

# Making Sense Of Text

## Identifying Non Functional Requirements Early

by Amir Jalali, Rob Goff, Mark Jackson, and Nathaniel Jones



### Outline

1. What is an NFR?
2. Classification Method Originally by Jane Cleland-Huang
3. Our Experimental Results and Comparisons with Other Learners
4. Future Work
5. Conclusions

### What Is An NFR And Why Do I Care?

Non functional requirements (NFRs) refer to characteristics that can be used to judge the system's operation as opposed to functional requirements that provide specifics of how the system operates. Examples of non-functional requirements are security, look & feel, usability, and can be found in almost any business. Identifying these NFRs early in the development process ensures that they are efficiently designed into the project. NFRs can be found in almost any medium such as memos, formal specifications documents, and emails. They can exist in so many places and from so many sources that identifying which documents should go to the people that need them becomes a problem. A solution to this problem is text mining for the NFR documents requirements. The approach used for mining NFRs is not limited only to this interest, but can be applied to a wide variety of problems dealing with subject identification of text.



- **In a large Company how do you efficiently Identify NFRs across possibly thousands of unstructured documents?**
  - We are talking about an extremely large volume of documents. At which point manual classification becomes impractical.
- **How do you ensure that the identified NFR arrives to the right person?**
  - We are more worried about ensuring that the correct NFRs arrive to the correct people than people having to read a few documents that don't apply to them. After all as long as they are not reading all the documents it is an improvement.
- **How do you weed out noise?**
  - We need to separate NFRs from functional requirements (which don't require any further processing and should already be factored into the design) and other useless documents.

### A Classification Method

Originally proposed by Jane Cleland-Huang in her paper [The Detection and Classification of Non-Functional Requirements with Application to Early Aspects](#)

**Goal:** Automating the detection and classification of non-functional requirements.

Huang's work theorizes that there are relatively distinct keywords that may be used to identify a specific type of NFR document (sentences, phrases, etc).

## Current Methodology

Existing methods for requirements elicitation typically require a significant portion of human interaction and negotiation. Some research has been performed in semi-automation techniques that offer promising benefits, but all still suffer from a large user involvement overhead.

## Huang's Experiment

Huang's dataset comes from documents created during a graduate level software engineering course at DePaul University. It consists of 626 specific requirement statements belonging to one of twelve specific NFR types:



- Availability
- Look and Feel
- Legal
- Maintainability
- Operational
- Performance
- Scalability
- Usability
- Security
- Functional
- Portability
- Process

Additionally, the documents are labeled as belonging to one of the fifteen projects submitted by different student groups in the course.

Standard text mining preprocessing techniques of eliminating stop words and stemming the remaining terms are employed on the data. ([More Info](#))

## Training

The stemmed documents are mined for indicator terms (keywords) relevant to specific NFR types based on the following formula:

$$\text{Pr}_{\mathcal{D}}(t) = \frac{1}{N_{\mathcal{D}}} \sum_{i=1}^{N_{\mathcal{D}}} \frac{\text{freq}(d_{\mathcal{D}i}, t)}{|d_{\mathcal{D}i}|} \cdot \frac{N_{\mathcal{D}}(t)}{N(t)} \cdot \frac{NP_{\mathcal{D}}(t)}{NP_{\mathcal{D}}}$$

The formula assigns each term found under a specific type a score based on the following criteria:

- Frequency of the term relative to the document sizes ([More Info](#))
- Commonality of the term between NFR types ([More Info](#))
- Commonality of the term between projects ([More Info](#))

## Classification

Given an arbitrary unlabeled and preprocessed NFR, the classifier evaluates the terms in the document against the indicator term list for each type in the following formula:

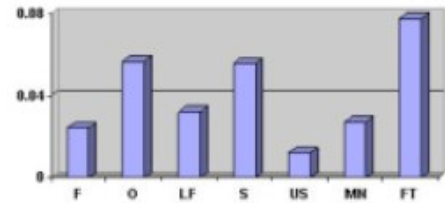
**$Pr_Q(R)$ : the Probability that a Certain NFR Document  $R$  is Classified as Type  $Q$**

$$Pr_Q(R) = \frac{\sum_{t \in R \cap I_Q} Pr_Q(t)}{\sum_{t \in I_Q} Pr_Q(t)} = \frac{\text{sum of type Q indicator term weights for all terms contained in R}}{\text{sum of all type Q indicator term weights}}$$

## Evaluation

At this point the candidate NFR has an associated classification score for each NFR type. Huang's classifier sets a threshold measure, 0.04, and assigns the NFR to each type that scores above this threshold.

The NFR Classifier uses this multi-classification to ensure that the algorithm operates in a risk-adverse environment where the person who needs to see the document will most likely see the document.

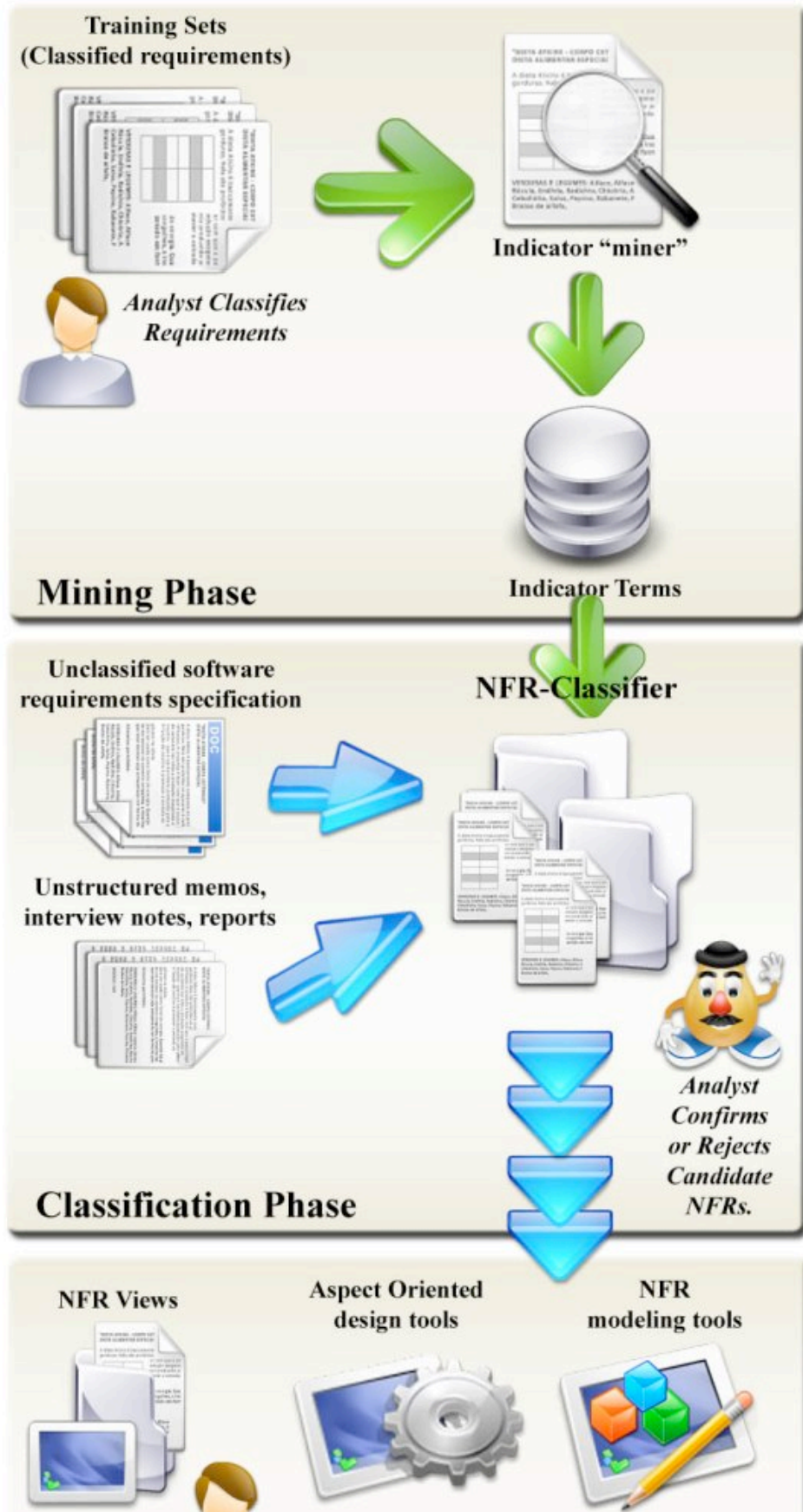


The cross-validation technique used by Huang is based on the project field of the dataset. The data, comprised of fifteen projects, is divided into a fourteen project training set and a single project test set. This division is repeated fifteen times to make every possible combination of 14/1. Each combination is sent through the classifier and the final confusion matrix from which the metrics are gained is a combination of each run's multi-classification.

### Evaluation Criteria:

- **Recall**- How many Did we get right out of all the instances.[\(More Info\)](#)
- **Precision**- How sure are we of our answers.[\(More Info\)](#)

## In Summary



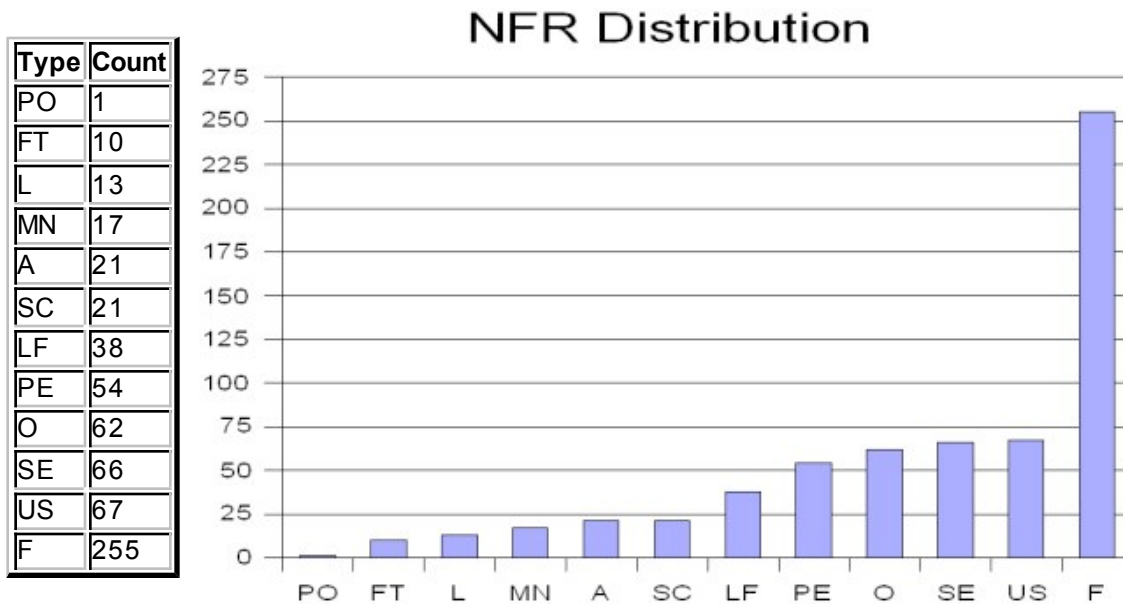
## Results

### One More Word About Pre-Processing

Out of the classifiers that we evaluate here the one that we developed to reproduce Huang's results is the only one that uses text attributes natively, so we have to use filters to convert our text attributes to numeric attributes for all the other classifiers. We do this by converting each word or the top reoccurring words to attributes and then filling in their values as their frequencies in each document or instance. We normalize this value over the length of the document. We still remove stop words and do stemming so we end up with a pretty good representation of the document in numeric form that opens our data up to many other learners.

### Experimental Data

We were lucky enough to be given access to the data used in this paper. It is predominantly made up of Functional Requirements that will need to be weeded out. It also has two classes (PO and FT) that occur such a miniscule amount of times that it is best to remove them.



## Results Of Original Experiment

Table 5. Confusion matrix of the classification

Actual	Total#	Classified as									
		A	L	LF	MN	O	PE	SC	SE	US	
Availability	18	16	0	3	6	10	10	6	4	10	
Legal	10	0	7	2	0	3	1	0	1	5	
Look-and-feel	35	0	7	18	9	20	0	1	9	22	
Maintainability	17	8	0	5	15	11	4	5	3	10	
Operational	61	10	0	32	10	44	3	11	17	42	
Performance	48	20	1	7	7	27	30	12	20	35	
Scalability	18	11	0	5	4	12	3	13	6	11	
Security	57	6	3	10	12	38	5	8	46	38	
Usability	62	13	4	26	7	34	14	24	35	61	

Table 6. Results using top-15 terms at classification threshold value of 0.04

NFR Type	Recall	Precision	Specificity
Availability	0.8889	0.2462	0.7792
Legal	0.7000	0.3684	0.9525
Look-and-feel	0.5143	0.2093	0.6907
Maintainability	0.8824	0.2459	0.8220
Operational	0.7213	0.2604	0.4151
Performance	0.6250	0.1887	0.8561
Scalability	0.7222	0.2000	0.7825
Security	0.8070	0.2771	0.6468
Usability	0.9839	0.2798	0.3447
Overall	0.7669	0.2480	

## Results Of Our Reproduction Of The Original Experiment

==== Evaluation result ====

Scheme: NFRClassifier

Relation: nfr-weka.filters.unsupervised.instance.RemoveWithValues-S0.0-Clast-L11-last-H

Correctly Classified Instances 290 85.5457 %  
 Incorrectly Classified Instances 49 14.4543 %  
 Kappa statistic 0.0636  
 Mean absolute error 0.07  
 Root mean squared error 0.1681  
 Relative absolute error 37.1585 %  
 Root relative squared error 48.1795 %  
 Total Number of Instances 339

==== Detailed Accuracy By Class ====

TP Rate	FP Rate	Recall	Precision	Specificity	F-Measure	ROC Area	Class
0	0	0	1	0	0.562	F	
0.202	0.094	0.95	0.123	0.906	0.218	0.935	A
0.15	0.023	0.545	0.15	0.977	0.235	0.709	L
0.181	0.1	0.767	0.14	0.9	0.237	0.735	LF
0.151	0.052	0.824	0.157	0.948	0.264	0.892	MN
0.21	0.164	0.933	0.212	0.836	0.346	0.919	O
0.164	0.098	0.667	0.218	0.902	0.329	0.77	PE
0.162	0.109	0.857	0.103	0.891	0.185	0.905	SC
0.208	0.124	0.902	0.258	0.876	0.401	0.873	SE
0.196	0.174	0.969	0.23	0.826	0.372	0.949	US

==== Confusion Matrix ====

```

a b c d e f g h i j <- classified as
0 0 0 0 0 0 0 0 0 0 | a = F
0 19 2 6 4 15 12 12 9 15 | b = A
0 3 6 6 3 7 2 2 4 7 | c = L
0 5 10 23 12 27 5 8 13 24 | d = LF
0 13 3 8 14 13 9 12 8 13 | e = MN
0 23 5 41 14 56 23 25 34 46 | f = O
0 35 1 15 9 30 36 24 30 39 | g = PE
0 14 2 10 7 19 12 18 12 17 | h = SC
    
```

0 21 5 16 12 46 28 32 55 50 | i = SE  
 0 21 6 39 14 51 38 41 48 63 | j = US  
 Average Precision= 0.177    Average Recall= 0.824

### Our Results

*(w/o functional results)*

Type	Accuracy	Recall	Precision
A	19	0.95	0.123
L	6	0.545	0.15
LF	23	0.767	0.14
MN	14	0.824	0.157
O	56	0.933	0.212
PE	36	0.667	0.218
SC	18	0.857	0.103
SE	55	0.902	0.258
US	63	0.969	0.23
Average	290/339=0.8546	0.824	0.177

### Original Results

*(w/o functional results)*

Type	Accuracy	Recall	Precision
A	16	0.8889	0.2462
L	7	0.7000	0.3684
LF	18	0.5143	0.2093
MN	15	0.8824	0.2459
O	44	0.7213	0.2604
PE	30	0.6250	0.1887
SC	13	0.7222	0.2000
SE	46	0.8070	0.2771
US	61	0.9839	0.2798
Average	250/326=0.7669	0.7669	0.2480

### OneR Results

*(w/o functional results and functional removed for training)*

Type	Accuracy	Recall	Precision
A	0	0	0.123
L	0	0	0.15
LF	0	0	0.14
MN	0	0	0.157
O	0	0	0.212
PE	2	0.667	0.218
SC	0	0	0.103
SE	22	1	0.258
US	0	0	0.23
Average	024/029=0.8276	0.185	0.202

### J48 Results

*(w/o functional results and functional removed for training)*

Type	Accuracy	Recall	Precision
A	16	0.762	0.41
L	5	0.385	0.385
LF	28	0.737	0.224
MN	6	0.353	0.15
O	45	0.726	0.281
PE	36	0.667	0.429
SC	9	0.429	0.225
SE	49	0.742	0.426
US	41	0.612	0.295
Average	235/359=0.6545	0.601	0.314

### Naive Bayes Results

*(w/o functional results and functional removed for training)*

Type	Accuracy	Recall	Precision
A	10	0	0.123

### J48+FSS Results

*(w/o functional results and functional removed for training) BestFirst Subset Selector and CFS Subset Evaluator*

Type	Accuracy	Recall	Precision
A	18	0.857	0.400

L	1	0	0.15
LF	22	0	0.14
MN	6	0	0.157
O	25	0	0.212
PE	22	0.667	0.218
SC	11	0	0.103
SE	37	1	0.258
US	30	0	0.23
Average	164/355=0.4620	0.431	0.397

L	5	0.385	0.135
LF	31	0.816	0.157
MN	8	0.471	0.046
O	56	0.903	0.223
PE	49	0.907	0.205
SC	14	0.667	0.093
SE	64	0.970	0.260
US	57	0.851	0.224
Average	302/359=0.8412	0.758	0.194

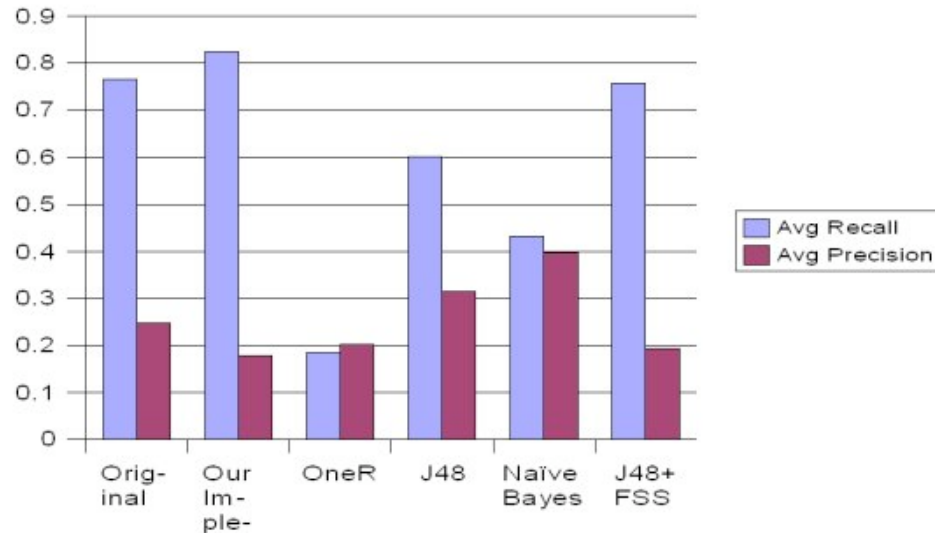
### Naive Bayes+FSS Results

*(w/o functional results and functional removed for training) BestFirst Subset Selector and CFS Subset Evaluator*

Type	Accuracy	Recall	Precision
A	13	0.65	0.448
L	13	1.0	0.433
LF	13	0.765	0.333
MN	6	0.545	0.400
O	2	0.053	0.111
PE	25	0.625	0.714
SC	8	0.533	0.276
SE	35	0.686	0.714
US	16	0.457	0.889
Average	131/240=0.5458	0.591	0.480

### Recall Vs. Precision

## Recall and Precision



## Future Work

- **Change our selection of the top performer**
  - Instead of selecting all above a certain threshold, select only top M classes
  - Experiment with different thresholding techniques
- **Change our training and testing cross validation techniques**
  - ramp the training set up from 1/15 to 14/15 to see if it plateaus
  - use boosting to incrementally increase training set until a specified percentage of documents score above a specified threshold
- **Modify our evaluation methods**
  - It is unfair to compare multi-class results with single class results.
  - What would be a good compromise?
- **Revise the data set into a new ARFF file that can be ran on other learners**
  - run it through classifier then make 15 attributes for each class score of the document
- **Order the class documents based on the highest scoring values and see if they qualitatively 'seem' more important**
- **Apply our theories to other data**
  - Data from Jane Huang's "Speech Detection of Stakeholders' Non-Functional Requirements"
- **Clustering the data and then classifying**
- **Feature subset selection**

## Conclusion

The idea of automating the identification of NFRs seems to be very feasible and very useful. It seems like a higher precision would be desirable and based on our results we feel that a more complicated classifier used in conjunction with Huang's indicator term mining techniques might be able to outperform just simple statistical analysis. More work needs to be done on distinguishing between functional and non-functional requirements.

*Data Maturity Model>>*