*i*Sentenizer: An Incremental Sentence Boundary Classifier

Fai WONG
CIS, University of Macau
Macau S.A.R., China
derekfw@umac.mo

Sam CHAO
CIS, University of Macau
Macau S.A.R., China
lidiasc@umac.mo

Abstract:

In this paper, we revisited the topic of sentence boundary detection, and proposed an incremental approach to tackle the problem. The boundary classifier is revised on the fly to adapt to the text of high variety of sources and genres. We applied *i*$^+$Learning, an incremental algorithm, for constructing the sentence boundary detection model using different features based on local context. Although the model can be easily trained on any genre of text and on any alphabet language, we emphasize the ability that the classifier is adaptable to text with domain and topic shifts without retraining the whole model from scratch. Empirical results indicate that the performance of proposed system is comparable to that of similar systems.

Keywords:

***i*$^+$Learning; incremental learning; tagging; sentence boundary detection**

1. Introduction

There is a consensus that the fundamental unit of text in theoretical and computational linguistics is sentence, and many natural language processing systems are generally performed on sentences, e.g. part-of-speech (POS) tagging, syntactic parsing, information retrieval, machine translation, etc. Although the task to segment an input text into sentences seems to be an uninteresting step performed before any linguistic analysis is undertaken, in reality, it is a non-trivial task. The detection of sentence boundaries has been recognized as one of the vital preprocessing steps in the development of practical language processing applications. This involves resolving the use of ambiguous punctuations and determining if the current punctuation is a true sentence delimiter [1]. Punctuations such as the exclamation mark, question mark, semicolon and period usually indicate the end of sentence. Among them, exclamation mark and question mark are generally unambiguous sentence delimiters. The semicolon can either be a separator of elements or a delimiter of sentences. But the most common ambiguous punctuation is period. It can either be used as a sentence boundary marker or part of abbreviation and initial. This makes the detection task hard to decide when it is a full-stop or an abbreviation marker, especially when the period is used at the same time for both cases. For phrases using periods in e-mail address, web address, decimal numbers and ellipsis between two words (or successive periods), they are usually not followed by a blank space, and as a consequence, the system can easily determine that they are not sentence separators.. However, in order to implement a reliable sentence boundary detection model, sophisticated logics and algorithms are required to tackle this ambiguity problem.

978-1-4244-6899-7/10/$26.00 ©2010 IEEE

2. Related work

In literature, several sentence boundary detection systems have been reported, and basically these systems can be categorized into two types according to the different approaches the systems use to identify the ending of sentence. The works of Grefenstette and Tapanainen [1] and Silla Jr., et al., [2] are the representatives of rule-based approach. The systems encode rules as regular expressions, and use a set of regular expressions to represent possible patterns of *non-boundary* period that marks abbreviations, numbers and other sequences like e-mail and web addresses. In classification, period in text is checked and surrounding context is analyzed against the regular expressions. If any of them is matched, then the period is determined as an abbreviation marker, otherwise, it is considered as a sentence separator. However, rules are never exhaustive, hard to prevent from conflicts and not robust to the text with domains and genres shifts. The second type of boundary detection systems is based on machine learning approach. It treats the detection task as a classification problem, uses features of local context of potential boundary markers such as spelling of word, capitalization, length of abbreviation, part-of-speech, etc. Compared to manually constructed systems, machine learning models are easier to develop since only annotated training data is required [3]. The representative systems based on this approach are Satz [4] and MxTerminator [5]. Satz uses either a C4.5 decision tree or a neural network to disambiguate the role of punctuation mark in sentence, using the prior distributions of word class surrounding the possible

end-of-sentence punctuation mark as features. On the other hand, MxTerminator applies maximum entropy model to learn the contextual features of ambiguous punctuations by considering the token preceding and following a sentence boundary, together with the heuristic information regarding the abbreviations from the annotated training corpus. Recently, Kiss and Strunk [6] propose an unsupervised sentence boundary detection system Punkt that uses collocation information as evidences derived from unannotated corpora for detecting abbreviations, initials, and ordinal numbers. The collocation information and other model parameters are empirically derived from large development corpus of *Wall Street Journal*. Although systems based on machine learning approach are easier to construct, if there are any revisions, due to new domains (new languages) or text genres that have different characteristics from the developed corpus, they must be reconstructed and retrained. Furthermore, newly constructed model has to be balanced with the data from new domain and the existing training data from old domain, such that the effect of the one domain does not dominate the other.

3. Motivation

In practical application, a reliable sentence boundary detection model is very important for preparing input text of an *online* machine translation system. On the other hand, the detection system must be capable of learning from new data as well as features without affecting the normal operation of the translation service. According to our experience, the translation system that we are working has the following characteristics:

- The texts are of a high variety of sources, domains, and formats. The input text can be plain texts, web pages, word documents or PDF files. Thus, the shifts of topics and text genres are very common.
- The system supports the translation for Chinese, English and Portuguese. Since the segmentation of Chinese sentence is different from the task we discussed in this paper, English and Portuguese will be the focus languages. The detection of sentence boundaries acts as a preprocessing step and it is constructed independently from the translation system.
- The translation system runs as a non-stop Web service over the internet, and normally, it is not allowed to be interrupted in any circumstances.

Therefore, in order to cope with the described issues, we employ an incremental machine learning approach as another framework to deal with the problem of sentence boundary detection that is different from the existing systems reported in the literature. Moreover, since these systems are conventionally constructed and modified offline, minor changes require them to be rebuilt. In this paper, the boundary detection system *i*Sentenizer is implemented based on the i^+Learning principle [7], which is an incremental decision tree learning algorithm.

4. i^+Learning algorithm

In this section, we generally review the underlying theory of i^+Learning. It has been proposed as a new attempt that contributes the incremental learning community by means of intelligent, interactive, and dynamic learning architecture. In the meanwhile, it complements the traditional incremental learning algorithms in terms of performing knowledge revision in multiple dimensions. The algorithm grows an on-going decision tree with respect to either the new incoming instances or attributes in two phases: 1) Primary Off-line Construction of Decision Tree (POFC-DT): as a fundamental decision tree construction phase in batch mode that is based on the initial training data, where a C4.5-like decision tree model is produced; 2) Incremental On-line Revision of Decision Tree (IONR-DT): as incoming of the new instances or feature attributes, this phase is responsible for merging the new data into the existing tree model to learn incrementally the new knowledge by tree revision instead of retraining from scratch.

4.1. POFC-DT phase

The first step is an ordinary top-down decision tree construction phase that starts from the root node, using a splitting criterion to divide classes as “pure” as possible until a stopping criterion is met. The objective of this phase is to construct an optimal base tree, in order to have a robust foundation for further tree expansion. Binary tree structure is adopted in constructing such base tree. Binary tree has the same representational power as the non-binary tree, but it is simpler in structure and has no loss of the generated knowledge. This is because binary decision tree employs a strategy that a complex problem is divided into simpler sub-problems, in which it divides an attribute space into two sub-spaces repeatedly, with the terminal nodes associated with the classes [8].

To build a primitive binary tree, it starts from a root node d derived from whichever attribute a_i in an attribute space A that minimizes the impurity measure. A binary partition can be denoted by a four-tuple representation (d, T, d^L, d^R), where d is a decision node and T is a splitting criterion on d, and d^L and d^R are the node labels for partitions of the left and right data sets respectively. Since a binary tree is a collection of nested binary partitions, it can be represented in the following

recursive form,

$$D = \left\{ \left(d, SPL, d^L, d^R \right), D^L, D^R \right\} \quad (1)$$

where D^L and D^R denote the left and right sub-trees respectively, which are induced by the partition node d [9]. Kolmogorov-Smirnoff (KS) distance [10][11] is employed as the measure of impurity at node d, which is denoted by $I_{KS}(d)$ and is shown in equation (2) below:

$$I_{ks}(d) = \max_{v \in value(d)} \left(| F_L(v) - F_R(v) | \right) \quad (2)$$

where v denotes either the various values of a nominal attribute a with test criterion $a = v$, or a cut-point of a continuous-valued attribute a with test criterion $a < v$; $F_L(v)$ and $F_R(v)$ are two class-conditional cumulative distribution functions that count the number of instances in the left and right sub-trees respectively, which is partitioned by a value v of an attribute a at a decision node d. KS is a well-known measure for the separability of two distribution functions, it is especially simple and computationally fast both in the training and classification stages. Hence, a best single test is picked across all attributes by enumerating the possible tests and selecting the one with the greatest KS distance. A decision tree grows by means of successive partitions until a terminal criterion is met.

4.2. IONR-DT phase

IONR-DT phase acts as a central character in the incremental decision tree algorithm. It embraces the faith that whenever a new instance and/or a new attribute is coming, this phase dynamically revises the fundamental tree constructed in POFC-DT phase without sacrificing the final classification accuracy, and eventually produces a decision tree as same as possible to those algorithms with all training examples available at the beginning. IONR-DT phase adopts the tree transposition mechanism of ITI [12] as a basis to grow and revise the base tree. Besides, it preserves the essential statistical information to manage the decision tree. Such style of decision tree differs from the batch mode trees, since it remembers the information of instances regarding the respective possible values as well as the class label, in order to process the transposition without rescanning the entire data set repeatedly.

KS measure is again applied in this phase for evaluating the goodness of a decision node. Once a (set of) new instance(s) is ready to be incorporated with an existing tree, IONR-DT phase carries out the following several steps: 1) updates the statistical information on each node that the new instance traversed; 2) merges the new instance into an existing leaf or grows the tree one level under a leaf; 3) evaluates the qualification for the test on each node downwards starting from the root node; 4) calls the pull-up tree transposition process recursively to revise the existing decision tree for any attribute test that is no longer best to be on a node; and 5) revises the new decision tree and performs the next classification.

4.3. i^+Learning regarding attributes (i^+LRA)

If an instance is available together with an unseen (new) attribute, except the above general steps, an additional procedure for incorporating a new attribute appropriately with an existing decision tree has to be called subsequently. In this algorithm, each new attribute has been treated at least medium important in default rather than noise in other algorithms, although its goodness measurement might be lower on its first occurrence. This is because for a new domain, when a new attribute is considered to incorporate into the learning process, it should be logically considered as one of the decision nodes, even though the evidence is rare and the attribute might be irrelevant from the statistical point of view.

Further significant, in order to avoid the situation that an attribute has been appended mistakenly, i^+LRA offers an alternative way to manually assign one of the four pre-defined importance grades to an attribute. This characteristic enables i^+LRA algorithm flexible enough to deal with the incremental data appropriately. Table 1 lists the importance grading schema as well as the respective action being taken during the tree revision process.

Table 1. Four classes of the importance grades.

Grade	Action in Tree Revision
High	Should be incorporated into the top several percentages of important decision nodes.
Medium	May not be a must, can be incorporated into the decision nodes above the average importance.
Low	Perhaps an irrelevant attribute that probably be appended closely to the leaf node and later on be pruned; or remain for supporting other attributes.
None	Treated as noise and ignored.

4.4. Importance measure (I_{KS})

After selecting the importance grade from Table 1 for a new attribute, the crucial step is to determine a preliminary coefficient ω for its impurity measure (I_{KS}). This coefficient is vital for a new attribute, and is used as a reference index for its importance measure. It is computed only once for each importance grade, in order to enable the new attribute to compete with its comparable attributes in being a decision node. This

coefficient is decided as the ratio between the average KS measure of the attributes in the same rank that the new attribute belongs to, and the KS measure of the new attribute itself. Equation (3) illustrates the situation:

$$\omega = \frac{mean\left(\sum_{i=1}^{l} I_{ks}(a_i)\right)}{I_{ks}(a_{new})} \qquad (3)$$

where a_{new} is a new incoming attribute; a_i is an attribute of the same rank with a_{new}; and l is the total number of such attributes, which is less than the total number of attributes in the attribute space A, i.e. $l \leq |A|$. Once such preliminary coefficient ω for a_{new} has been worked out, it could be applied to a_{new} by multiplying it to the KS measure of a_{new}, to enlarge the importance of a_{new} according to the given importance grade automatically. This strategy examines the initial importance of a_{new} regardless of its KS measure, which can prevent an actually important new attribute being treated as useless due to its newness and occupies little training data. Thereupon, a normal tree transposition process is carried on as usual to properly fit the new attribute a_{new} to a right position.

5. *i*Sentenizer

Our boundary detection system *i*Setenizer is built based on the described i^+Learning algorithm. The *i*Setenizer is trained by using the features extracted from the trigram contexts of the training corpus. That is, we consider only the words immediately preceding and following the potential boundary punctuation marks, including the exclamation mark, semi-colon, question mark and the period. For each possible sentence separator, we construct a feature vector v as input to the *i*Sentenizer and use its prediction to determine whether the punctuation mark denotes a sentence boundary or not. In order to construct v, we consider the following eleven linguistic features f_n surrounding the boundary marker, where $n = 0, ..., 10$:

- Capitalized initial of *preceding* word
- Capitalized initial of *following* word
- Capitalized of *preceding* word
- Capitalized of *following* word
- *Following* is the dollar sign, '$'
- *Following* is number
- Length of *preceding* word is less than 5 characters
- *Current* is potential punctuation, and *following* is dash or left quotation mark
- *Current* is potential punctuation, and *following* is not left quotation mark
- Length of *preceding* word
- Length of *following* word

The features are selected through experiments. The combination of features that gives the best performance is determined from the English articles of *Brown corpus*. In order to maximize the portability of the system, the features we define here do not reply on any sort of orthographic information and elaborate regular expression for specific content. This makes the system more robust and flexible enough in the application to a variety of languages without significant adjustments. For each of the training instance, the associated features are encoded as a binary vector v (beside the last two components representing the length of preceding and following words), each component of v corresponds to a possible feature value b_i of a feature f_i in the feature set. The value of the component corresponds to a question which has value *one* if the corresponding feature f_i has value b_i, or value *zero* if the corresponding feature f_i has another feature value.

6. Data sets

In order to test the performance of our system, we have evaluated *i*Sentenizer on the following English and Portuguese corpora: 1) *Brown corpus* from the Penn Treebank [13]; and 2) *Tycho Brahe corpus*, which is a parsed corpus of historical Portuguese [14]. In the first corpus, words are annotated with POS information and the text is split into sentences. However, we do not make use of this additional annotation information to support sentence boundary detection but extract all necessary data from the splitting sentences. In the second corpus, it contains 52 texts (about 20,000 sentences) from the Institute of Mathematics and Statistics of the University of São Paulo. The sentences have been manually tagged with POS and syntactic features at the University of Campinas in the lines of the Penn-Helsinki Parsed Corpus of Middle English [15]; and 3) Portuguese newspaper corpus that we collected from the internet newspaper of *Hoje Macau*. The corpus consists of 817 newspaper articles and contains a mixture of different text genres including the stories from newswires, sport, government, fashion, and business text. This corpus is used to investigate and determine whether *i*Sentenizer is suitable for different genres. The relative data size of

Table 2. Size of data sets.

Corpus	Sentences	Tokens
Brown	51,682	1,155,242
Tycho Brahe	21,454	672,950
Hoje Macau	39,795	1,158,734

each corpus is given in Table 2.

As described before, our system does not use any additional linguistic information beside the characteristics of tokens that straddle the sentence boundaries, and the tags to indicate the true boundaries of sentences labeled by Y (boundary indicator) and N (non boundary indicator). Figure 1 gives an example of sentence labeled for our system.

... married Aug . 2, 1913. They have a son, William Berry Jr., and a daughter, Mrs. J. M. Cheshire of Griffin. Attorneys ...
... married/N Aug/N ./N 2/N ,/N 1913/N ./Y They/N have/N a/N son/N ,/N William/N Berry/N Jr/N ./N ,/N and/N a/N daughter/N ,/N Mrs/N ./N J/N ./N M/N ./N Cheshire/N of/N Griffin/N ./Y Attorneys/N ..

Figure 1. Sentence labeled with boundary tags.

Instead of feeding the list of sentences into the *i*Sentenizer directly, the features of potential punctuation are extracted as an instance and are described using an 12-dimension vector $<b_0, b_1, ..., b_{10}, l>$, where l is the label (Y/N) to indicate if it is a sentence separator or not. Figure 2 shows the examples of instances constructed based on potential trigram contexts.

Context:	'/N ./Y It/N
Instance:	<0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 3, 4, Y>
Context:	Aug/N ./N 2/N
Instance:	<1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 5, 3, N>
Context:	M/N ./N Cheshire /N
Instance:	<1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 3, 8, N>

Figure 2. Examples of data instances represented as feature vector.

7. Evaluation

We evaluate the *i*Sentenizer classifier using the metrics of *precision*, *recall*, and *F1 score*, as shown in Equations (4), (5), and (6) respectively:

$$pre = \frac{\text{true positives}}{\text{true positive + false positives}} \quad (4)$$

$$recall = \frac{\text{true positives}}{\text{true positives + false negatives}} \quad (5)$$

$$F1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision + recall}} \quad (6)$$

7.1. Experiment results

Like other machine learning based boundary detection systems, the construction of *i*Sentenizer has two steps, namely training and testing. The training phase is to create the boundary classifier using the training instances. In the evaluation, a test suit of samples that are not included in the training set are used to test the classifier. Table 3 illustrates the size of test corpora used for the evaluations.

Table 3. Size of data sets (no. of instances).

Corpus	Train Data	Test Data
Brown	55,243	6,139
Tycho Brahe	29,217	3,247
Hoje Macau	44,923	4,492

We train and test *i*Sentenizer on the corpora using the best set of features as described in section 5. The results *i*Sentenizer achieved on different corpora are given in Table 4. Testing on the Brown corpus yields very good precision of 99.98%. While the results obtained on Tycho Brahe and Hoje Macau corpora are slightly lower, with precisions of 96.51% and 98.73% respectively.

Table 4. Results of classification.

Corpus	Precision	Recall	F1
Brown	99.98%	99.60%	99.79%
Tycho Brahe	96.51%	98.37%	97.43%
Hoje Macau	98.73%	92.13%	95.31%

7.2. Incremental learning experiment

The second part of the evaluation, according to the nature of *i*Sentenizer, the classifier is able to incrementally learn from the new instances that have not been previously seen in the training set. The classifier learns by revising the original decision tree structure according to the contribution of new evidence of features from the unseen instances. In setting up the experiment, we collected the false instances that have been incorrectly classified by the *i*Sentenizer. Considering that this portion of data is far less than 0.01% of the training data set, we magnify it by replicating the instances five times. Then, ask the *i*Sentenizer to learn the new instances on the fly, and test again on the testing data. Table 5 presents the results obtained by the new *i*Sentenizer.

Table 5. Results of new *i*Sentenizer.

Corpus	Precision	Recall	F1
Brown	99.98%	99.61%	99.80%
Tycho Brahe	97.34%	98.97%	98.15%
Hoje Macau	98.13%	97.10%	97.61%

Reviewing the results on different corpora, we found no improvement in precision for the Brown corpus. It is reasonable as the system already gives a very high precision on it. For Tycho Brahe corpus, we see a slight improvement in precision (0.84%), recall (0.6%) and F1 score (0.72%). While for the Hoje Macau corpus, there is little reduction in precision (0.59%) and a significant improvement in recall (4.97%) and F1 score (2.3%).

7.3. Cross corpora evaluations

The results we presented in previous section show that *i*Sentenizer is able to lower the classification error and achieve a good performance by means of incrementally learning from the false cases. In order to demonstrate *i*Sentenizer is well-suit to process different text genres and domains, in this section, we train *i*Sentence on one corpus, and evaluate it on other corpora to investigate its adaptability to text of different Roman-alphabet languages. The experiments are carried out as followings: 1) *i*Sentenizer is trained on one corpus and evaluated using the test data sets from other corpora; 2) the classifier is revised on the fly using the falsely classified instances as additional training data and a new classifier (*i*Sentenizer+) is obtained. The learning process will repeat until no further reductions in error cases. Normally, the process will cease within five iterations; and 3) the whole experiment is repeated to train the classifier on the corpora of Brown, Tycho Brahe and Hoje Macau respectively.

Table 6. Trained on (English) Brown corpus.

Corpus	Precision	Recall	F1
*i*Sent_B_T	75.0%	98.7%	85.2%
*i*Sent_B+T	97.3%	98.7%	98.0%
*i*Sent_B_H	96.9%	96.1%	96.5%
*i*Sent_B+H	98.5%	94.1%	96.3%

Table 6 shows the first result of *i*Sentenizer that is trained on English Brown corpus and tested on Portuguese corpora of Tycho Brahe (*i*Sent_B_T) and Hoje Macau (*i*Sent_B_H). Each of the classifiers is then revised using the misclassified instances as additional training data from the corresponding corpus, and re-evaluated again on the respective corpora of Tycho Brahe (*i*Sent_B+T) and Hoje Macau (*i*Sent_B+H). We found that there is a significant improvement in precision (22.3%) and F1 score (12.7%) for the *i*Sent_B+T classifier. While the classifier *i*Sent_B+H shows a minor improvement in precision (1.7%), but slightly reduction in recall (2%) and F1 score (0.2%). However, the performances of the classifiers are comparable to the classifiers evaluated with the respective corpus, as described in section 7.1 and 7.2.

Table 7. Trained on (Portuguese) Tycho Brahe corpus.

Corpus	Precision	Recall	F1
*i*Sent_T_B	97.4%	95.8%	96.6%
*i*Sent_T+B	99.9%	97.9%	98.9%
*i*Sent_T_H	97.7%	89.6%	93.5%
*i*Sent_T+H	98.3%	97.4%	97.9%

Table 8. Trained on (Portuguese) Hoje Macau corpus.

Corpus	Precision	Recall	F1
*i*Sent_H_B	98.9%	96.0%	97.4%
*i*Sent_H+B	99.9%	97.7%	98.8%
*i*Sent_H_T	96.8%	98.6%	97.7%
*i*Sent_H+T	97.5%	98.6%	98.1%

Table 7 and Table 8 present the evaluation results of *i*Sentenizers that are trained on each Portuguese corpus and tested on the rest of the respective English and Portuguese text corpora. The results are very promising, and give around 1% to 2% improvement in precision, 2% to 7% in recall, and 1% to 4% in F1 score. The experiments fully reflect the advantage of *i*Sentenizer that the classifier can be gradually improved by incorporating the new instances and evidences into the active classifier.

8. Conclusions

In this paper, we have described an incremental learning approach to identify sentence boundaries which performs comparably to other sentence boundary detecting systems. For example, although Satz and MxTerminator systems are ready for different text genres and domains, they have to be retrained from scratch. In comparison, *i*Sentenizer can be easily adapted to different text genres in English as well as text in other alphabet languages on the fly, without giving up the previous constructed knowledge. Second, *i*Sentenizer does not require additional annotated information or any supporting resources beyond the sentence-split corpus.

Acknowledgements

This work was partially supported by the Research Committee of University of Macau under grant UL019/09-Y2/EEE/LYP01/FST, and also supported by Science and Technology Development Fund of Macau under grant 057/2009/A2.

References

[1] Gregory Grefenstette and Pasi Tapanainen, "What is a word, What is a sentence? Problems of Tokenization," in 3rd International Conference on Computational Lexicography, Budapest, Hungary, pp. 1-11, 1994.

[2] Carlos N. Silla Jr., Jaime Dalla Valle Jr., and Celso A. A. Kaestner, "Detecção Automática de Sentenças com o Uso de Expressões Regulares," in III Congresso Brasileiro de Computação (CBComp 2003), Itajaí, Brazil, pp. 548–560, 2003.

[3] Carlos N. Silla Jr. and Celso A. A. Kaestner, "An analysis of sentence boundary detection systems for English and Portuguese documents," in Fifth International Conference on Intelligent Text Processing and Computational Linguistics - Lecture Notes in Computer Science. vol. 2945: Springer, pp. 135-141, 2004.

[4] David D. Palmer and Matrti A. Hearst, "Adaptive Multilingual Sentence Boundary Disambiguation," Computational Linguistics, vol. 23, pp. 241-267, 1997.

[5] Jeffrey C. Reynar and Adwait Ratnaparkhi, "A Maximum Entropy Approach to Identifying Sentence Boundaries," in Fifth Association for Computational Linguistics Conference on Applied Natural Langauge Processing, Washington, DC, pp. 16-19, 1997.

[6] Tibor Kiss and Jan Strunk, "Unsupervised Multilingual Sentence Boundary Detection," Computational Linguistics, vol. 32, pp. 485-525, 2006.

[7] Sam Chao and Fai Wong, "An Incremental Decision Tree Learning Methodology regarding Attributes in Medical Data Mining," in Eighth International Conference on Machine Learning and Cybernetics (ICMLC 2009), Baoding, China, pp. 1694-1699, 2009.

[8] Helio Radke Bittencourt and Robin Thomas Clarke, "Use of Classification and Regression Trees (CART) to Classify Remotely-Sensed Digital Images," in International Geoscience and Remote Sensing Symposium (IGARSS '03), Toulouse, France, pp. 3751-3753, 2003.

[9] S. Rasoul Safavian and David Landgrebe, "A Survey of Decision Tree Classifier Methodology," IEEE Transactions on Systems, Man and Cybernetics, vol. 21, pp. 660-674, 1991.

[10] J. H. Friedman, "A Recursive Partitioning Decision Rule for Nonparametric Classification," IEEE Transactions on Computers, vol. C-26, pp. 404-408, 1977.

[11] Paul E. Utgoff and Jeffery A. Clouse, "A Kolmogorov-Smirnoff Metric for Decision Tree Induction," University of Massachusetts, Department of Computer Science1996.

[12] Paul E. Utgoff, Neil C. Berkman, and Jeffery A. Clouse, "Decision Tree Induction Based on Efficient Tree Restructuring," Machine Learning, vol. 29, pp. 5-44, 1997.

[13] Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini, "Building a Large Annotated Corpus of English: The Penn Treebank," Computational Linguistics, vol. 19, pp. 313-329, 1993.

[14] Charlotte Galves and Helena Britto, "A Construção do Corpus Anotado do Português Histórico Tycho Brahe: o sistema de anotação morfológica," in IV PROPOR, Evora: University of Evora, pp. 55-67, 1999.

[15] Helena Britto, Charlotte Galves, Ilza Ribeiro, Marina Augusto, and Ana Paula Scher, "Morphological annotation system for automatic tagging of electronic textual corpora: from English to Romance languages," in 6th International Symposium of Social Communication, Santiago de Cuba, pp. 582-589, 1999.