# An Automatic Evaluation Framework for Improving a Configurable Text Summarizer

Loïs Rigouste[†], Stan Szpakowicz[⋆], Nathalie Japkowicz[⋆], Terry Copeck[⋆]

[⋆] School of Information Technology and Engineering
University of Ottawa
800 King Edward Avenue, P.O. Box 450, Stn. A
Ottawa, Ontario, K1N 6N5 Canada
(szpak, nat, terry)@site.uottawa.ca

[†] Département Traitement du Signal et des Images
École Nationale Supérieure des Télécommunications
46 rue Barrault
F-75634 Paris Cedex 13 France
rigouste@enst.fr

**Abstract.** CALLISTO is a text summarizer that fits its flexible module configuration to the text summarized, based on training that uses Machine Learning. That is, different texts can be summarized in different ways. Because CALLISTO crucially depends on learning, it is sensitive to biases established in advance that may not be wholly appropriate. We set out to test whether other biases, modifying the space that CALLISTO explores, lead to improvements in the overall quality of the summaries produced. We present an automatic evaluation framework that relies on a summary quality measure proposed by Lin and Hovy. It appears to be the first evaluation of a text summarization system conducted automatically on a large corpus of news stories. We show the practicality of our methodology on a few experiments with the Machine Learning module of CALLISTO. We conclude that, while the evaluation framework does not address all the issues, it gives reliable hints on the adequacy of a bias. A baseline and a statistical significance threshold can be easily computed and help interpret the results. This suggests that our framework could be useful in developing automatic text summarization systems that work with Machine Learning techniques.

## 1 Introduction

The configurable text summarization system CALLISTO [Copeck *et al.*, 2002] variously combines multiple realizations of several modules: text segmenters and key phrase extractors. There is a large space of configuration options. We use Machine Learning to find which settings of these options tend to generate good summaries. Training consists in generalizing a large set of raw summaries into rules that determine the best configuration given a document's properties. The

volume of the training data means that we need a reasonably trustworthy automated measure of the quality of a summary.

CALLISTO has many degrees of freedom and many biases that suggest research questions. These biases include the choice of the Machine Learning module. In the initial design we settled on C5.0, partly because it is widely used and generally considered reliable. Another bias is the specific measure of summary quality. We also selected easily computed text features to characterize a document for summarization.

Some of the initial design choices in CALLISTO were reasoned but arbitrary. The range of options was relatively narrow. For example, we had at our disposal three public-domain text segmenters, and three key phrase extractors. It is impossible to claim with confidence that any design decisions are better than others, let alone optimal. We participated twice in the Document Understanding Conference [DUC, 2001 and 2002], an annual evaluation exercise sponsored by the National Institute of Standards and Technology. While the feedback was very helpful, the complexity and timing of DUC events allowed us to evaluate only one fixed set of CALLISTO configurations each year. We could not, for example, try another Machine Learner. We need an easier way of gauging the effects of our design decisions, that is, trying out other experimental configurations of CALLISTO. We describe an evaluation framework that attempts to meet this need by allowing us to compare and choose between different settings of system parameters. This framework could also fit any other text summarizer based on Machine Learning techniques.

We first look at CALLISTO, and at the choices it requires. Next, we review the relevant literature on evaluation in automatic text summarization. We then explain how we used our findings to design an evaluation framework that automatically judges how well a given version of CALLISTO performs. The results of a few preliminary experiments support our expectation that such a framework helps CALLISTO improve the overall quality of summaries.

## 2 Evaluation and the CALLISTO System

### 2.1 CALLISTO

Automatic text summarizers either generate an abstract from some semantic representation of the document, or extract relevant document fragments, usually sentences. Many summarizers constructed by the research community to date are sentence extractors. In particular, some form of sentence extraction, optionally combined with postprocessing, underlies many systems that participate in DUC. CALLISTO belongs in this category.

Our fundamental premise is that summarization can be achieved by combining in various ways the techniques of segmentation and key phrase extraction. Segmentation divides a document into semantically coherent groups of sentences by identifying topic shifts. Key phrase extraction retrieves the most relevant words or phrases that characterize a document.

Suppose that a text has been segmented and a set of key phrases found in it. We rank the segments according to their density of key phrases – this in a way characterizes their informativeness – and proceed to extract sentences. In the decreasing order of key phrase density, we select in each segment the sentences containing the highest number of key phrases above a pre-set threshold, stopping when we have selected the number of sentences or words desired for the final summary. Figure 1 sums up this process.

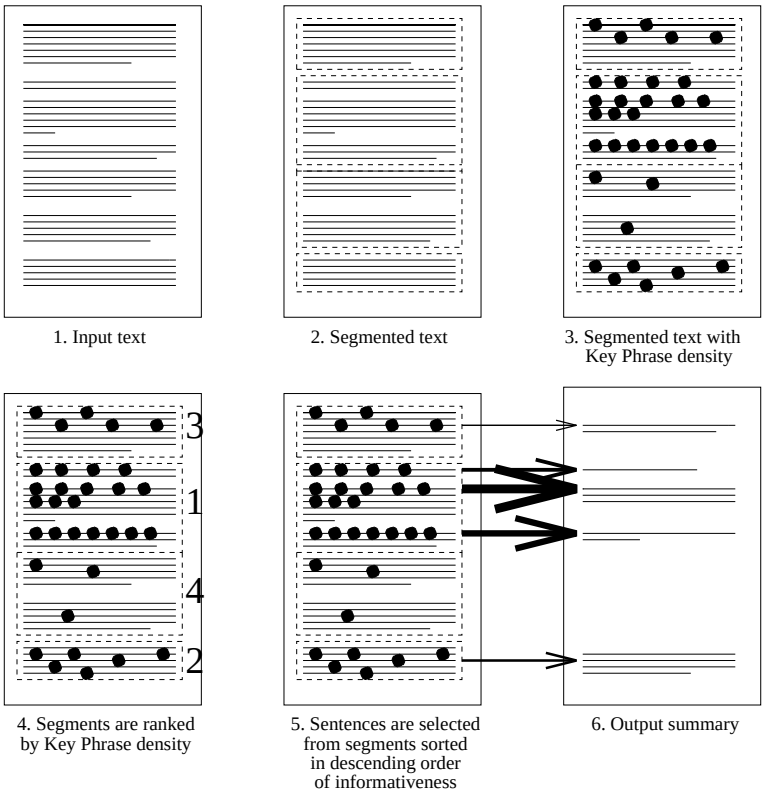


**Fig. 1.** Sentence extraction by segmentation and key phrase extraction.

While these steps produce a summary no matter which tools (segmenters and key phrase extractors) are chosen, we do not know how appropriate the summary is. We have access to several such tools, mostly in the public domaim. We assume that Machine Learning – Decision Tree induction with C5.0 [Quinlan, 1997] – helps find the best segmenter / extractor combination for every text. We calculate document features (the length, the number of content words, various N-gram counts and character/word/sentence statistics) and assume that these features along with the configuration parameters will enable the machine

learner to find patterns in the training data (documents with manually prepared model summaries).

Automatic summarization invites automatic evaluation. It is time-consuming but feasible to apply each summarizer configuration to all documents in the training set, but it is unrealistic to ask a judge to rate all summaries[1]. The application of supervised learning to text summarization is paradoxical: never has the need for quick and reliable evaluation methods been so pressing, yet the only trustworthy methods require human judgment at some point. We address this issue by having CALLISTO rate each summary it produces by computing a score based on Key Phrases in Abstract (KPiAs) – see section 4.

The KPiA method rates a generated summary by matching it against sequences of content words in the model summary (words that appear between stop words). While not too accurate, this scoring method rewards summaries that employ terms that authors use in their own summaries. We presume that multiple occurrences of a KPiA are less informative than one occurrence each of different KPiAs. KPiA evaluation is one of the design choices we want to challenge. Though it has the obvious advantages of being automatic, easily computed and reasonably fast, it is not flawless, as we will explain when we discuss an alternative evaluation method.

The measure gives a score between 0 and 1, while C5.0 classifies cases in terms of a discrete set of values, enumerated or computed. The score must be discretized for training to proceed. We should choose a discretization method to suit the character of our data. We chose the K-means procedure [MacQueen, 1967]. In preliminary experiments, 5 classes seemed to give good results, so we used the 5-means algorithm for DUC 2001 and 2002. This bias is also disputable; the number of classes is among the parameters we study in the experiment section.

Fig. 2 gives an overview of CALLISTO, including the ML module.

### 2.2 Does CALLISTO Produce Good Summaries?

Although C5.0 helps determine which segmenter and key phrase extractor is best suited to a given text, the way in which these tools are combined and data handled relies on our intuition. We still need a systematic way of evaluating the final quality of the summaries produced. It would also help to see how well CALLISTO performs when parameter values change. As already noted, CALLISTO produces such volume of intermediate data that we can only succeed by building a fully automated evaluation method. We cannot afford human evaluation on hundreds of thousands of summaries. A fundamental characteristic of our algorithm should be that we can have confidence in its results as somehow representative of the quality of the summaries. Intuitions are unlikely to suffice. We will explain in the next section what evidence we seek. We have conducted bibliographic research to find a method suited to our needs.

[1] Our training data, obtained courtesy of the Document Understanding Conference, contains over 1100 texts; the number of summaries generated exceeds 300,000.

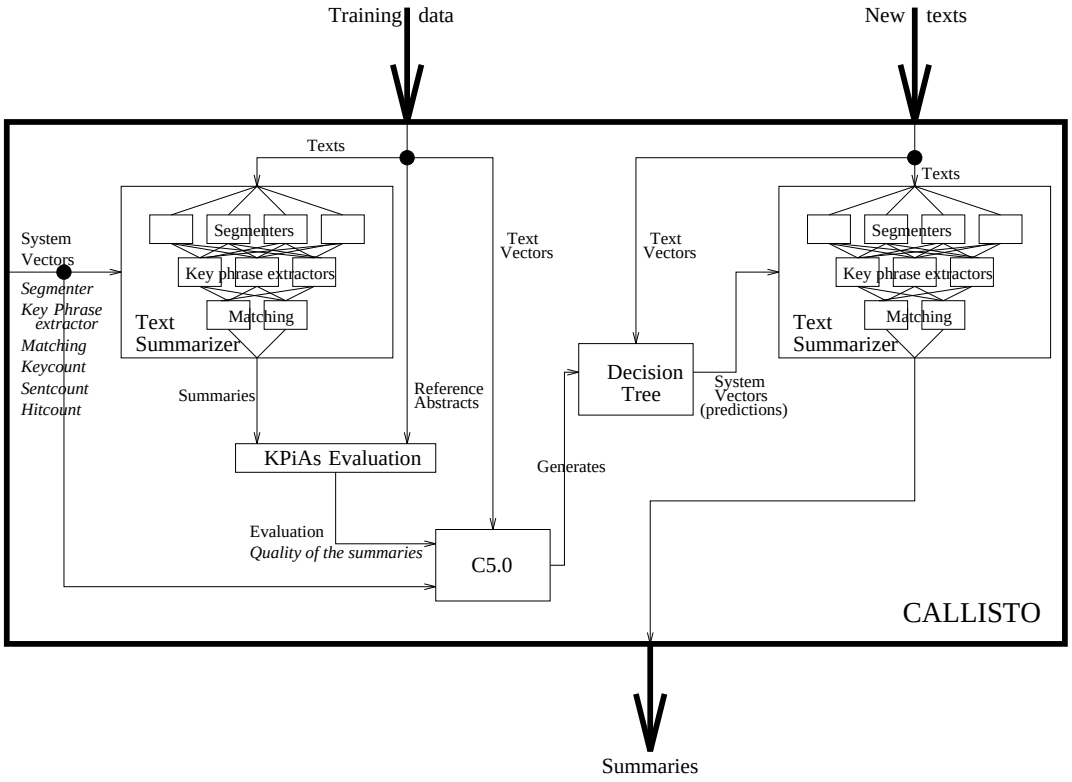


**Fig. 2.** Overview of the CALLISTO System.

## 3 Evaluation in Automatic Summarization

Many areas of Natural Language Processing consider evaluation difficult. In summarization, translation or generation, measuring how well a system performs is subjective and therefore poses challenges of its own. Researchers in automatic summarization have applied methods with various methodologies, cost and degree of reliability. Some can be dismissed as not useful for us: either not suited to our situation or too resource-intensive. Others deserve closer attention. In the end, however, we are left with very few methods among which to choose.

Faced with the difficulty of evaluating summary quality in general, one may follow Mani [2001] and look at determining the suitability of a summary for a particular task. Mani distinguishes *extrinsic* and *intrinsic* methods.

– Extrinsic methods test a summarizer in relation to some task, for example allowing one to answer questions about the source text more accurately. Human judges are needed to work on the output of the system, so these methods are too time-consuming for us (we cannot afford evaluating many thousands of summaries).

– Intrinsic methods are intended to give an absolute measure of either the *quality*, how well a summary is written, or the *informativeness*, how much knowledge it contains. Our policy of building a summary by concatenating sentences extracted from the full text allows us to be a little less concerned about the first issue; we take advantage of the document's author's writing skills. (This does not mean that we avoid problems of dangling references and irrelevant logical links, but these can be addressed later – see [Mani *et al.*, 1999] for more about readability.) Therefore, we focus on informativeness.

Model summaries are available for our training data, so comparison with them seems the most appropriate. Although human judges have successfully compared generated summaries against a model summary [DUC, 2001 and 2002], there are also automatic methods [Donaway *et al.*, 2000] that compute measures at the sentence or word level. We have a sentence-extracting summarizer but the model summaries are produced by abstracting. They need not differ much from extracts, but mechanical comparison is problematic, since wording is often changed for clarity. Time-consuming preprocessing would be needed to work at the sentence level. The alternative is a content-based measure to compute how close the summary is to the model.

Mani [ibid.] suggests that such a method would be adequate in a case like ours. We are working with news reports, and model summaries tend to be cut-and-paste material from the source. A good extract may have a lot in common with the model abstract as long as it contains the most relevant sentences. For example, synonymy need not be the problem that it would be if we were working with more completely rewritten reference abstracts. In the final analysis, almost as with sentence recall, a summary will score well if and only if the best sentences are extracted from the source.

A measure can be deemed trustworthy if it is shown to correlate well with human judgment. Lin and Hovy tested several content-based measures using stemming and N-gram matching [Lin and Hovy, 2002]. They found that many of their measures agreed well with human evaluation. We will present in detail the one we chose, but first we review the KPiA measure and explain why we decided not to use it in the evaluation framework.

## 4 Details of the Key-Phrase-in-Abstract Method

The Key-Phrase-in-Abstract (KPiA) method requires a model summary – often the author's abstract – for each text in the training data. We remove stop words and count occurrences in the model summary of the "phrases" between them. Our formula favours the first occurrence of a KPiA:

$$\begin{aligned} score = \quad & \sum 1.0 * KPIA_{unique} \\ & + \sum 0.5 * KPIA_{duplicates} \end{aligned}$$

We then compute the *nearly best* score from the text: we extract the $n$ sentences with the highest density of KPiAs, where $n$ is the desired summary length. Placed in document order, those sentences are taken as a model extract. We rate this model using our formula. A flaw of this procedure is that the reference may not achieve the highest score possible. Due to the bonus term in the formula, the sentences containing the largest number of KPiAs may not be the most informative[2]. This is why the model is labelled "nearly best".

We rate extracted summaries by dividing their score by that of the model. Because the model is only "nearly best", the quotient can exceed 1. When this happens, we take the rating to be 1. This evaluation method has the obvious advantages of being automatic, easily computed and reasonably fast.

We use the KPiA method to train CALLISTO in its current incarnation. When the system was designed, KPiA appeared comparable to other content-based measures of similarity with the abstract. We have decided to replace it because we do not know how it correlates with human judgment; section 5 presents a method that has been shown to correlate well on DUC dataset. It is also easily computed without human intervention.

## 5 The Proposed Evaluation Framework

Our evaluation framework is based on a measure proposed by Lin and Hovy [Lin and Hovy, 2002]. We present their method, explain how we modified it and discuss our framework.

### 5.1 Lin and Hovy's Measure

Lin and Hovy's work was inspired by IBM's method of evaluating automatically output of Machine Translation [Papieni *et al.*, 2001]. It compares a text with model translations by matching N-grams (length 1-4). Lin and Hovy adapt this measure to text summarization, and show [Lin and Hovy, 2002] that it agrees well with human judgment[3]. They used only recall (how many N-grams from the model abstract are in the summary?) instead of BLEU's modified weighted precision that considers multiple reference translations[4].

Lin and Hovy considered several N-gram matching methods: with or without stemming, with several weights for various N-grams (chiefly unigrams and bigrams). Then they proposed a ranking for the systems that participated in DUC 2001, and compared it with the official ranking established by human judges. *All*

[2] Consider an extreme example: the same KPiA ten times in three different sentences, and no other sentences with more than ten KPiAs. This three-sentence extract has a KPiA score 15.5. Suppose there are two more sentences with only eight KPiAs but all different. This two-sentence extract would score 16 with one less sentence.

[3] A continuation of that work [Lin and Hovy, 2003] has led to the "ROUGE" metric [Lin, 2003], adopted as the only evaluation criterion in the DUC 2004 exercise.

[4] One problem is that the length of the summary is not considered in the evaluation. We will return to this in more detail.

the methods considered correlate well with human evaluation, at least as well as two people agree. This significant finding suggests that mechanical assessment cannot achieve perfect agreement with human judgment because human evaluation is imperfect: there cannot be one and only accurate human rating.

Our framework implements a method based on [Lin and Hovy, 2002] that agrees more closely with the official DUC ranking. Weights of 1/3 are applied to unigram and 2/3 to bigram scores produced by Porter's stemmer [Porter, 1980] after removing stop words from the text. This produces reference lists of stemmed unigrams and bigrams from the model summary. We similarly stem the summary under evaluation and count the number of its unigrams and bigrams appearing in the reference lists. This number is the overlap. The recall measure is:

$$Recall = \frac{1}{3}\frac{Overlap_{Unigrams}}{N} + \frac{2}{3}\frac{Overlap_{Bigrams}}{N-1}$$

### 5.2 F-Score

A tested summary should agree with the model not only in content but also in length, an aspect measured by *Precision*. Combining informativeness and brevity is possible through F-Score, a measure used in Information Retrieval:

$$\text{F-Score} = \frac{2 * \text{Recall} * \text{Precision}}{\text{Recall} + \text{Precision}}$$

Popescu-Belis [Popescu-Belis, 1999] explains that the F-Score formula uses the harmonic average[5] rather than, for instance, the arithmetic average because we want F-Score to be low whenever either Recall or Precision is really low (if one is 0 and the other 1, we prefer the score of 0 rather than 0.5). The summaries have to be both informative *and* concise.

In [Papieni *et al.*, 2001], using F-Score as such was not possible because of multiple references. We do not have this problem, so we can consider F-Score rather than Recall. We only need to prove the correlation with human judgment. As discussed in [Lin and Hovy, 2002], the important point is to preserve ranking. We have a dataset of 334210 summaries on which it is very easy to see if the measures correlate. We computed the correlation of Recall and F-Score with the Spearman Rank-order correlation[6] as Lin and Hovy do in their article:

[5] Given non-zero $x$ and $y$, we compute a weighted average $\frac{1}{\alpha\frac{1}{x}+(1-\alpha)\frac{1}{y}}$ with $\alpha \in [0,1]$. If $\alpha = \frac{1}{2}$, we get the harmonic average:

$$\frac{1}{\frac{1}{2}(\frac{1}{x}+\frac{1}{y})} = \frac{2 * x * y}{x+y}$$

If $x$ or $y$ is 0, the harmonic average is also 0.

[6] If we want to compare two rankings of $n$ objects, and we denote $a_i$, $b_i$ the rankings of the same object $i$, the Spearman Rank-order correlation is:

$$\rho = 1 - 6\frac{\sum_{i=1}^{n}(a_i - b_i)^2}{n(n^2-1)}$$

$$\rho = 86.94\%$$

In the end, we find that recall correlates well with human judgment at more than 98% [Lin and Hovy, 2002] and the correlation between Recall and F-Score is satisfactory. We can assume that F-Score correlates well with human judgment. The transitivity rule we use here is only acceptable because the correlation scores are very high. It is, strictly speaking, only true when the Spearman rank-order correlation is 100%[7]. However, as both scores are high, we can assume that the correlation between F-Score and human judgment is good.

### 5.3 Methodology

This study has been motivated by the need to evaluate the results of CALLISTO and assess the effects of its modifications. At several levels, biases were initially chosen without knowing that they were indeed apt; examples appear in the next section. The evaluation framework described here should enable us either to validate these choices, or to identify better alternatives.

We have applied the evaluation framework to the DUC 2002 training and test sets merged, given that reference summaries are available for the test set as well. We then generated a large set of summaries: we ran CALLISTO with plausible parameter settings on this collection of over 1100 texts. The system's machine Learning component must be trained before it is applied to new texts. We used standard 10-fold cross validation. CALLISTO was trained on 9/10 of the DUC 2002 data set and then applied to the remaining 1/10 of the data[8] and its results on this 1/10 evaluated using Lin and Hovy's method. Because this measure has been shown to correlate highly with the ratings of human judges, we can assume that our results are indicative of the actual quality of the summaries.

### 5.4 Baseline

[Brandow *et al.*, 1995] has established that a very efficient baseline for automatic summarization of news reports is to extract the first few sentences of the text (approximately the first one or two paragraphs). This can be explained by the journalistic style: when one reads a news report, one expects to find the answer to the W-Questions (who, when, where, what, possibly why) in the first lines. This can be deemed an acceptable summary in many cases. Therefore, it is important to include such a challenging baseline in every evaluation study. Note that this baseline is highly genre-specific. If we were to work on another kind of texts, it would be quite probably much less interesting.

---

It is 1 (or 100%) for a perfect correlation and -1 for a "perfect" disagreement.

[7] In that case, if two measures have exactly the same ranking as a third one, the correlation between the two is 100% too.

[8] The data is divided into 10 equal parts, and learning repeated 10 times: 9 parts are used for training, one for evaluation. In effect, the results of learning – averaged – are more reliable.

We have extracted the first sentences of each text in the corpus and run Lin and Hovy's evaluation method on them. The length of the summary is one of the parameter and can take the following values in our dataset: 3, 4, 5, 8 and 12. We have computed the baseline for each of those and averaged the results to obtain the score to beat. As expected, and as we will see in the next section, this baseline is extremely challenging.

### 5.5 Statistical Significance

To establish the statistical significance thresholds of our results, we looked at Information Retrieval methods [Hull, 1993], since retrieving relevant sentences from a document can be seen as equivalent, at least from a statistical point of view, to retrieving relevant documents from a dataset.

Two-way ANOVA appears to apply well to our case since it enables us to get significance results when more than two methods are to be compared[9]. The calculations needed to apply this method are explained in [Hull, 1993]. We will only mention the value we found for the statistical significance when reporting the results in section 6.

## 6 An Experiment Using the Framework

### 6.1 Choices to Evaluate

In the experiment we report here, we have made four changes in the system, regarding the measure and the learning module.

- *The evaluation measure.* We chose to replace the Key-Phrase-in-Abstract evaluation method in the framework. It makes sense, for the same reasons, to train the system with F-Score rather than KPiAs and see if we get any improvement. Since we evaluate the overall results of our system with F-Score, we should get better results if we drop KPiA from the training process, assuming that the process to get large F-Score scores is learnable at all. The solution, however, is not that obvious because it is possible, for instance, that the dependencies between the attributes and the F-Score measure are too difficult to learn and that, paradoxically, we get better results using KPiA.
- *The learning algorithm.* The learning component is a crucial part of CALLISTO since it is responsible for predicting the right configuration to apply to each text. Therefore, the main idea underlying the system – that the setting to choose depends on the input text – is valuable only if we can efficiently take advantage of knowledge in the training data to predict choices for further texts.
  C5.0 [Quinlan, 1997] was chosen as a fast, efficient state-of-the-art program. It does happen that other learners work better for a certain application, and

[9] We have many configurations to evaluate, so pair-tests – tests that evaluate statistical significance for only two methods at the same time – would hardly be doable.

it could be the case for us. Out of the wide range of Machine Learning algorithms we considered, very few were applicable to our dataset because of the large number of examples. We present here the only one that brought significant improvement in comparison to C5.0: Naive Bayes. A good description of the Naive Bayesian classifier can be found in [John and Langley, 1995].

- *The selection process.* The values representing the quality of the summaries are discretized (with K-means). So, we do not get one only best configuration after having run the model from the learning algorithm, whatever it is, on the test data; we get a *set* of configurations predicted as yielding good summaries. How to choose between those is an open problem. In the original version of CALLISTO, the first found was picked, a process which is equivalent to random selection. Now, an interesting property of the Naive Bayes classifier is that it does not just predict a class, it also outputs a confidence in its prediction (the probability). This is particularly important for us with regard to the selection process inside the best class: we can select the one predicted as yielding a good summary with the highest confidence[10].
- *The number of classes.* As we already said, we do not know the optimal $K$ for the K-means discretization algorithm. Therefore, we tried every value between 2 and 20 and this number is taken as the horizontal coordinate when reporting the results on a graph.

### 6.2 Experiments

The results are reported in Fig. 3. The horizontal axis is the number of classes while the vertical axis represents the normalized F-Score scores[11] that is the average quality of the summaries generated. Therefore, every dot is one different version of CALLISTO. The statistical significance computed for these experiments is 2.59%.

### 6.3 Discussion

The baseline turns out to be better than every version of the system we test here. This is a problem we had in almost all our experiments (only very few configurations of the system beat that baseline and never significantly), we believe that it is due to the way journalists write their reports, as explained earlier, and that this baseline would not obtain such high scores for other genres.

Beside that, Naive Bayes significantly outperforms C5.0 on all configurations but two special cases. For the small number of classes with F-Score, we think that

[10] C5.0 outputs a confidence percentage with its predictions. We tried to use it inside the framework but it did not improve the overall quality of the summaries. Therefore we kept the random selection process, initially used in CALLISTO, when applying the C5.0 learning algorithm.

[11] As the set of parameters we use is finite, the number of summaries is as well and there exists a maximal possible score for each text that we computed and used to normalize the results.

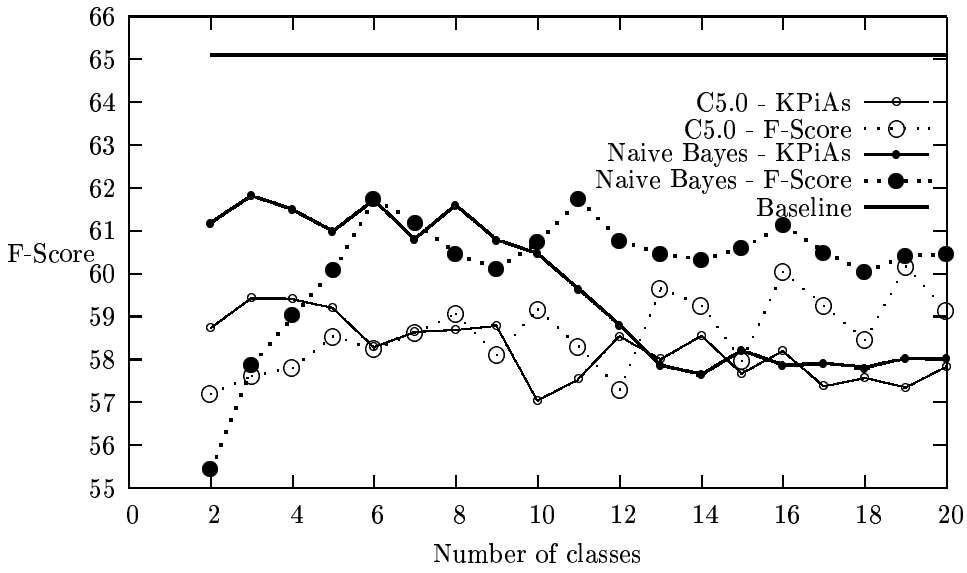


**Fig. 3.** F-Score Scores of CALLISTO (with different learners and measures).

Naive Bayes does not perform well because the discrimination between bad and good summaries is not sufficient and for the large number of classes with KPiA, it is probably a phenomenon of overfitting. As for the measures, it seems that, in effect, F-Score is difficult to learn for both learners. Slightly better results are achieved on average when we use F-Score with Naive Bayes but these are below the statistical significance threshold and are not sufficient to conclude reliably.

The reason why Naive Bayes gives in the end better results than C5.0 is not clear. The fact that we have a reliable selection method, being wholly part of the learning process may be one reason, (We have tried several selection methods with C5.0, including the use of the learner confidence, and none was significantly better than random). This assumption is confirmed by another experiment [Rigouste, 2003] that proves that the performance of Naive Bayes in terms of summary quality drops significantly when we do not select the best configuration based on the probability.

The fact that Naive Bayes helps produce better summaries than C5.0 is all the more surprising when we look at the accuracy of the classifiers on our dataset in 10-fold cross validation: C5.0 seems better than Naive Bayes. However, we are only interested in a small subset of the predictions (the ones regarding the best class, in which we pick the configurations to apply) and, with this selection method, we are only interested in the predictions Naive Bayes makes with high confidence. It is possible that on those, Naive Bayes outperforms C5.0. This hypothesis is in part confirmed by other experiments [Rigouste, ibid.] that show that, whereas the selection process for Naive Bayes helps improve the error rate and select the most reliable predictions, the effect is totally opposite with C5.0,

whichever selection method is chosen. The accuracy decreases, which means that C5.0 is worse on the best class, the one we are interested in, than on the others.

This analysis demonstrates the usefulness of our framework. Based just on the global error rates, we would have preferred C5.0, without necessarily having the idea to investigate further and examine the error rate on selection. And we would have missed the fact that the configurations with Naive Bayes are better than the ones with C5.0. With the framework, however, we can judge configurations of the system on the overall qualities of the summary, that is, on the final output of the system, and not on an intermediate measure in CALLISTO, such as the error rate of the learning component. Thus the framework enabled us to make decisions based on what we are really interested in (producing better summaries) and not on intermediate measures (the error rate of the learning algorithm) for which we do not know how they affect the final performance of the system.

To conclude with Naive Bayes, we have to say that the accuracy is a bit better on the configurations selected than on the whole but the absolute improvement is not great. In other experiments [Rigouste, ibid.], where we modified the discretization process (which is k-means in all the experiments above), we found a configuration which is better than all the others in terms of F-Score averages but, even then, the error rate of the learning algorithm is still surprisingly high: 55%. Therefore, we believe that there is still a lot of room for improvement using the framework.

## 7 Conclusions and Future Work

We have presented the CALLISTO summarization system and tried to modify it in several ways to improve it. We have judged the quality of the overall summaries produced and therefore the relevance of the biases chosen, using an evaluation framework. This framework, based on a measure proposed by Lin and Hovy, is fully automatic and can be applied to large corpora, which gives more significance to the results. It comes with a very demanding baseline obtained with the leading text extraction technique and a measure of the statistical significance of the results with 2-way ANOVA. We found that Naive Bayes produces good results and can learn how to get high F-Score scores more efficiently than C5.0. We think that this result is due in part to the fact that the selection process based on probabilities is better than those we investigated with C5.0.

For the framework to be deemed totally trustworthy, a reliable experiment with human judges would be necessary. Other than that, the most promising direction of future work is to keep using the framework to find useful text features and validate CALLISTO's methodology. Indeed, in experiments not presented in this paper, we found that the choices of the measure, discretization and selection processes were very learner-dependent and we have not found other learners better than Naive Bayes (most were intractable anyhow, given the size of our dataset). However, while experimenting with various attributes to characterize the text, we concluded that we had not found the right features to characterize a document yet.

Besides, the best configuration we found achieves only around 66% of the best possible performance and there is still much room for improvement, even without adding other tools or changing parameters, provided that we find the right attributes to characterize a document. The experiments we conducted to find the best possible scores reveal that every configuration could indeed be useful and brings a better F-Score than the others a non-negligible number of times, as reported in table 1. These limits can even be pushed further by adding other new and more powerful components to the system. Thanks to its methodology taking advantage of other tools, CALLISTO may fully benefit from any improvement in the fields of segmentation or key phrase extraction.

| Segmenter | Key Phrase extractor | Number of texts |
|---|---|---|
| C99 | NRC Extractor | 220 |
| No segmentation | NRC Extractor | 146 |
| Columbia Segmenter | NRC Extractor | 118 |
| No segmentation | Kea | 98 |
| C99 | Kea | 89 |
| C99 | NP Seeker | 83 |
| TextTiling | NRC Extractor | 79 |
| No segmentation | NP Seeker | 78 |
| Columbia Segmenter | Kea | 68 |
| TextTiling | NP Seeker | 49 |
| Columbia Segmenter | NP Seeker | 47 |
| TextTiling | Kea | 46 |

**Table 1.** Number of texts on which a given combination segmenter/extractor produces a higher score than all the others.

## Note

A shorter version of this technical report will be published in the proceedings of the Seventeenth Canadian Conference on Artificial Intelligence (AI'2004).

## Acknowledgements

The National Institute of Standards and Technology, the organizer of the Document Understanding Conferences, has been instrumental in all evaluation exercises related to text summarization. Partial support for the first author came from the Natural Sciences and Engineering Research Council of Canada.

## References


[Brandow *et al.*, 1995] Brandow, R., Mitze, K. and Rau., L. Automatic condensation of electronic publications by sentence selection. *Information Processing and Management*, 31(5):675–685.

[Copeck *et al.*, 2002] Copeck, T., Japkowicz, N., and Szpakowicz, S. Text Summarization as Controlled Search. *Proc 15th Conf of the Canadian Society for Computational Studies of Intelligence (AI'2002)*. 268-280.

[Donaway *et al.*, 2000] Donaway, R.L., Drummey, K.W., and, Mather, L.A. A Comparison of Rankings Produced by Summarization Evaluation Measures. *Proc Workshop on Automatic Summarization*, 69-78. New Brunswick, NJ: Association for Computational Linguistics.

[DUC, 2001 and 2002] Document Understanding Conference, National Institute of Standards and Technology. http://duc.nist.gov/

[Hull, 1993] Hull, D. Using statistical testing in the evaluation of retrieval experiments. *Proc SIGIR '93*, 329-338. Association for Computing Machinery.

[John and Langley, 1995] John, G., and Langley, P. Estimating continuous distributions in Bayesian classifiers. *Proc 11th Conf on Uncertainty in Artificial Intelligence*, 338-345.

[Lin, 2003] Lin, C.-Y. ROUGE, Recall-Oriented Understudy for Gisting Evaluation. http://www.isi.edu/~cyl/ROUGE/.

[Lin and Hovy, 2002] Lin, C.-Y., and Hovy, E.H. Manual and Automatic Evaluations of Summaries. *Proc Workshop on Automatic Summarization*, ACL-02, Philadelphia, PA, July 2002.

[Lin and Hovy, 2003] Lin, C.-Y., and Hovy, E.H. Automatic Evaluation of Summaries Using N-gram Co-occurrence Statistics. *Proc 2003 Language Technology Conference (HLT-NAACL 2003)*, Edmonton, Canada, 150-157.

[MacQueen, 1967] MacQueen, J. Some methods for classification and analysis of multivariate observations. *Proc Fifth Berkeley Symposium on Mathematical Statistics and Probability*, (1), 281-297.

[Mani, 2001] Mani, I. *Automatic Summarization*. John Benjamins Pub Co.

[Mani and Maybury, 1999] Mani, I. and Maybury, M.T. (eds.) *Advances in Automatic Text Summarization*. Cambridge, Massachusetts: MIT Press.

[Mani *et al.*, 1999] Mani, I., Gates, B. and Bloedorn, E. Improving Summaries by Revising Them. *Proc 37th Annual Meeting of the Association for Computational Linguistics*, University of Maryland, College Park, MD, 558–565.

[Papieni *et al.*, 2001] Papieni, K., Rouckos, S., Ward, T., and Zhu. W.-J. BLEU: a Method for Automatic Evaluation of Machine Translation. IBM Research Report RC22176(W0109-022).

[Popescu-Belis, 1999] Popescu-Belis, A. Evaluation of Natural Language Processing Systems: A Model for Coherence Verification of Quality Measures. A Blueprint for a General Infrastructure for Natural Language Processing Systems Evaluation Using Semi-Automatic Quantitative Black Box Approach in a Multilingual Environment. European project LE4-8340ELSE: Evaluation in Language and Speech Engineering.

[Porter, 1980] Porter, M. An algorithm for suffix stripping. Program, 14(3):130– 137.

[Quinlan, 1997] Quinlan, J.R. Data Mining Tools See5 and C5.0. http://www.rulequest.com/see5-info.html.

[Rigouste, 2003] Rigouste, L., under the supervision of Japkowicz, N. and Szpakowicz, S. Evolution of a Text Summarizer in an Automatic Evaluation Framework. Master's thesis. http://www.site.uottawa.ca/~rigouste/thesis.ps.