Summary

Automatic language identification in written text documents is an issue which deserves significant attention in the context of the ever-growing volume of web documents. This paper deals with language identification in the domain of web documents. The proposed system is built on Hidden Markov Models (HMMs) that enable the modeling of character sequences. To our knowledge the use of HMMs has not been widely examined in such a task. The aforementioned observation combined with the flexible stochastic properties of HMMs motivated us to conduct further research on this topic. A parallel structure of discrete HMMs is used in the training phase. During testing, a previously unseen document is divided into its sentences and each of them is independently characterized with respect to the language it is written in. For this purpose, proper HMM features are used. Several HMM parameters are examined and adjusted for better results. Experiments conducted on sentence-long documents, written in five European languages, have demonstrated high identification rates. Furthermore, HMMs allow for language tracking; that is language identification across the segments of a multilingual document. This is a promising application for the proposed method.

Language identification in web documents using discrete HMMs

A. Xafopoulos C. Kotropoulos * G. Almpanidis I. Pitas

Dept. of Informatics, Aristotle Univ. of Thessaloniki Box 451, Thessaloniki 54124, GREECE

Abstract

This paper deals with language identification in the domain of web documents. The proposed system is built on Hidden Markov Models (HMMs) that enable the modeling of character sequences. Furthermore, the use of HMMs provides the means for language tracking, that is, language identification across the segments of a multilingual document.

Key words: Statistical language identification, Web documents, Tourism domain, Language tracking, Discrete Hidden Markov Models (DHMMs)

1 Introduction

One of the first steps taken in order to understand a written text is to identify the language it is written in. Related areas of interest are authorship attribution, subject identification, and text summarization. Apart from language, other natural processes are also described by a string of characters, like genetic DNA sequences.

* Corresponding author. Tel.: +30-2310-99.8225; fax: +30-2310-99.8225; email: costas@zeus.csd.auth.gr

It is being argued that text-based language identification (TLI) is straightforward [1]. In this paper we revisit this task applied to web documents and we argue that improvements are still possible. Several difficulties arise when dealing with web documents. Firstly, the web documents contain additional information for the visual appearance of a web page, which may interfere with the text, especially in the case of a faulty page composition. Secondly, web documents may have textual information in a form that is useful when displaying the page, but is disorganized when the documents are considered consolidated texts (e.g., data formatted as lists). Moreover, spelling and syntax errors are more frequent in document collections from the web than in corpora constructed from texts extracted from books or newspapers. Another issue is that web documents do not use the same character encoding, that is, characters are not always represented as the same byte values, even when they are in the same language, due to the existence of quite a few different textual character encodings. Although *Unicode encoding* would help toward this direction, the web documents that do not follow this standard are still markedly numerous. Finally, the plethora of international terms and proper names occurring in web documents introduces an additional difficulty to the identification process.

To cope with some of the above difficulties, hidden Markov models (HMMs) are used for TLI, counting on the fact that there is a possibility to model the linguistic structure statistically. HMMs have been successfully applied in spoken language identification [2,3]. Our target is to test their application to text documents, and in particular, to documents that have been extracted from HTML pages, by establishing a correspondence between language characters and integer values. The latter implies the treatment of the language as a signal. The rationale behind opting for HMMs is that they capture the stochastic nature of language.

Our effort focuses on the achievement of high identification rates using a small corpus of web documents for training and testing. During training, HMMs for several languages are created from the training corpus. Accordingly, the computational requirements for

the training are small. During testing, each of the selected test documents is split into its sentences and identification rates are measured based on the results of the identification procedure on each sentence. The splitting into sentences is performed so that the extraction of statistics is facilitated. The size of test documents used is generally small due to the fact that the test documents are only sentence-long documents. It is worth mentioning that, by the term "sentence", we do not refer to a syntactically and grammatically correct sequence of vocabulary words but to a contiguous collection of words ending with a period that is possibly processed by some "cleaning" rules. More details about the experimental setup are given in Subsection 4.1.

Five languages that are members of the Indo-European family have been selected. Two of them, that is English and German, are members of the West Germanic group of the Germanic subfamily, while the rest three languages, that is French, Spanish and Italian, are members of the the Romance group of the Italic subfamily. When a full HTML document is provided for identification, a slightly different procedure than the aforementioned is followed. The document is not characterized at the sentence level. A parallel structure of HMMs keeps track of the language used in the document. In this way, a multilingual document can be characterized and further processed in more detail, taking into account language changes even within a sentence. The latter fact compensates for a possible lack of the period between the text segments in different languages, which is something not so unusual in the domain of web documents.

The outline of the paper is as follows. An overview of TLI is provided in Section 2. The application of discrete HMMs to TLI is presented in Section 3. Experimental results are reported in Section 4 and conclusions are drawn in Section 5.

2 Text-based language identification

TLI is treated as a *categorization task*. That is, given a collection of texts written in a number of known languages, the objective is to determine the language an input docu-

ment is written in. The decision is made upon document characteristics, usually at the word or character level [1]. Working at the character level generally seems to be a more robust approach than working at the word level. Another treatment is to consider TLI a clustering task. The difference between a text clustering and a text categorization task is that the former attempts to construct clusters of texts written in the same language, possibly without knowing the identity of each language a priori. In that sense it can be said that text categorization implies a supervised pattern recognition task, while text clustering an unsupervised one. An overview of TLI issues can be found in [1].

Several methods have been proposed for TLI. One of them is to use the vocabulary of each language and decide upon the number or percentage of words found in each vocabulary. However, this approach has difficulties in coping with inflected words, that is grammatical variations, or spelling errors. Variations of this method employ the most frequent words in each language or grammatical or function words like prepositions, determiners, pronouns and conjunctions [4]. Another extensively applied approach is the so-called character n-grams. The most frequent sequences of n characters, where n can take more than one values, are found from a training corpus. The number of occurrences of these sequences in the test document is used in the decision criterion either directly or through the formation of probability estimates [4,5]. In [6] the ranking of the most frequent character n-grams is used instead of their absolute frequencies. There is also the possibility of using word n-grams, where probabilities of word sequences of length n are estimated instead of character sequences. In the latter case, more training data and computational resources are required. Among the aforementioned techniques, that based on character n-grams appears to be the most flexible. Several other identification approaches are found in the literature. In [7], both character n-grams and words are considered features of a vector-space based categorizer. The relative entropy, also called Kullback-Leibler distance, is considered in [1,8]. The use of Markov models for TLI is considered in [9], while the application of decision trees is studied in [10]. Potential ap-

plications of TLI to web documents are closely related to cross-language information retrieval, construction of digital libraries, and machine translation of online texts.

A challenge in TLI is the characterization of a multilingual document. In this case, the points of language change have to be identified as well as the language used between them. Such a task becomes arduous, especially in web documents, where the period is not always present to signify the end of a sentence.

3 Discrete HMMs for text-based language identification

3.1 Hidden Markov Models

Hidden Markov Models (HMMs) [11] are statistical models of sequential data thoroughly investigated and used not only for speech recognition, but also for various applications including biometrics, biosciences, climatology, automatic control, communications, econometrics, handwritting and text recognition, signal processing, image processing, and computer vision. There are two general kinds of HMMs, the continuous (density) models and the discrete models depending on whether the observations modeled are continuous or discrete random variables, respectively. In our case discrete models are of interest, since we represent characters as non-negative integer-valued random variables. What is achieved by means of HMMs is that the model parameters are adjusted to specific inputs. Accordingly, a suitable representation of these inputs is obtained. The type of input we are interested in, is written text in one of the languages under investigation. The representation attained by the HMMs can be thought as the probability distribution of the language characters induced by text realizations from a given language. In all experiments, one discrete HMM (DHMM) is constructed per language.

The training of the just mentioned DHMM is done using portions of the corpus as observation sequences $\mathbf{O}_i$ containing integers. These sequences, which are more closely examined in Subsection 3.2, are iteratively presented to the DHMM an adequate number of times until a termination criterion is met. Having done this for as many DHMMs as the number

of languages under examination, the language identified in the test document is the one associated to the DHMM that best represents this document. Further details about the training and identification stages can be found in Subsection 3.3.

A DHMM contains a set of N interconnected states $S = \{s_1, s_2, \ldots, s_N\}$ and M (observation) symbols $V = \{v_1, v_2, \ldots, v_M\}$. At a given time instant t it can enter one of the states $q_t \in S$ by providing an observation (feature) vector $\mathbf{o}_t$ (with components v_i) as output of the entered state q_t. $\mathbf{o}_t$ is generated using an output (observation) discrete probability distribution $b_l(k) = b_{q_t}(\mathbf{o}_t) = P(\mathbf{o}_t = \mathbf{v}_k | q_t = s_l)$, $2 \leq l \leq N-1$ for the entered state. The probability of transition between the states is $a_{ij} = P(q_{t+1} = s_j | q_t = s_i)$, $1 \leq i, j \leq N$, for a first order HMM. Higher order HMMs can take into account more than one history instances. Unconnected states have $a_{ij} = 0$. The entire model can be written as $\lambda = (\mathbf{A}, \mathbf{B}, \pi)$, where $\mathbf{A}$ is the transition probability matrix, $\mathbf{B}$ is the output probability distribution matrix, and π denotes the initial state distribution $\pi_i = P(q_1 = s_i)$, $1 \leq i \leq N$.

In the case under study, two special states are used, an entry state s_1 and an exit state s_N, in addition to the other "standard" states. These special states are non-emitting, i.e. they do not have output probability distributions associated with them but only transition probabilities. The entry state is always the first state of the model: $q_1 = s_1$, $\pi_1 = 1$, $\pi_i = 0$, $2 \leq i \leq N$, and the exit state is the last state. The summation of the transition probabilities initiating from a given state equals one ($\sum_{j=1}^{N} a_{ij} = 1$, $1 \leq i \leq N$), except for the exit state, since there is no possible transition after the exit state ($a_{Nj} = 0$, $1 \leq j \leq N$). The value $j = 1$ in the summation can be disregarded, since there is no possible transition to the entry state ($a_{i1} = 0$, $1 \leq i \leq N$). We also used $a_{12} = 1$, meaning that the only transition from the entry state is toward the first standard state. In our experiments, three topologies were tested: left-right (LR) models without skips, where $a_{ij} = 0$, $j \neq i, i+1$, LR models with skips, where $a_{ij} = 0$, $j \neq i, \ldots, i + \Delta i$, and ergodic models, where $a_{ij} \neq 0$, $\forall i, j$ standard states. An LR model without skips including the

two non-emitting states is depicted in Figure 1.

A possible abstract explanation of the meaning of states is that of considering them groups of characters with common characteristics. For example, vowels and consonants could form two different groups. This association of the model states with group of characters can lead to the consideration of the transition probability matrix **A** as a bigram language model, while the output probability distribution matrix **B** can be considered a unigram language model. The formation of these groups is not a priori known and is left to be decided at the learning phase. The latter fact agrees with the theoretical assumption of DHMMs that the states are not directly observable but hidden, as opposed to Markov chains, i.e. the discrete-time discrete-space Markov processes. When using DHMMs, each observation is a probabilistic function of the state from which it is derived, while in Markov chains the observation is deterministically determined by the state.

3.2 Formation of observation vectors

The observation characters, that is those considered for the formation of observation (feature) vectors, are the 26 English alphabet letters ('a'-'z' or ENGset), the 32 ISO Latin-1 characters used in western European languages (ISO1set), the period ('.') and a symbol assigned for the blank space between words (' '), yielding an alphabet of size $M = 60$ observation symbols. The names of the ISO1set letters as they appear in HTML entities are: 'agrave', 'aacute', 'acirc', 'atilde', 'auml', 'aring', 'aelig', 'ccedil', 'egrave', 'eacute', 'ecirc', 'euml', 'igrave', 'iacute', 'icirc', 'iuml', 'eth', 'ntilde', 'ograve', 'oacute', 'ocirc', 'otilde', 'ouml', 'oslash', 'ugrave', 'uacute', 'ucirc', 'uuml', 'yacute', 'thorn', 'szlig', 'yuml'. No distinction between uppercase and lowercase types is made for the 26 English alphabet letters as well as for 30 ISO Latin-1 characters (the 32 previously mentioned ones except 'szlig' and 'yuml'). Furthermore, multiple blank space occurrences are considered a single blank space.

An observation vector can be regarded as the numerical representation of an observation

symbol at a specific time point. Several representations were considered based on the observation characters. The first is the character mapping according to their ISO Latin-1 code value. The resulting (representation) codes are hereafter called *symbol features,* f_{s}. In our case, the alphabet size for the symbol features is $M_{\mathrm{s}} = 60$. An observation symbol may include more than one observation characters, for example if differences of code values are considered. The resulting representation uses the value of the difference between the ISO Latin-1 code values of two characters. This value can be negative, zero-valued or positive, depending on whether the code value of the currently considered character is less than, equal to, or greater than the code value of the previous character, respectively. The observation vectors constructed as described are one-dimensional.

Depending on the proximity of the two characters taken into account a series of observation vectors results. Thus, when the value of the difference is regarded between two consecutive characters, the current character and the one on its left, the resulting codes are called *delta-zero features* $f_{\Delta 0}$. When the difference is regarded between two characters separated by another character, that is between the current character and the one two places leftward, the resulting codes are called *delta-one features* $f_{\Delta 1}$, and so forth, resulting in $f_{\Delta i}$ for i a non-negative integer. As for the indices, Δ denotes the difference. The index i on its right denotes the number of observation characters which are present in the corpus between the observation characters consisting an observation symbol. The alphabet size for these delta features is $M_{\Delta i} = 2M_{\mathrm{s}} - 1$ observation symbols, where $i \geq 0$. In the cases, where there are not enough previous characters for the computation of a difference, a value considering only the first character is computed.

The mapping function of the symbol features $\mathrm{m_s}(\cdot)$ is defined by:

$$\mathrm{m_s}(c) = \begin{cases} \mathrm{up}(c) - 64, & c \in \mathrm{ENGset} \\ \mathrm{m_s}('\mathrm{Z}') + 1\,(= 27) & c =' .' \\ \mathrm{m_s}('\mathrm{Z}') + 2\,(= 28) & c =' \ ' \\ \mathrm{up}(c) - 191 + \mathrm{m_s}('.') \ = \mathrm{up}(c) - 163 & c \in \mathrm{ISO1setA} \\ \mathrm{up}(c) - 192 + \mathrm{m_s}('.') \ = \mathrm{up}(c) - 164 & c \in \mathrm{ISO1setB} \\ 31 + \mathrm{m_s}('.') \ = 59 = M_\mathrm{s} - 1 & c =' \mathrm{szlig}' \\ 32 + \mathrm{m_s}('.') \ = 60 = M_\mathrm{s} & c =' \mathrm{yuml}' \end{cases}$$

where

$$\mathrm{up}(c) = \begin{cases} \mathrm{ISO1(uppercase\,c)}, & c\,\mathrm{is\,lowercase} \\ \mathrm{ISO1}(c), & \mathrm{c\,is\,uppercase} \end{cases},$$

ISO1(c) is the ISO Latin-1 code value for the character c, ISO1setA denotes the ranges 'Agrave-Ouml' and 'agrave'-'ouml', and ISO1setB the ranges 'Oslash'-'THORN' and 'oslash'-'thorn'.

The formula used for $\mathrm{m}_{\Delta i}(\cdot)$, that is the mapping function of the delta-i features, referring to two characters separated by i characters is

$$\mathrm{m}_{\Delta i}(b \overbrace{\ldots}^{i\,\mathrm{characters}} c) = \mathrm{m_s}(c) - \mathrm{m_s}(b) + M_\mathrm{s} \ = \mathrm{m_s}(c) - \mathrm{m_s}(b) + 60,$$

while the formula referring to a single character, at the beginning of a character sequence where the difference cannot be applied, is $\mathrm{m}_{\Delta i}(c) = \mathrm{m_s}(c) + M_\mathrm{s} \ = \mathrm{m_s}(c) + 60$.

The argument to the mapping function $\mathrm{m}_{\cdot}(\cdot)$ is an observation symbol, while the result

of this function is an observation vector. Through these functions the code values are uniquely mapped on a continuous integer range. In more detail, $\mathrm{m_s}(\cdot)$ maps first the ENGset on the range $[1, 26]$, then the period on 27 and the blank space on 28 and then the ISO1set on $[29, 60]$. $\mathrm{m}_{\Delta i}(\cdot)$ regards the differences of the values as considered in the ranges of m_s, which are just shifted by M_s so as to move the range $[-M_\mathrm{s}+1, M_\mathrm{s}-1]$ to $[1, 2M_\mathrm{s}-1]$.

For the calculation of the feature values, the following formulas are used, where $[\ldots]$ denotes concatenation:

$$\begin{aligned} f_\mathrm{s}(\mathrm{abc}) &= [\mathrm{m_s(a)}\,\mathrm{m_s(b)}\,\mathrm{m_s(c)}] \\ f_{\Delta 0}(\mathrm{abc}) &= [\mathrm{m}_{\Delta 0}\mathrm{(a)}\,\mathrm{m}_{\Delta 0}\mathrm{(ab)}\,\mathrm{m}_{\Delta 0}\mathrm{(bc)}] \\ f_{\Delta 1}(\mathrm{abc}) &= [\mathrm{m}_{\Delta 1}\mathrm{(a)}\,\mathrm{m}_{\Delta 1}\mathrm{(b)}\,\mathrm{m}_{\Delta 1}\mathrm{(ac)}] \\ &\vdots \end{aligned}$$

The resulting set from the mappings for a whole sequence of observation characters is called an observation sequence and the set of the mapping functions on a whole character sequence is referred to as features or codes.

Thus for a character sequence "acb",

$$\begin{aligned} f_\mathrm{s}([\mathrm{'a''c''b'}]) &= [\mathrm{m_s('a')}\,\mathrm{m_s('c')}\,\mathrm{m_s('b')}] = [1\,3\,2] \\ f_{\Delta 0}([\mathrm{'a''c''b'}]) &= [\mathrm{m}_{\Delta 0}\mathrm{('a')}\,\mathrm{m}_{\Delta 0}\mathrm{(''ac'')}\,\mathrm{m}_{\Delta 0}\mathrm{(''cb'')}] = [61\,62\,59] \\ f_{\Delta 1}([\mathrm{'a''c''b'}]) &= [\mathrm{m}_{\Delta 1}\mathrm{('a')}\,\mathrm{m}_{\Delta 1}\mathrm{('c')}\,\mathrm{m}_{\Delta 1}\mathrm{(''ab'')}] = [61\,63\,61]. \end{aligned}$$

In addition, multi-dimensional observation vectors were tested comprising of combinations of the aforementioned features, for example *symbol-delta-zero* $f_{\mathrm{s}\&\Delta 0}$ and *symbol-delta-zero-delta-one* $f_{\mathrm{s}\&\Delta 0\&\Delta 1}$. These features make concurrent use of the individual mappings of their constituent codes $(f_\mathrm{s}, f_{\Delta 0}, f_{\Delta 1}, \ldots)$. The latter are arranged in the so-called streams, which are regarded as independent information sources. Multiple data streams enable separate modeling of multiple sources. Each combined observation vector $\mathbf{o}_t$ of dimension $\widehat{S}$, uses $\widehat{S}$ features and can be thought of as a merging of the code values of its constituent

streams at time t. If an examined document consists of T observation characters an observation sequence can be regarded as an $\widehat{S} \times T$ vector which includes all the streams over the whole document. As an example the word "acb" is encoded as follows for the 60 observation characters used when having $f_{\mathrm{s}\&\Delta 0\&\Delta 1}$ as the feature:

$$\begin{aligned} f_{\mathrm{s}\&\Delta 0\&\Delta 1}([\mathrm{'a''c''b'}]) = [&f_{\mathrm{s}}(\mathrm{'a'})\, f_{\Delta 0}(\mathrm{'a'})\, f_{\Delta 1}(\mathrm{'a'})\,;\ f_{\mathrm{s}}(\mathrm{'c'})\, f_{\Delta 0}(\mathrm{''ac''})\, f_{\Delta 1}(\mathrm{'c'})\,;\\ &f_{\mathrm{s}}(\mathrm{'b'})\, f_{\Delta 0}(\mathrm{''cb''})\, f_{\Delta 1}(\mathrm{''ab''})]\\ = [&1\,61\,61\,;\ 3\,62\,63\,;\ 2\,59\,61] \end{aligned}$$

(ranges: 1-60 f_{s}, 1-119 $f_{\Delta 0}$, 1-119 $f_{\Delta 1}$). In this case, "abc" is a 9-dimensional observation sequence consisting of $\widehat{S} = 3$ streams or codes, where each of them is a 3-dimensional ($T = 3$) observation vector.

3.3 Learning and identification processes

As for the learning process, there are two main optimization criteria: Maximum Likelihood (ML) and Maximum Mutual Information (MMI). In ML, the probability of a given observation sequence $\mathbf{O}_i$, belonging to a given category (language), is maximized, given the model parameters λ of the category. The ML criterion can be expressed mathematically as the maximization of $P(\mathbf{O}_i|\lambda)$, where:

$$P(\mathbf{O}_i|\lambda) = \sum_{s_1,\ldots,s_N} \pi_1 \prod_{t=2}^{T} a_{s_{t-1}s_t} b_{s_t}(\mathbf{o}_t)$$

There is no known way to analytically determine the model parameters λ which maximize $P(\mathbf{O}_i|\lambda)$. Nevertheless, model parameters such that $P(\mathbf{O}_i|\lambda)$ is locally maximized can be chosen, using an iterative procedure, like the Baum-Welch method or a gradient based method [11]. In contrast to ML, where an HMM of only one category at a time is maximized, keeping HMMs for other categories intact at that time, in MMI, the concept of discriminative training is applied. That is, the HMMs of all the categories are trained simultaneously, so that the parameters of the correct model are updated to enhance its contribution to the observations, while the parameters of the alternative models are updated to reduce their contributions.

In our experimentation, the ML criterion is used. The general learning proceeds as follows: Initially, all symbols in each standard state are set to be equally likely. That is, $b_j(k) = \frac{1}{M}$, $1 \leq j \leq M$, $2 \leq k \leq N-1$. Let D denote the dimension of the vector $\mathbf{o}_t$, constant over time. At the initialization phase, the observation vectors $\mathbf{o}_t = (o_{t1}, o_{t2}, \ldots, o_{tD})^T$, for the different values of t, are presented to the corresponding DHMM and they are segmented uniformly, in a timely fashion, among its (standard) states s_i. The vectors assigned to a state are considered to be generated by this state. Having done this for every observation sequence $\mathbf{O}_i$, maximum likelihood probability estimates are found for the model parameters $\lambda = (\mathbf{A}, \mathbf{B}, \pi)$, so that $P(\mathbf{O}_i|\lambda)$ is maximized. The *Viterbi alignment* [12] is used next for the resegmentation of the observation vectors and the recomputation of the estimates of the transition probability matrix $\mathbf{A}$ and the output probability distribution matrix $\mathbf{B}$, until convergence is achieved, for example when having reached a maximum $P(\mathbf{O}_i|\lambda)$, or when an upper iteration limit is reached. The transition probabilities at each iteration are estimated by counting the number of times each transition is made in the Viterbi alignments and normalization. Lateron, there is the possibility of using a re-estimation phase for assigning each observation vector to every state in proportion to the probability of the model being in that state when the vector was observed. The probability of state occupation is calculated efficiently by means of the forward-backward algorithm. The whole process is called *Baum-Welch re-estimation* or otherwise EM (Expectation-Maximization) algorithm [11]. Experimental evidence showed that the application of the Baum-Welch re-estimation leaves the results almost unmodified and was subsequently omitted. This is probably due to the effective usage of the Viterbi alignment.

At the identification stage, the *Viterbi decoding* is used. The purpose of this algorithm is to find the best path of state transitions $\mathbf{q} = (q_1, q_2, \ldots, q_T)$, given the observation vectors $\mathbf{o}_t$ and the model parameters λ. The log probability of a path is computed by taking the summation of the log output probabilities and the log transition probabilities. For LR models without skips, the Viterbi decoding does not determine the order of the

succesion of states, which is known due to the sequential nature of the model, but the number of times each state is used as a generator. Internally the token passing algorithm is used [12].

For the implementation of TLI, a parallel structure of the trained DHMMs is constructed. For each DHMM, there is a probability outcome computed for the best path that was found by the Viterbi algorithm. The identification result is found by choosing, among the DHMMs, the one that is more likely to have produced the test observation sequence $\mathbf{O}_{\text{tst}}$. This approach is also used in isolated word recognition using HMMs [11]. The HMM that maximizes the probability of $\mathbf{O}_{\text{tst}}$ when the parameters of the HMM (λ) are given is chosen and, since each HMM represents one language, this is declared as the identified language. For the implementation of the learning and the identification phases HTK toolkit version 3 [12] is used. The identification process is depicted in Figure 2.

3.4 Model refinement

In order to achieve the best results, several model parameters were investigated leading to a refined model. By keeping the remaining parameters unmodified, the experiments gave some indications about the investigated parameters. Experimental evidence showed that the average performance deteriorates when including characters than those mentioned in Subsection 3.2 as observation characters, or when distinguishing between lowercase and uppercase letters.

As for the features used, there are two parameters: their identity and their number. We considered all 15 possible combinations of $f_{\text{s}}, f_{\Delta 0}, f_{\Delta 1}$, and $f_{\Delta 2}$ in which each of these features is regarded as a different stream. The experimental results show that the inclusion of more delta features than those considered ($f_{\Delta 0}, f_{\Delta 1}, f_{\Delta 2}$) would not justify the additional complexity. $f_{\text{s\&}\Delta 0}$ also appears to be a particularly good choice when viewed in terms of both identification rate and simplicity of implementation. Identification rates with respect to training size of about 22500 characters, test sentences of length 20-

100 characters and equal stream weighting (the notion of stream weighting is explained subsequently) are given in Table 1. Moreover, the order of the constituent features did not change the results, as expected, while the combined features include a small time overhead. The rates presented in the tables of this paper are rounded to the closest integer.

Another parameter examined was the selection of stream exponents [13], which can be used to weight the influence of each stream on the calculation of the probability of an observation sequence, when multiple streams are used. When considering separate independent streams, the overall probability of an observation sequence becomes:

$$P(\mathbf{O}_i|\mathbf{q},\lambda) = \prod_{t=1}^{T} P(\mathbf{o}_t|q_t,\lambda) = \prod_{t=1}^{T}\prod_{s=1}^{S} P_s^{\gamma_s}(\mathbf{o}_t^{(s)}|q_t,\lambda),$$

where independence of the observation vectors $\mathbf{o}_t$ is assumed. An observation vector at the discrete time t consists of its stream components $\mathbf{o}_t = [\mathbf{o}_t^{(1)}\ \mathbf{o}_t^{(2)}\ \cdots\ \mathbf{o}_t^{(S)}]^T$, S denotes the number of streams, as they are defined in Subsection 3.2, and γ_s denote the stream exponents. By taking logarithms, the above probability becomes:

$$\log P(\mathbf{O}_i|\mathbf{q},\lambda) = \sum_{t=1}^{T}\sum_{s=1}^{S} \gamma_s \log P_s(\mathbf{o}_t^{(s)}|q_t,\lambda).$$

Two usually applied constraints are $0 \leq \gamma_s \leq 1$ and $\sum_{s=1}^{S}\gamma_s = 1$. Finding an optimization criterion to evaluate the values of γ_s which maximize the above log probability has been researched quite much. Even state dependency was assumed. We have not taken into account the state dependency, but restricted ourselves to tied exponents at the global level [13]. In [14] the use of gradient descent is proposed for this purpose. Direct use of ML estimation leads to one exponent being 1 and the rest 0, which is unwanted. Alternative constraints for ML are presented in [15], such as $\sum_{s=1}^{S}(\gamma_s)^m = K$, leading to reestimation formulas for the exponents. Another approach is offered by discriminative training techniques, like MMI [16] and the minimum classification error (MCE) method [13]. Several combinations of exponents were tested yielding the one employing equal weights, whose

sum amounts to 1, as the choice with the highest identification rate. The latter was used in the subsequent experiments.

Furthermore, multiple observation sequences were constructed by segmenting the training part of each language into more parts of virtually equal length, not necessarily ending with a period. In this fashion, the different parts of text written in a language were considered to represent this language. This can be likened to the way that the several spoken utterances of a word are considered, when this word is to be learned in training. The significance of multiple observation sequences in training is also mentioned in [11]. The segmentation into observation sequences yielded slightly higher rates when more training data were available, whereas in the case of few training data, the rates were lower than those attained without segmentation. The latter fact is clarified in Table 4, where 1 and 4 observation sequences are used.

As for the topology, three kinds of DHMMs were tested, namely LR models without skips, LR models with skips, and ergodic models. The former yielded better results and were therefore selected for the subsequent experiments. A possible explanation of this fact is that fewer parameters are needed for the LR models without skips, compared to the other two models in terms of transition probabilities [3]. Although models with skips have given higher identification rates than models without skips when few training data were used, the use of skips or ergodic models appears to produce overfitting. As an example, we mention the rates (rounded) from these three different models for a range of small sentences in Table 2. Having setup the experiments in a certain way, experimental evidence showed that the best case overall is to use three standard states, thus having $N = 5$ total states per DHMM.

3.5 Language tracking in multilingual documents

An application area which is successfully confronted by the proposed technique, as opposed to other standard techniques used for TLI, is language tracking. That is, in the

presence of multilingual documents, the identifier manages to pinpoint the location in the document, where a transition from one language to another occurs, apart from determining the identity of the language, thus presenting a flexibility with respect to transition from one language to another. This is achieved in a manner similar to the one used in speech recognition, where the detection of the end of one word and the beginning of the next one is performed. That is, instead of selecting a simple DHMM for the description of the entire document, a sequence of DHMMs is selected. The latter can be considered a network of DHMMs at a higher level than the network formed inside each DHMM with respect to its states. A decision for the language is made every as many characters as the number of standard states of each DHMM. This is explained by the fact that each observation vector, which is produced for each observation character, is generated by a DHMM state. Since a sequence of DHMMs is adjusted to the document being examined and there is a decision by every DHMM, it transpires that a decision is made every so many characters as the emitting states in a DHMM, that is $N - 2$ in our case.

4 Experimental Results

4.1 Experimental Setup

In order to test the performance of the suggested TLI technique, 151 HTML pages p_i were manually collected over the Internet, so that a small tourist corpus is created. Pages with "running" text were preferred, avoiding those full of short lists, price tables or abbreviations. We also tried to find monolingual documents with little text in other languages, but without requiring all the text to be strictly in one language. Most documents were collected from hotel promotion sites (e.g. `www.rosciolihotels.com`, `www.venere.com`, `www.hotels.fr`). Moreover, we exploited the fact that some web sites provide their context in more than one language. At its current state, the corpus consists of web pages related to hotels and accommodation written in five languages: English, German, French, Spanish, and Italian, with ISO 639-1 or alpha-2 codes "en", "de", "fr", "es", and "it", respectively. The corpus is used for both training purposes, that is the construction of statistical

language models, and test purposes, that is the evaluation of the language identification technique. Approximately thirty documents were collected for each language. The total size in characters of the texts per language is given in Table 3. Although the size of the corpus may seem to be small, experimental evidence showed that convergence to high identification rates is attained after a certain training size. Small sizes, on the order of tenths or hundreds of kilobytes, are also used in other similar efforts, such as [6].

The documents selected were annotated so that ground truth is incorporated. There are five categories L_i used for the annotation, one for each of the five languages considered. The annotation was not performed in detail. That is, each sentence s_{pi} of an entire document p was considered to be in the same language, $s_{pi} \in L_j \, \forall i$, as the one that was assigned to the document ($p \in L_j$). Nevertheless, there were actually few sentences s_{pk} written in a different language than the language of the document they were part of. This fact introduces slight errors in the training and the evaluation processes.

After having annotated the collection of the entire multilingual corpus C, the documents formed five groups C_l according to their language l. For each group, the procedure depicted in Figure 3 is followed to segment it into the training part $C_{l(\mathrm{trn})}$ and the test part $C_{l(\mathrm{tst})}$. A hold-out estimation approach was followed. More specifically, firstly, an initial cleaning process is executed for the removal of HTML structures and other useless data in p_i, while all ISO Latin-1 representations are converted to one. Plain documents $p_{i(\mathrm{pln})}$ result as an outcome of this procedure. After this step, the separation of $p_{i(\mathrm{pln})}$ into documents used for training $p_{i(\mathrm{pln})(\mathrm{trn})}$ and others used for testing $p_{i(\mathrm{pln})(\mathrm{tst})}$ is made. A sufficient number of documents $N_{l(\mathrm{trn})} = 20$ is selected to be included in the training corpus of each language and the remaining $N_{l(\mathrm{tst})} \simeq 10$ documents are selected for inclusion in the test corpus. The plain train and test documents are, in turn, split into their "sentences", thus forming a training $s_{li(\mathrm{trn})}$ sentence pool and a test $s_{li(\mathrm{tst})}$ sentence pool per language. By this split, the evaluation of identification rates for different ranges of sizes of test documents is facilitated. Sentences with less than 20 characters are omitted.

As for the "sentence" notion, it should be noted that the sentence splitting in this context is performed in a wide sense and not in a strict grammatical sense. Generally, sentences are phrases delimited by a period except for certain cases, which include emails, URLs and acronyms, where the period does not signify the end of a sentence. When a period is followed by a letter, which in turn is followed by a period and so on, the whole string is considered an acronym inside a sentence. To ensure that a sentence does not expand over more than one HTML document, an extra period is added at the end of each HTML page when merging takes place.

In our work "clean text" characters are used. By this term we mean the characters that are used in feature construction, i.e., the symbols referred in Subsection 3.2, except the ones that are included in emails, URLs, acronyms and a small stoplist (containing currency acronyms). The stoplist contains some frequently used tourism and currency related words that are found in the corpus documents irrespective of the language. An algorithm for the detection of URLs, emails and acronyms is used so that these do not interfere with the linguistic content of the documents. Finally, if 75% or more of the words in a sentence are "only-first-capital" ones, that is, begin with a capital letter and do not contain other capital letters, the latter words are not taken into account for feature extraction. This fact compensates for the inclusion of sentences in a list form, where many proper names are enumerated.

The training corpus $C_{(\mathrm{trn})}$ was built by five texts $C_{l(\mathrm{trn})}$, one for each language l. Each text is created as follows. First, from the textually ordered sentences of the training sentence pool for the corresponding language l, $s_{li(\mathrm{trn})}$, those that lie in a specific range of lengths of "clean text" characters are selected. The range of sentence lengths considered is $20 - 100000$. Thus, practically all sentences over 20 "clean text" characters are included. Afterward, these sentences are sampled taking one out of r_{samp} sentences. We experimented with $r_{\mathrm{samp}} = 1$ (all samples) and $r_{\mathrm{samp}} = 10$, so that a comparison for the training corpus size effect on the identification rates is enabled. The resulting sentences

are merged into $C_{l(\mathrm{trn})}$. Finally, the sizes of the five $C_{l(\mathrm{trn})}$ are shortened to the length of the smallest text by omitting the last extra characters, in order to provide the same amount of training material for each language. The fact that the training corpus may include some text that is not characteristic of the language structure, or the fact that a very small part of it may even be in a different language, since the pages were not manually modified, may be the source of wrong identifications. One reason for not performing or requesting a manual intervention is that the latter is usually onerous and it may also be subjective.

On the other hand, the test corpus $C_{(\mathrm{tst})}$ consists of five groups $C_{l(\mathrm{tst})}$ of separately considered sentences s_{li}, one per language l. From the textually ordered sentences of the test sentence pool for language l, $s_{li(\mathrm{tst})}$, only those that lie in a specific range of lengths in "clean text" characters are selected. Sentences with the same "clean text" content are detected and taken into account only once. An identification result is extracted for each sentence-long "document" $s_{li(\mathrm{tst})}$ of $C_{(\mathrm{tst})}$.

Using $C_{(\mathrm{trn})}$ the training observation vectors and sequences are created and 5 DHMMs are trained, one per language as described in Section 3. Subsequently, $C_{(\mathrm{tst})}$ is used to implement the evaluation procedure. For each sentence of the 5 languages, the observation vectors are extracted and they are presented to the structure of DHMMs in parallel to make the decision for the text language.

4.2 Performance Evaluation

Our experiments were focused on the evaluation of the effect of various parameters on the identification rates and also the comparison of our technique with a standard technique using variable character n-grams, as described in [6]. The same preprocessing steps were followed for both techniques, as described in Subsection 4.1. Table 4 was created using $f_{\mathrm{s\&\Delta0\&\Delta1\&\Delta2}}$ as features for DHMMs. DHMMs/1 refers to DHMMs where one observation sequence is used for training, while DHMMs/4 denotes DHMMs which use four obser-

vation sequences for training. By comparing the entries of these tables we conclude the following:

- As for the size of the training corpus, it appears that the bigger the size is, the higher identification rates are achieved. Nevertheless, this dependency does not seem to be very strong. A tenfold training size results in an average increase of two or three percent in the identification rates.
- On the other hand, the average length of the test sentences appears to play a significant role, since its increase leads to a significant increase in the identification rates. A shift of the average length range from 20–100 to 50–150 causes at least 3% increase in the identification rate.
- Although the variable character n-grams attain better results, there is a very small difference in the identification rates, of about one percent overall.
- What is more, in some cases regarding French and Italian texts, the DHMMs yielded higher rates. This enables the consideration of a hybrid technique for better results [17,18].

Experiments using only non-tourism related plain text documents for the training, and tourism related test documents gave worse average results for both techniques (DHMMs, n-grams). Finally, it should be noted that the reported identification rates include a slight error percentage, since the ground truth may contain small errors, mainly due to rare small English phrases regarding the navigation of the web page.

Confusion matrices were also computed, which provide with a better insight into the wrong identifications made per language. These confusion matrices are defined as follows: entry (i, j) is equal to the number of sentences assigned to output (language) category i, that were generated as part of input (language) category j. The last row gives the total number of sentences generated as part of input (language) category j. For the smallest range of test sentences examined, that is 20–100 characters long, the confusion matrices for DHMMs and variable character n-grams are given in Table 5. As can be

seen, there are cases for which DHMMs behave better. For example, there are more Italian texts categorized as Italian, fewer Italian texts categorized as English, as well as fewer German, Spanish, and Italian texts categorized as French. Additionally, the wrong classifications appear to be located in similar areas of the two matrices. This, in a sense, reveals relationships across languages, such as Spanish with Italian, English with German, and French with Spanish. The latter relationships are in agreement with the classification into language subfamilies, as noted in Section 1.

For further evaluation, our method was tested with full HTML documents and language tracking was performed. First, the HTML document is cleaned in the way that was described in the corpus preparation procedure. For every $N-2$ characters long segment of the cleaned document, a language identification result was found, taking its context into account. The latter is achieved using a network of DHMMs as described in Subsection 3.5. The way that each sentence segment is handled, in conjunction with its context, is what makes the use of DHMMs most suited than other techniques for language tracking. In the final stage, these results are merged into a language annotated document using an XML 1.0 conformant attribute *xml:lang*. In the latter document, whenever a group of more than one contiguous segments is categorized into the same language, the annotation is given for the entire group. The results of a few examples tested, including some plain text documents, were satisfactory, showing signs of generalization. One such example is given in Figure 4.

5 Conclusions and Future Work

We demonstrated by experiments that the use of DHMMs for TLI on web documents attains high recognition rates, taking into account the inherent domain difficulties. Therefore it can be regarded as a viable alternative of the other techniques used for TLI. The method does not require any linguistic or a priori knowledge of the corpus under investigation. Judging from some non-overlapping wrong results, especially for the smaller ranges of lengths of test sentences, there seems to be room for a hybrid technique be-

tween DHMMs and variable character n-grams, which would yield better results than either of them. The proposed method also works satisfactorily in a language tracking framework, when applied in multilingual documents. Further evaluation of the method can be achieved by using multifold cross-validation or bootstrapping [17]. The method should also be tested on other corpora.

Acknowledgements

This work was supported by the European Union IST Project "Hypergeo: Easy and friendly access to geographic information for mobile users" (IST-1999-11641).

References

[1] P. Sibun, J. Reynar, Language identification: Examining the issues, in: Proc. 5th Annual Symp. Document Analysis and Information Retrieval (SDAIR'96), 1996, pp. 125–135.

[2] Y. Muthusamy, E. Barnard, R. Cole, Reviewing automatic language identification, IEEE Signal Processing Mag. 11 (4) (1994) 33–41.

[3] M. Zissman, Automatic language identification using Gaussian mixture and hidden Markov models, in: Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP'93), Vol. 2, 1993, pp. 399–402.

[4] G. Grefenstette, Comparing two language identification schemes, in: Proc. 3rd Int. Conf. Statistical Analysis of Textual Data (JADT'95), 1995.

[5] H. Combrinck, E. Botha, Text-based automatic language identification, in: Proc. 6th Annual South African Workshop Pattern Recognition, 1995.

[6] W. Cavnar, J. Trenkle, N-gram-based text categorization, in: Proc. 3rd Annual Symp. Document Analysis and Information Retrieval (SDAIR'94), 1994, pp. 161–175.

[7] J. Prager, Linguini: Language identification for multilingual documents, in: Proc. 32nd Annual Hawaii Int. Conf. System Sciences (HICSS-32), 1999.

[8] D. Benedetto, E. Caglioti, V. Loreto, Language trees and zipping, APS Physical Review Letters 88 (4) (2002) 048702.

[9] T. Dunning, Statistical identification of language, Tech. Rep. CRL MCCS-94-273, New Mexico State University, Las Cruces, NM (1994).

[10] J. Häkkinen, J. Tian, N-gram and decision tree based language identification for written words, in: Proc. Automatic Speech Recognition and Understanding Workshop (ASRU'01), 2001.

[11] L. Rabiner, B.-H. Juang, Fundamentals of Speech Recognition, Prentice Hall, Englewood Cliffs, NJ, 1993.

[12] S. Young, D. Kershaw, J. Odell, D. Ollason, V. Valtchev, P. Woodland, The HTK Book (for HTK version 3.0), Microsoft Corporation, 2000.

[13] G. Potamianos, H. Graf, Discriminative training of HMM stream exponents for audio-visual speech recognition, in: Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP'98), Vol. 6, 1998, pp. 3733–3736.

[14] I. Rogina, A. Waibel, Learning state-dependent stream weights for multi-codebook HMM speech recognition systems, in: Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP'94), Vol. 1, 1994, pp. 217–220.

[15] J. Hernando, Maximum likelihood weighting of dynamic speech features for CDHMM speech recognition, in: Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP'97), Vol. 2, 1997, pp. 1267–1270.

[16] Y.-L. Chow, Maximum mutual information estimation of HMM parameters for continuous speech recognition using the N-best algorithm, in: Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP'90), Vol. 2, 1990, pp. 701–704.

[17] A. Jain, R. Duin, J. Mao, Statistical pattern recognition: A review, IEEE Trans. on PAMI 22 (1) (2000) 4–37.

[18] T. Ho, G. Nagy, Multiple classifier combination: Lessons and next steps, in: A. Kandel, H. Bunke (Eds.), Hybrid Methods in Pattern Recognition, World Scientific, 2002.

List of Figures

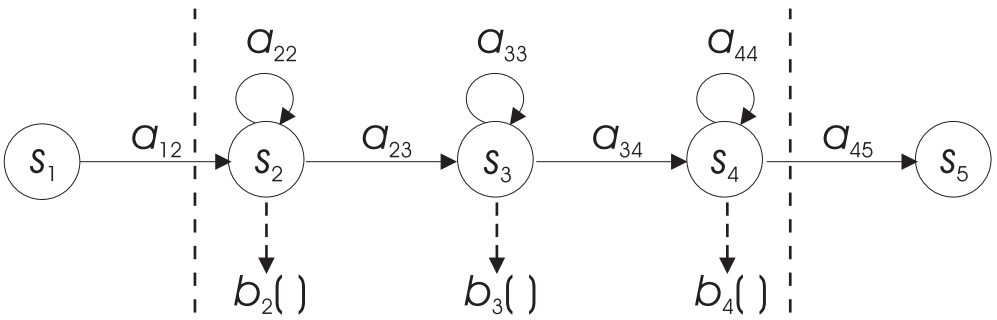

Figure 1. An LR DHMM without skips, having two non-emitting states.

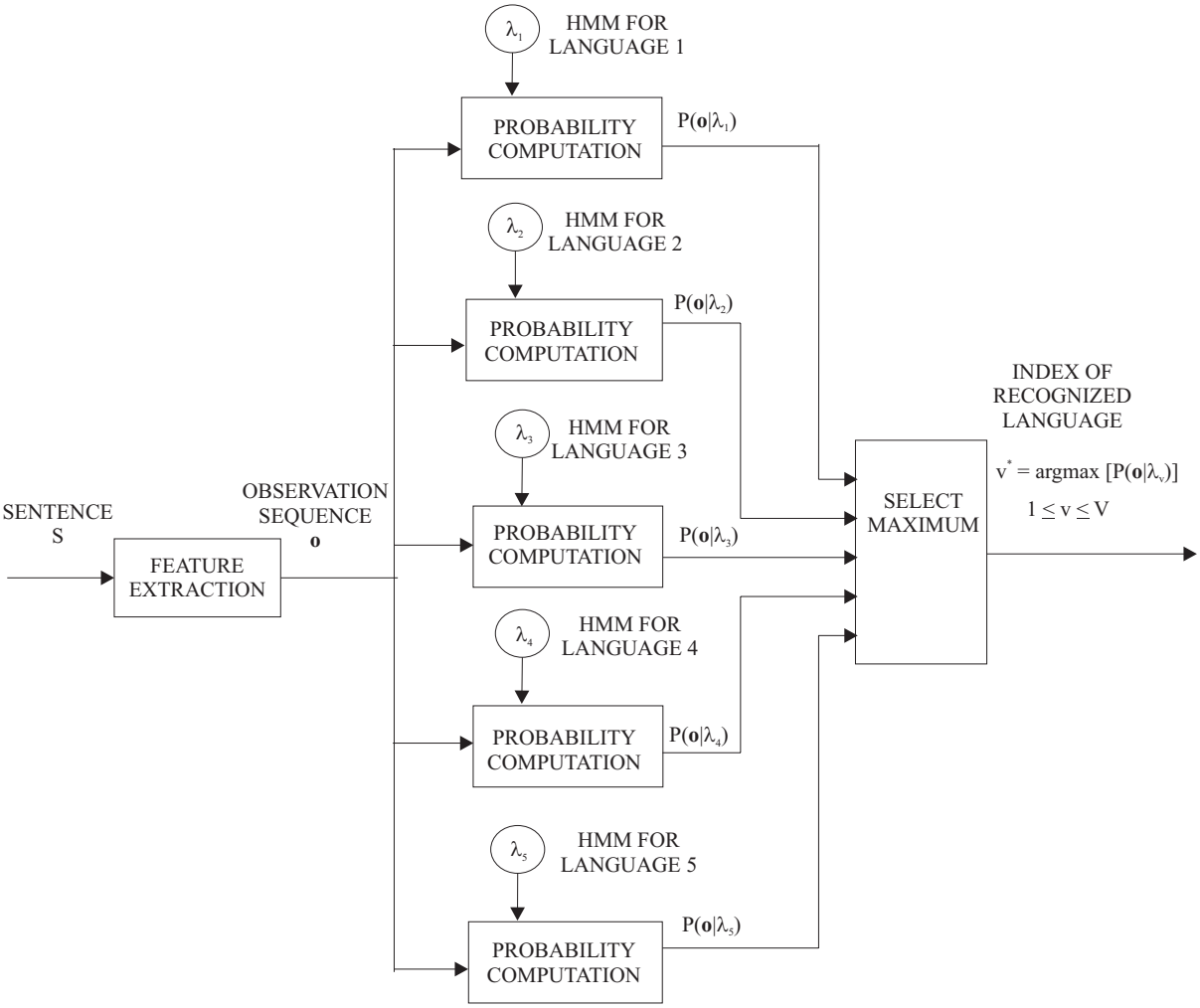

Figure 2. Block diagram of the identification stage of a language identifier using DHMMs.

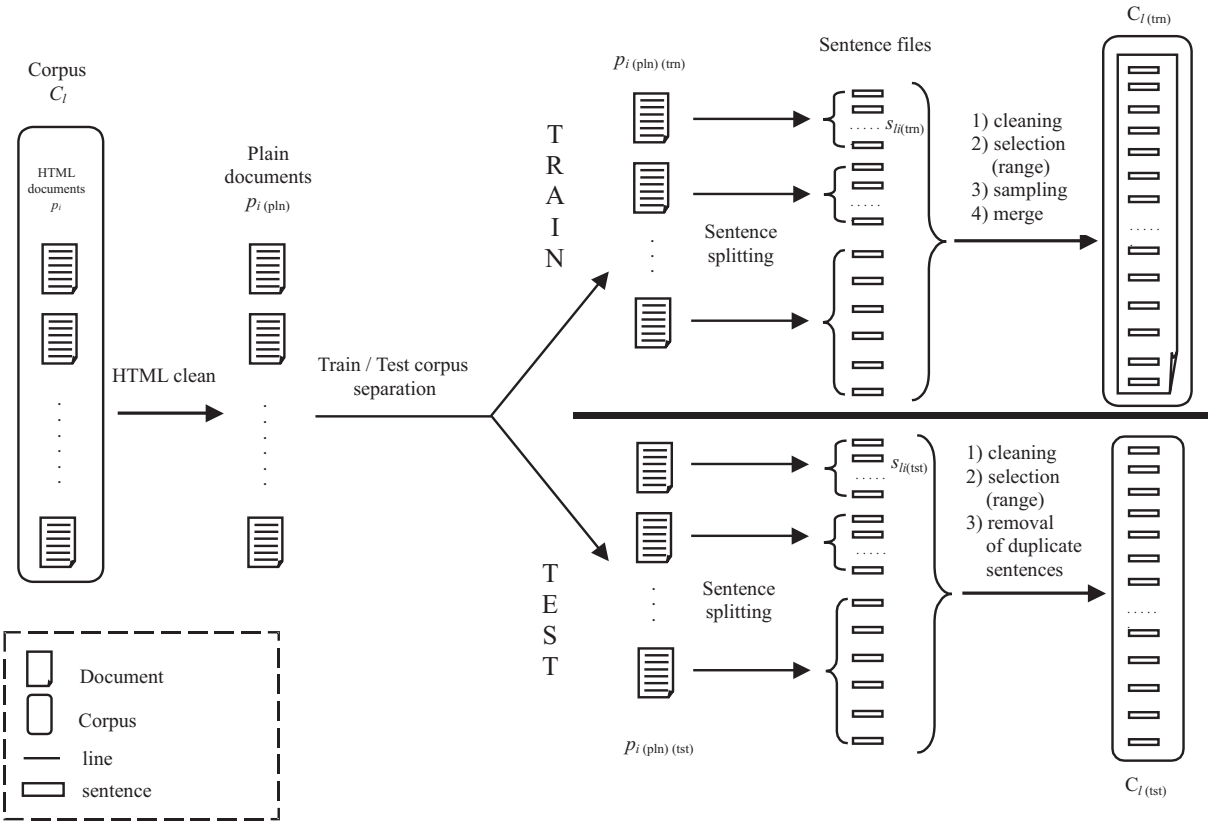

Figure 3. Block diagram of the segmentation of the corpus into the training and test part.

Hotel Gabriella Presentazione L Hotel Gabriella accogliente albergo a conduzione familiare è situato nel centro storico della città. Le camere tutte recentemente ristrutturate sono provviste di servizi privati tv color cassette di sicurezza aria condizionata su richiesta asciugacapelli e telefono. Inoltre l albergo dispone di bar con servizio di prima colazione a buffet. Prenotazioni per gite turistiche per la città e dintorni. Si accettano carte di credito e traveller s cheque . The Hotel Gabriella is a comfortable family run establishment located in the town s old centre. the rooms which have all recently been restructured have a private bathroom colour television strong box air conditioning on request hair dryer and telephone. What is more the hotel has a bar with breakfast and buffet service. Tours of the town and surrounding area can be booked here. Credit cards and travellers cheques welcome.

Figure 4. The output of the algorithm (DHMMs/1) having as input the multilingual web page `http://www.expoitalia.com/hotelgabriella/presentazione.htm`. The language change was detected with great precision.

List of Tables

Table 1
Encoding methods and obtained identification rates for a case study.

Feature	f_{s}	$f_{\Delta 0}$	$f_{\Delta 1}$	$f_{\Delta 2}$	$f_{\mathrm{s}\&\Delta 0}$	$f_{\mathrm{s}\&\Delta 1}$	$f_{\mathrm{s}\&\Delta 2}$	$f_{\Delta 0\&\Delta 1}$	$f_{\Delta 0\&\Delta 2}$	$f_{\Delta 1\&\Delta 2}$
Rate	88%	88%	79%	63%	94%	92%	87%	93%	90%	81%

Feature	$f_{\mathrm{s}\&\Delta 0\&\Delta 1}$	$f_{\mathrm{s}\&\Delta 0\&\Delta 2}$	$f_{\mathrm{s}\&\Delta 1\&\Delta 2}$	$f_{\Delta 0\&\Delta 1\&\Delta 2}$	$f_{\mathrm{s}\&\Delta 0\&\Delta 1\&\Delta 2}$
Rate	94%	93%	93%	92%	95%

Table 2
Average identification rate in five languages. The training sentences have at least 20 characters and their total size is 22446 characters for each language. The test sequences have 20–100 characters. The number after the first slash denotes the number of observation sequences used for training, while the expression after the second slash denotes the topology used.

Method	Identification rate
DHMMs/4/LR without skips	95%
DHMMs/4/LR with skips	86%
DHMMs/4/ergodic	82%

Table 3
Size (in characters) for each language in the small multilingual tourist corpus collected over the Internet. The corpus is used for both training and testing. The size refers to characters observed in texts after the initial cleaning process.

Language code	Size (characters)
en	50680
de	57083
fr	46774
es	58401
it	42766
Total	255704

Table 4
Average identification rate in five languages without and with (1 out of 10 training sentences) sampling. The training sentences have at least 20 characters and their total size is 22446 (without sampling) and 2098 (with sampling) characters for each language. For the DHMMs the number after the first slash denotes the number of observation sequences used for training.

No sampling

Method	Range of test sequence length in characters			
	20–100	100–200	50–150	20–200
	(avg) 68	140	100	101
	(total) 19507	30941	33729	50145
	Identification rate			
DHMMs/1	95%	99%	98%	96%
DHMMs/4	95%	99%	98%	97%
Var. n-grams	95%	100%	99%	97%

Sampling (1/10)

Method	Range of test sequence length in characters			
	20–100	100–200	50–150	20–200
	(avg) 68	140	100	101
	(total) 19507	30941	33729	50145
	Identification rate			
DHMMs/1	92%	99%	96%	94%
DHMMs/4	87%	97%	94%	92%
Var. n-grams	93%	99%	98%	96%

Table 5
The confusion matrices for test sentences in the range of 20–100 characters using DHMMs/1 and variable character n-grams, respectively.

DHMMs/1

Identified language	Ground truth language				
	en	de	fr	es	it
en	34	8	0	3	0
de	0	113	0	0	0
fr	0	0	20	2	0
es	0	0	0	62	2
it	0	2	0	4	55
Ground truth total # of sent.	34	123	20	71	57

Variable character n-grams

Identified language	Ground truth language				
	en	de	fr	es	it
en	34	2	0	2	1
de	0	116	0	0	0
fr	0	4	20	3	2
es	0	0	0	62	0
it	0	1	0	4	54
Ground truth total # of sent.	34	123	20	71	57