N-GRAM AND DECISION TREE BASED LANGUAGE IDENTIFICATION FOR WRITTEN WORDS

Juha Häkkinen[1] *and Jilei Tian*[2]
[1] Nokia Mobile Phones, Tampere, Finland
[2] Nokia Research Center, Tampere, Finland

ABSTRACT

As the demand for multi-lingual speech recognizers increases, the development of systems which combine automatic language identification, language-specific pronunciation modeling and language-independent acoustic models becomes increasingly important. When the recognition grammar is dynamic and obtained directly from written text, the language associated with each grammar item has to be identified using that text. Many methods proposed in the literature require fairly large amounts of text, which may not always be available. This paper describes a text-based language identification system developed for the identification of the language of short words, e.g., proper names. Two different approaches are compared. The *n*-gram method commonly used in the literature is first reviewed and further enhanced. We also propose a simple method for language identification that is based on decision trees. The methods are first evaluated in a text-based language identification task. Both methods are also tested as preprocessors for a multilingual speech recognition task, where the language of each text item has to be determined, in order to choose the correct text-to-pronunciation mapping. The experimental results show that the proposed methods perform very well, and merit further development.

1. INTRODUCTION

Automatic language identification plays an increasingly important role in the introduction of multilingual speech recognition systems. Quite often the phonetic transcriptions of vocabulary items must be obtained on-line from written text using either rules or some other kind of pronunciation models. Most pronunciation models depend on explicit knowledge of the language, and hence, it must be identified by the system in order to enable the correct model. Language identification is often based on only written text, which creates an interesting problem. User intervention is always a possibility, but a completely automatic system would make this phase transparent and increase the usability of the system. Searching for names in a phonebook using voice input is a good example of a system requiring language identification.

Generally speaking, the language identification task can be divided into two classes: spoken and written language identification, i.e., language identification from speech or text. Obviously, spoken language identification methods have to adopt signal processing techniques, and language identification from text is a typical symbolic processing task. In our specific application framework, the language needs to be identified before the actual speech is available. So we will focus on language identification from written words, rather than from speech. The main reason for the apparent lack of activity in written language identification is probably that it is not considered a difficult problem. This might be true if the amount of written text available in the identification stage is large enough, and computational resources are not constrained. However, we are interested in performing language identification on a combination of very few words such as names and commands and within the strict computational and memory constraints present in embedded systems.

A few approaches have been presented that could be applicable in our case. *N*-gram-based methods have been applied to language identification by using either *n*-gram probabilities [1] or *n*-gram match counting [2] to estimate the probability of a given piece of text belonging to a given language. Intuitively, common words such as determiners, conjunctions and prepositions are good clues for identifying the language. A technique relying on short words is presented on the basis of this observation in [1]. Combining the short word and *n*-gram methods into a vector-space-based approach is investigated in [3], where good performance was achieved on long sentences. In the case of only few words, like with names or commands, the performance degrades significantly with the *n*-gram method, and the short words technique is not at all suitable for this task.

In this paper, we present two approaches to text-based language identification. The *n*-gram-based method is improved and a novel method utilizing decision trees is presented. The methods are first compared in a text-only language identification task and then in a complete multilingual speech recognition task.

2. *N*-GRAM-BASED APPROACH

The *n*-gram method uses letter *n*-grams, representing the frequency of occurrence of various *n*-letter combinations in a particular language. The language identification process can be divided into two phases: training and identification. A language identification model is trained for each targeted language. In the training phase, a list of words with a known language are presented as alphabetic strings. The frequency of occurrence of sequences of consecutive *n* letters is estimated from a large language specific training sample. Since it is not feasible to train all the possible partial letter sequence probabilities, a simplifying assumption is made that the probability of the current word depends only on the previous *n-1* letters, which can be implemented using *n*-grams. Typically, text from the application

area in question should be considered for training, but generic text may produce models that generalize better.
For example, the frequency P of letter l_i, given a sequence $l_{i-n+1},\ldots,l_{i-1}$ is calculated as

$$P(l_i \mid l_{i-n+1}...l_{i-1}) = \frac{count\,(l_{i-n+1}...l_i)}{count\,(l_{i-n+1}...l_{i-1})}. \qquad (1)$$

An n-gram is trained for each language and it is used in the language identification phase.

Given a word $l_1, l_2, \ldots, l_p$, the language is decided by maximizing $P(lang_i|word)$. However, this cannot be computed directly so the Bayes' rule is used instead.

$$\begin{aligned} lang &= \arg\max_{lang_i} P(lang_i \mid word) \\ &= \arg\max_{lang_i} \frac{P(word \mid lang_i) \cdot P(lang_i)}{P(word)} \\ &= \arg\max_{lang_i} P(word \mid lang_i) \\ &\approx \arg\max_{lang_i} \prod_{i=1}^{p} P(l_i \mid l_{i-n+1},...l_{i-1}, lang_i) \end{aligned} \qquad (2)$$

Hence, *lang* can be found by using the maximum-likelihood criterion because both $P(lang_i)$ and $P(word)$ can be omitted. An n-gram provides an approximation of the true probability.

In the language identification step, the likelihood value can be simply calculated by equation (2) for a given word. The language giving the highest likelihood among the available languages is chosen. The probability calculations are done in the logarithmic domain. The likelihood is computed by summing up all logarithmic n-gram probabilities. In practice, only bigrams or trigrams are commonly used due to resource limitations.

From the linguistic point of view, prefixes and suffixes are important clues for language identification. For example, words ending with "ful", "ness" or "tion" are likely to be English words. Names ending with "nen" have a very high probability for being Finnish words. Thus, separate n-grams in the beginning and end of a word can be used to enhance the performance of the language identification algorithm.

On the basis of the observation mentioned above, we propose an enhanced n-gram method. A word is decomposed into three parts: head, body and tail (head-body-tail format). The three parts are trained differently. N-grams for the head and tail are trained from the text extracted from the head and tail parts separately. The body part is trained on whole words. When computing the likelihood in equation (2), an appropriate n-gram is chosen according to the location in the input letter sequence.

3. DECISION TREE BASED APPROACH

Our second approach to language identification uses decision trees to determine the most likely language for each letter in the input word. The language is obtained by asking a series of questions about the context of the current letter, as defined by the corresponding decision tree. Decision trees are known to be very efficient in utilizing context information [4].

The most relevant context for a given input letter is found by splitting the data set in such a way that entropy is minimized. Our system uses letter context both on the left and right hand side of the current letter.

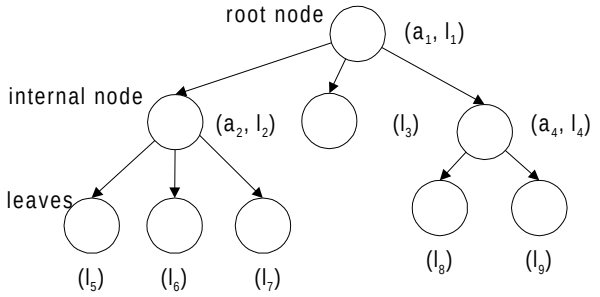

Figure 1: Exemplary decision tree showing the nodes and leaves with attributes a_i and language tags l_j.

Since only the letter context is used, and no frequency information is stored in the tree, a very compact representation is obtained. A separate tree is trained for each letter of the alphabet of the system. Each node of the tree contains an attribute (information about the most relevant context, such as "What is the second letter on the left hand side?") and a tag representing the most likely language in the current context. Figure 1 depicts a simple decision tree. The tree is composed of a root node and internal nodes, each containing an attribute and a language tag, and leaves that contain only language tags. The following sub-sections illustrate the training of decision trees and their application to language identification.

3.1 Training lexicon

In order to train a language identification system, a lexicon is required for each of the target languages. Our approach uses a large lexicon composed of all the individual lexicons tagged in such a way that the language corresponding to each word is explicit, as shown in Figure 2.

Juha	Fin,Fin,Fin,Fin
Peter	Eng,Eng,Eng,Eng,Eng
Bosch	Ger,Ger,Ger,Ger,Ger
Carlos	Spa,Spa,Spa,Spa,Spa,Spa
John	Eng,Eng,Eng,Eng

Figure 2: A combined lexicon for the training of the language identification system illustrating the use of language tags.

The choice of the lexicon is quite important for the performance of decision trees. Therefore, if the application domain is known beforehand, the content of the lexicon should be chosen accordingly.

3.2 Training of decision trees

When a decision tree is trained for a given letter, all the training samples for the letter are considered. During training, the decision tree is grown by splitting the nodes into child nodes according to an information theoretic optimization criterion [5]. The root node of the tree is split first. Splitting of the nodes continues until an ending criterion has been satisfied, or not enough training samples are left. In order to split a node into child nodes, an attribute has to be chosen. The attribute represents the context question that will be asked at the current node. All the different attributes are tested, and the one that maximizes the optimization criterion, the information gain in our

case, is chosen. In order to compute the information gain of a split, the language tag distribution before splitting has to be known. On the basis of the language tag distribution of the training samples matching the current context at the current node, the entropy E is computed according to

$$E = -\sum_{i=1}^{N} f_i \log_2 f_i , \quad (3)$$

where f_i is the relative frequency of occurrence of the i^{th} language tag, and N is the total number of language tags. The entropy after the split, E^s, is computed as the average entropy of the entropies of the subsets. Let E_j^s denote the entropy of the subset j after the split. Now the average entropy is calculated as

$$E^s = \sum_{j=1}^{K} \frac{|S_j|}{|S|} E_j^s , \quad (4)$$

where $|S|$ is the total number of training cases at the parent node, $|S_j|$ is the number of training cases in the j^{th} subset, and K is the number of subsets. The information gain for an attribute is given by

$$G = E - E^s . \quad (5)$$

The information gain is computed for every attribute, and the one that has the highest value is selected for the current node.

The splitting of the nodes continues as long as the information gain is greater than zero and the entropies of the nodes can be improved by splitting. In addition to the information gain criterion, a node can be split only if there are at least two child nodes that will have at least a preset minimum number of training cases after the split.

3.3 Identification of the language using decision trees

After the decision trees have been trained, they can be used for the identification of the language of any word. Language tags are first generated for each letter of the input word. The decision tree corresponding to the letter in question is selected. The tree is climbed starting at the root node, by answering the questions presented by the attributes, until a leaf is found, or no answer to the question is found. The language tag that corresponds to the letter can be found in the leaf or the node where the search ended. Then the process moves on to the next letter. The final decision can be made quite effectively simply by choosing the language that is the most common result for that particular word, although more elaborate schemes might also exist.

4. EXPERIMENTS

In our experiments, we focused on a multilingual speaker-independent name recognition system. The language identification models were trained and tested on our in-house name databases. Four languages were used in the experiments: English, Finnish, Spanish and German. About 17,000 distinct names were collected to form a training database. A test name database was independently collected and originally intended to test the performance of our recognition system. It was also used to evaluate the generalization capability of the language identification methods.

In addition, in order to evaluate the language identification methods in a realistic application, speech recognition experiments were also carried out by incorporating the language identification algorithms into our recognition system (described in detail in [6]). The standard Mel-Frequency Cepstral Coefficient front-end algorithm, with statics, deltas and D-deltas, was used. In order to improve the noise robustness, mean normalization was also applied to all cepstral features and variance normalization was applied only to the energy terms [7,8]. Language and speaker-independent sub-words HMMs with 8 mixtures were trained from a cross-language speech database. The Viterbi decoder was implemented according to the token passing scheme.

4.1 Text-based evaluation

Both *n*-gram and decision tree based methods were evaluated in the experiments. All the name databases were composed of short names since this is believed to be one of the most difficult tasks for language identification. The *n*-gram language identification models, including bigram, trigram and their enhanced counterparts, e_bigram and e_trigram, were trained on our in-house name databases for each of the four experimented languages. Decision trees with the context length of four were trained for each letter on the same training databases. The first results, illustrated by Table 1, were obtained on the training set to measure the learning performance of the proposed approaches. The decision tree approach outperforms the *n*-gram in terms of the learning capability. The trigram method is better than the bigram method and the *n*-gram method is clearly improved by the enhanced approach. In order to have reliable statistical information, the trigram requires a larger training set (particularly the enhanced trigram). Moreover, the trigram also consumes a significant amount of memory that might be a problem in embedded systems. Therefore the bigram was chosen to be the focus of our study.

Training Sets	Bigram	Trigram	e_bigram	e_trigram	DecTree
English	55.8	75.6	63.0	80.8	87.2
Finnish	84.2	89.8	87.5	93.9	94.9
Spanish	82.2	87.3	83.6	88.3	88.5
German	63.9	74.6	70.4	79.1	93.1
Average	71.5	81.8	76.1	85.5	90.9

Table 1: The *n*-gram language identification models and decision trees were trained and tested on the training set in order to measure their learning ability. The percentage of correctly identified languages is presented.

The generalization capability of the methods was measured by evaluating them on the independent testing set. It is widely believed that decision trees have a limited generalization capability, but a fairly good learning ability from the training samples. Table 2 shows the performance degradation of the decision tree method compared to the enhanced bigram method. In general, the *n*-gram method can work better for longer letter sequences since enough statistical information can be collected. The shorter the name is, the more important the lexical structure becomes. The decision tree is good at describing the lexical structure information of a given word.

Testing Sets	e_bigram	DecTree
English	66.3	63.0
Finnish	84.5	71.3
Spanish	71.4	54.7
German	65.0	75.4
Average	71.8	66.1

Table 2: *N*-grams and decision trees evaluated by training on the training set and testing on the test set.

4.2 Speech recognition performance evaluation

Since the language identification module was designed to work in a multilingual name recognition system, we wanted to check its effect on the performance of the complete integrated system. The speech recognition system was briefly described in the beginning of this section. A vocabulary entry is first fed into the language identification module. The text-to-phoneme mapping module corresponding to the identified language is then invoked in order to produce a phonetic transcription of the vocabulary entry. After the whole vocabulary has been processed, the recognition network is constructed using a set of multilingual monophone HMMs. The baseline system is otherwise the same, except that the language of each vocabulary entry has been predetermined. Therefore, the effect of language identification errors, as they are propagated through the text-to-phoneme mapping module, should be visible in the results.

The vocabulary of our test system is composed of names. The majority are full names (first name and surname), but also a number of first names is included to simulate a typical phonebook. A decision tree based text-to-phoneme mapping was trained for each target language as described in [9]. The baseline recognition rates for the four tested languages are shown in Figure 3. The recognition results obtained when the language identification methods were enabled are also shown. Both methods perform very similarly for the full names. In comparison with the baseline system, the average recognition rate degraded from 95.37% to 92.99% and 92.68%, for the decision tree based method and the enhanced bigram-based method, respectively.

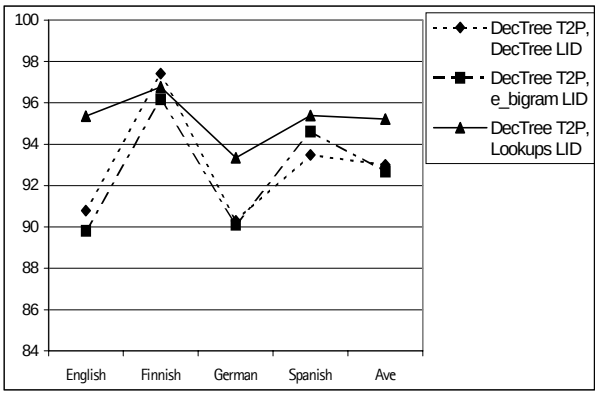

Figure 3: The recognition results tested in in-house clean speech database using the baseline system (solid line), the decision tree based language identification method (dotted line), and the enhanced bigram-based language identification method (dash-dot line).

5. CONCLUSIONS

Two novel language identification approaches for written words were proposed and further evaluated on written names. In text-based experiments, our results show that the decision tree based method outperformed the *n*-gram-based method in a closed test, but performed slightly worse in a generalization test. The decision tree method can learn lexical structure information very well, which makes it suitable for short words such as names. It can also be assumed that a majority of proper names encountered when the system is used can be included in the training set. Therefore, the importance of closed-set performance can be emphasized. The *n*-gram method performs better when long names or even a subset of a document is available to accumulate enough statistical information.

The proposed language identification methods were also evaluated in a speech recognition test in order to see the effect of language identification on the complete application. These results show that both language identification methods perform well in this task and look promising for further development. For instance, the combination of the *n*-gram and decision tree based methods could be studied.

6. REFERENCES

[1] G. Grefenstette, "Comparing Two Language Identification Schemes", *3rd International Conference on Statistical Analysis of Textual Data*, pp. 1-6, Italy, December 1995.

[2] J. Schmitt, "Trigram-based Method of Language Identification", *U.S. Patent.* Number 5062143, October 1991.

[3] J. Prager, "Linguini: Language Identification for Multilingual Documents", *32nd Hawaii International Conference on System Sciences*, pp. 1-11, Hawaii, 1999.

[4] T. Mitchell. *Machine Learning*. McGraw-Hill, USA, 1997.

[5] J. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Mateo, CA, 1993.

[6] O. Viikki, I. Kiss, J. Tian, "Speaker- and Language-Independent Speech Recognition in Mobile Communication Systems", *International Conference on Acoustics, Speech, and Signal Processing*, Salt Lake City, USA, 2001.

[7] J. Häkkinen, O. Viikki, J. Tian, and M. Vasilache, "Development of Robust Speaker Independent Command Word Recognition for Car Hands-Free Applications", *IEEE Workshop on Robust Methods for Speech Recognition in Adverse Conditions*, pp. 139-142, Tampere, Finland, May 2000.

[8] O. Viikki, D. Bye, and K. Laurila, "A Recursive Feature Vector Normalization Approach for Robust Speech Recognition in Noise", *International Conference on Acoustics, Speech, and Signal Processing*, pp. 733-736, Seattle, USA, 1998.

[9] J. Suontausta and J. Häkkinen, "Decision Tree Based Text-To-Phoneme Mapping For Speech Recognition", *6th International Conference on Spoken Language Processing 2000*, pp. 831-834, Beijing, October 2000.