonomy, so that each pipeline can be studied in isolation
- Experimental results validate the assumption that the proposed approach performs better than a flat approach in presence of input imbalance

Future directions

- Performing new experiments aimed at comparing the proposed approach with state-of-the-art systems and techniques
- Investigating the whole taxonomy instead of the corresponding set of pipelines
- Adopting and calculating further metrics to assess the performances of PF
- Testing PF on further datasets, such as TREC or MeSH

Thanks for your attention!

Contact: Eloisa Vargiu vargiu@diee.unica.it