Language Identification of Short Text Segments with N-gram Models

Tommi Vatanen, Jaakko J. Väyrynen, Sami Virpioja

Aalto University School of Science and Technology
Department of Information and Computer Science
PO Box 15400, FI-00076 Aalto, Finland
{tommi.vatanen, jaakko.j.vayrynen, sami.virpioja}@tkk.fi

Abstract

This paper studies a language identification task, in which the test samples have only 5–21 characters. We compare two distinct methods that are well suited for this task: a naive Bayes classifier based on character n-gram models, and the ranking method by Cavnar and Trenkle (1994). For the n-gram models, we test several standard smoothing techniques, including the current state-of-the-art, the modified Kneser-Ney interpolation. Experiments are conducted with 281 languages using the Universal Declaration of Human Rights. Advanced language model smoothing techniques improve the identification accuracy and the respective classifiers outperform the ranking method. The classifier based on n-gram models requires more memory, but language model pruning provides a way to balance the model size and the identification accuracy. We also compare the results to the language identifier in Google AJAX Language API, using a subset of 50 languages.

1. Introduction

Language identification of text has become increasingly important as large quantities of text are processed or filtered automatically for tasks such as information retrieval or machine translation. The problem has been researched long both in the text domain and in the speech domain (House and Neuburg, 1977). Existing methods have utilized various levels of information present in the text domain, for instance, short words, probabilities of various character combinations, n-grams of words, n-grams of characters, diacritics and special characters, syllable characteristics, morphology and syntax (Cole et al., 1997).

There are several very accurate methods for language identification of long strings of text (McNamee, 2005). However, these methods rarely work well with very short texts, and are typically evaluated only on a small set of languages, that may not reveal all the problems in applications with hard contexts (da Silva and Lopes, 2006).

We study character-based language identification with n-gram language models. Because of the compact models that do not need word-based features, this approach is well suited for language identification tasks that have dozens of languages, little training data and short test samples. Character-based n-gram models have been widely used in language identification; see, e.g., Beesley (1988), Cavnar and Trenkle (1994), Dunning (1994) and Teahan (2000). However, our work extends the previous contributions in at least three ways. First, the emphasis has been mostly with the identification of texts that contain at least several words. In contrast, our test samples consist of 5–21 characters. Furthermore, we do not take word boundaries into special consideration. That is, a test sample may only be a part of a word. Second, the set of languages has always been more limited; usually less than thirty languages. Our experiments are carried out with 281 languages using the Universal Declaration of Human Rights as a corpus. In addition to making the task more challenging in general, an extensive set of languages prevents using any language-specific information, such as existence of word boundaries in the text.

Third, the previous studies on written language identification do not include modern n-gram modeling techniques such as n-gram smoothing and model pruning, which have been important for improving, e.g., speech recognition systems. The smoothing methods applied in language identification have usually been either very simple (Dunning, 1994) or non-standard (Valencia and Yvon, 1997), and there has been no extensive study of their effect on the task. We test several smoothing methods, including state-of-the-art methods such as modified Kneser-Ney interpolation by Chen and Goodman (1999). In addition, we consider the effects of selecting the n-gram length and removing non-relevant n-grams with the pruning algorithm by Stolcke (1998), which both relate to the trade-off between the size of the n-gram model and its prediction accuracy.

1.1. Related Work

Character n-grams have been applied to language identification together with, for instance, language modeling (Beesley, 1988; Dunning, 1994), frequency profile matching (Cavnar and Trenkle, 1994) and compression (Teahan, 2000). For short test samples, they have previously been considered in foreign name identification (Valencia and Yvon, 1997), but without rigorous experiments and using ad-hoc techniques. In addition, there are at least two recent papers that consider identification of short test samples with word-based models.

Hammarström (2007) models word emission probabilities with relative frequencies of words. An unsupervised affix detection component models unseen words and language switches are minimized in a sequence of words. The method is tested with a set of 32 languages but is not compared to any other method.

Řehůřek and Kolkus (2009) model words in a document using a Bernoulli distribution and taking into account the discrimination power of the words in feature selection. The method is tested with nine languages against a modified version of the n-gram method by Teahan (2000). For short

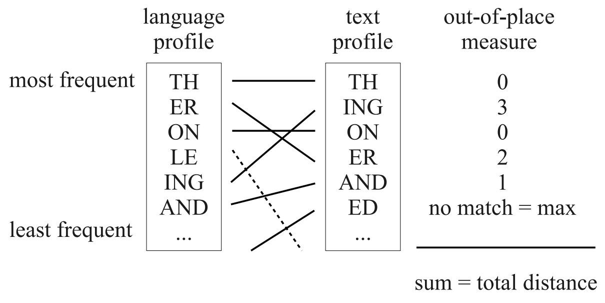

Figure 1: An illustration of the rank order method. Adopted from Cavnar and Trenkle (1994).

text segments (2–5 words), they report higher recall but, on average, lower precision for their method.

2. Methods

The language identification task is often divided into two separate steps: language modeling and classification. Language models are typically constructed independently of each other, without taking into account the final classification task in the process. A language model might then contain features that have no or very little discriminative power in the classification. However, there are also approaches that build the language models and the classification system at the same time (Řehůřek and Kolkus, 2009).

The classifier used in the language identification plays an important role. There are several general machine learning methods that have been applied in the language identification task, for instance, support vector machines (Kruengkrai et al., 2005), normalized dot product (Damashek, 1995), k-nearest neighbor and relative entropy (Sibun and Reynar, 1996). Other applied methods include decision trees, neural networks and multiple linear regression (Botha and Barnard, 2007).

In this section, the ranking method and the classifiers based on n-gram models applied in our experiments are described.

2.1. Ranking Method for Language Identification

The ranking method by Cavnar and Trenkle (1994) is widely used for language identification and text categorization. The method generates language specific profiles which contain the m most frequent character n-grams of the training corpus sorted by their frequency. Figure 1 illustrates the ranking method. A similar text profile is created from the classified text. A cumulative "out-of-place" measure between the text profile and each language profile is calculated. It determines how far out of place any n-gram in one profile is from its place in the other profile. If an n-gram in the text profile is missing from the language profile, it yields the maximum distance, which equals the number of n-grams in the language profile.

For short text inputs, the text profile typically contains much less n-grams than the language profile. Therefore, our experiments are carried out with a revised ranking method that does not penalize for language profile n-grams that are not present in the text profile.

In their original paper, Cavnar and Trenkle (1994) identify long text samples with $m \in \{100, 200, 300, 400\}$ and obtain the best results with $m = 400$. As our identification task is different, consisting of short segments, we will optimize the parameter values for it.

2.2. Language Identification with N-gram Models

An n-gram model defines a probability distribution over utterances of a language, making the $(n-1)$:th order Markov assumption. That is, the probability of an observation (usually a word or a character) is assumed to depend only on the previous $n-1$ observations. Maximum likelihood estimates of the probabilities tend to overlearn the training data. The estimates can be improved by various smoothing techniques which move probability mass from the n-grams that occur rarely to those that do not occur at all. The simplest but a crude way is to distribute the probability uniformly over unseen events. A thorough overview of common smoothing methods is given by Chen and Goodman (1999).

For the language identification task, we use character-based n-gram models. A classifier based on the n-gram models is created as follows: One model is trained for each language. Given a test sample, its likelihood is calculated for all the models, and the language that gives the best likelihood is selected. Taking individual n-grams as feature variables, this can be considered a naive Bayes classifier.

Next, we shortly describe the methods for building n-gram models used in the experiments.

2.2.1. Additive Smoothing

One of the first to experiment with character n-gram models in language identification was Dunning (1994). He used character-based language models smoothed with the simple "add one" smoothing (originating from Laplace's rule of succession), which adds one to the counts of all possible n-grams. However, this leads to overestimation of the probabilities of unseen n-grams.

A simple generalization is the general additive (or Lidstone) smoothing, where a smaller value $\lambda < 1$ is added to the counts:

$$P_{\text{add}}(x_i|x_{i-n+1}^{i-1}) = \frac{C(x_{i-n+1}^{i}) + \lambda}{C(x_{i-n+1}^{i-1}) + \lambda V}, \tag{1}$$

where x_i^j denotes the sequence $x_i \dots x_j$, V is the size of the vocabulary (number of different characters in the language) and $C(x)$ denotes the number of occurrences of an item x. The parameter λ can be optimized by maximizing the probability of the held out data.

In this article, additive smoothing with $\lambda = 1$ is called Laplace smoothing and additive smoothing with optimized λ is called Lidstone smoothing.

2.2.2. Katz Smoothing

One of the methods used widely in speech recognition domain is Katz smoothing (Katz, 1987), in which the n-gram counts are modified using the Good-Turing estimate (Good, 1953). In addition, lower-order n-grams are used whenever the higher-order n-grams are not available. This type of models are called back-off models, in contrast to interpolated models, where the probabilities of the lower order n-grams are always interpolated with the higher-order n-grams (Chen and Goodman, 1999).

2.2.3. Absolute Discounting

Absolute discounting (Ney et al., 1994) is a simple method that usually gives results close to the Katz smoothing (Chen and Goodman, 1999). An optimized discount parameter D is removed from every count, and the left-out probability mass is used as interpolation weight with the lower-order model:

$$P_{\text{abs}}(x_i \,|\, x_{i-n+1}^{i-1}) = \frac{C(x_{i-n+1}^{i}) - D}{C(x_{i-n+1}^{i-1})} + \lambda_{x_{i-n+1}^{i-1}} P(x_i \,|\, x_{i-n+2}^{i-1}), \qquad (2)$$

where $\lambda_{x_{i-n+1}^{i-1}}$ is a scaling factor that makes the conditional distribution sum to one.

2.2.4. Kneser-Ney Smoothing

Kneser and Ney (1995) proposed a method where the probability of a lower-order n-gram is set to be proportional to the number of different units x_{i-n+1} that it follows

$$N(\bullet x_{i-n+2}^{i}) = |\{x_{i-n+1} : C(x_{i-n+1}^{i}) > 0\}|, \qquad (3)$$

instead of being proportional to the number of occurrences of the lower order $(n-1)$-gram $C(x_{i-n+2}^{i})$. I.e., if the number of different contexts is high, we assume that it will probably occur also with context types that we have not seen before. Note that $C(x)$ denotes the number of occurrences of an item x, and $N(\bullet x)$ denotes the number of context types in which x occurs. Similarly,

$$N(\bullet x_{i-n+2}^{i-1} \bullet) = \sum_{x_j} N(\bullet x_{i-n+2}^{i-1} x_j). \qquad (4)$$

Using this notation, the probabilities are estimated as

$$P_{\text{KN}}(x_i \,|\, x_{i-n+2}^{i-1}) = \frac{N(\bullet x_{i-n+2}^{i})}{N(\bullet x_{i-n+2}^{i-1} \bullet)}. \qquad (5)$$

The actual smoothing used by Kneser and Ney (1995) was absolute discounting with back-off to lower-order models. The modified Kneser-Ney interpolation by Chen and Goodman (1999) differs from the original method in two ways: Interpolation is applied instead of back-off, and the discount parameter D is optimized separately in the cases where the number of occurrences to discount is one, two, and three or more. The modified Kneser-Ney interpolation has been shown to outperform other smoothing methods in cross-entropy and speech recognition tests, especially when high-order n-grams are used (Goodman, 2001).
Later in the article, "Kneser-Ney" and "KN" will refer to an interpolated model with Kneser-Ney smoothing, and "modified Kneser-Ney" to the modified Kneser-Ney interpolation that applies three discount parameters.

2.2.5. Model Pruning

The size of an n-gram model grows rapidly with increasing n and training corpus size. The simplest way to reduce the model size is to exclude n-grams that occur fewer times than a given cut-off count. The n-grams that contribute only little to the modeling accuracy can also be excluded using pruning algorithms such as entropy-based pruning (Stolcke, 1998) and revised Kneser pruning (Siivola et al., 2007), which works better for Kneser-Ney smoothed models. Some results indicate that pruning combined with cut-offs can produce better results than pruning alone (Goodman and Gao, 2000). In addition to removing n-grams from a full model, it may be useful to select the initial n-grams with growing algorithms that make it possible to use very long n-grams (Siivola et al., 2007).

Language	**Document size**		**Character set size**
	(kilobytes)	(characters)	(characters)
English	10.494	10 730	57
French	12.082	11 990	60
German	11.964	12 065	70
Spanish	11.998	12 060	61
Portuguese	11.578	11406	64
Finnish	12.559	12 328	57
Greek	22.260	12 529	68
Russian	21.341	11 902	67
Tagalog	13.192	13 488	57
Ashéninca	**26.399**	**26 844**	60
Cashinahua	**4.945**	5 016	59
Japanese	12.177	**4 368**	506
Chinese	10.854	5 457	**539**
Kikongo	11.669	11 931	**44**
⋮	⋮	⋮	⋮
Average	11.943	11 334	66.0

Table 1: Language specific document sizes, measured in kilobytes and characters, and the character set sizes of the selected languages. The extreme values within the corpus are in bold and the averages are calculated over all 281 languages.

3. Experiments

The goal of our experiments is to find out how the different character-based methods for language identification perform with short text segments and a set of 281 languages. Especially, we study how common language modeling techniques from the speech recognition research could be adopted to this task. In this section, we describe our data set, evaluation procedure, and the applied implementations of the identification methods.

3.1. Data

For the experiments, we used documents from the Universal Declaration of Human Rights[1] in 281 languages. The corpus consists of one pdf document of few pages in each language. Text was extracted in UTF-8 encoding with the `pdftotext` tool for Linux. Some documents contained headers or comments in English which were manually removed. White space sequences were replaced by a single space. The preprocessed data is available at `http://www.cis.hut.fi/research/cog/data/udhr/`.

[1]`http://www.un.org/en/documents/udhr/`

Table 1 shows file sizes, character counts and character set sizes on the selected languages. The extreme values within the corpus are emphasized with bold font. The median document length was 11 095 characters, with range of 4 368–26 844 characters. The first and third quartiles were 10 038 and 12 448, resulting to the relatively small interquartile range of 2 411 characters.

The language specific character set sizes had range of 44–539 characters. The median was 61 characters and the interquartile range 57–66 characters. Five languages had character vocabularies larger than 100 characters: Arabic, Chinese, Japanese, Korean and Ukrainian.

A smaller data set of 50 languages was selected from the full language set for comparison with with the Google AJAX language API[2]. The language set consisted of all languages from the full language set that the Google API could identify. The median length of a document in this data set was 11 304 characters with interquartile range of 10 375–12 065 characters.

We made as few prior assumptions as possible on our text data. For instance, we did not assume that word boundaries are marked. Rather, all characters, including spaces, were treated equally. This is reasonable, because our corpus has of a very diverse set of languages. For instance, the Japanese text has only 213 spaces, whereas the texts in Maori and Nigerian Pidgin English documents have more than 3 000 spaces. Furthermore, on Chinese there is a space between every character. Counting space as a regular character in the models provided additional information of its average frequency for the language model.

3.2. Evaluation

As an evaluation measure, we apply accuracy, which is the proportion of correctly identified test samples. For individual languages, we can calculate precision and recall. Precision is the proportion of correctly classified test samples in all samples classified to the given language, whereas recall is the proportion of correctly classified test samples in all samples of the given language. As we have the same amount of test samples for each language, the accuracy of a method can be calculated by taking the average over the recall of each language.

Because of the small size of the corpus, we apply 10-fold cross validation, in which the text in each language is divided into ten equal-sized parts. Eight parts are used as a training set for calculating the n-gram frequencies for the models, one part is used as a held-out set for optimizing the parameters of the models, and one part is reserved for testing. From each test set, we randomly select 50 samples per each sample length. We do not restrict the test samples to whole words—they are chosen randomly regardless of the word boundaries. To estimate the statistical significance between the results of any two methods, we apply 10-fold cross validated paired t-test with 95 % confidence level.

3.3. Tools and Implementation

N-gram models with absolute discounting, Kneser-Ney smoothing and modified Kneser-Ney smoothing were build

[2]`http://code.google.com/apis/ajaxlanguage/`

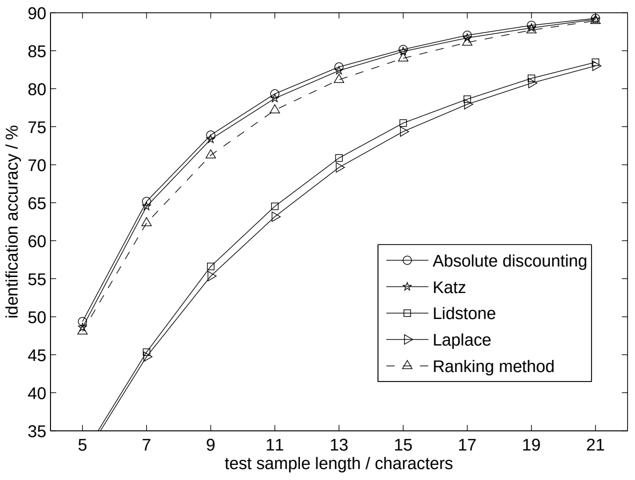

Figure 2: The overall identification accuracy for selected methods as a function of test sample length. Parameters for the methods are $n = 5$ (absolute discounting and Katz smoothing), $n = 3$ (Laplace and Lidstone smoothing) and $n = 6, m = 7\,000$ (ranking method).

and pruned with the VariKN toolkit (Siivola et al., 2007). Models with Katz smoothing were build with the SRI Language Modeling Toolkit (Stolcke, 2002). N-gram models with additive smoothing (Laplace and Lidstone) were build with our own Python implementation. The n-gram model based classifier and the ranking method were implemented in Python. Google AJAX Language API results were retrieved automatically on March 21–22, 2010.

4. Results

The overall identification accuracy of selected methods as a function of the test sample length can been seen in Figure 2. The number of n-grams for the ranking method and the maximum n-gram length for all methods have been optimized, as described later. Results with Kneser-Ney smoothing methods are excluded, as they overlap with the results of the absolute discounting method. As expected, the accuracy improves as the test sample length increases. The differences between the methods are most obvious with shorter test samples. As we continue to analyze the results, we will show the averages over both *all* test sample lengths $(5, 7, \ldots, 21)$, and *short* test sample lengths $(5, 7, 9)$, in order to emphasize the challenges with very short test samples. The final results and the optimal parameters for all the evaluated methods are presented in Table 2.

4.1. Ranking Method

We optimized the parameters n and m of the ranking method for our task by experimenting with the parameter combinations with n ranging from one to eight and m in the range 200–7 000. Figure 3 shows the average accuracy over all sample sizes with the tested n as a function of m. The curves for $n \in \{1, 7, 8\}$ are not shown. The unigram ranking model $(n = 1)$ gave 28.8 % accuracy with no improvement as m increases, because most languages have a character vocabulary of size $V < 200$. The curves for $n \in \{7, 8\}$ would be below the curve for $n = 6$. The same

Method	Parameter(s)	Accuracy / % short	Accuracy / % all
Absolute discounting	$n = 5$	62.8	77.8
Katz	$n = 5$	62.1	77.4
KN	$n = 4$	60.2	76.9
Modified KN	$n = 4$	59.8	76.6
Ranking	$n = 6$, $m = 7\,000$	60.6	76.3
Lidstone	$n = 3$	53.7	71.0
Laplace	$n = 3$	52.0	70.6

Table 2: The optimal parameters and results for short and all test sample lengths for the evaluated methods.

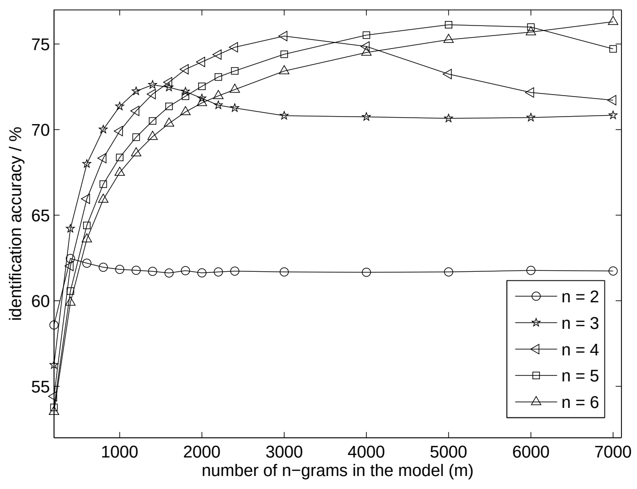

Figure 3: Average accuracy over all sample lengths for the ranking method with different n as a function of m.

comparison with short sample lengths produced similar behavior but with accuracies roughly 20 % lower.

Figure 3 shows that the accuracy of the ranking method is increasing as m grows until some n-specific extreme point. After this point adding more n-grams to the models starts to impair the accuracy. The larger the n, the further we can grow the model to improve the results. On the other hand choosing too large n makes the results worse.

4.2. N-gram Language Models

Figure 4 shows the accuracy of different classification methods as a function of n, the maximum length of n-grams in the model. We performed the experiments up to 8-grams, but the results for $n > 6$ are not shown, since increasing n over six only reduced accuracy. The methods applying language models with additive smoothing (Laplace, Lidstone) perform relatively well with $n < 4$, but with longer n-grams the accuracy drops dramatically. The methods using advanced smoothing techniques (absolute discounting, Katz and both KN smoothing methods) have the higher accuracy the longer n-grams are used, up to 4 or 5-grams. The ranking method with optimal profile sizes performs slightly worse than the best n-gram models, but gets closer as n grows.

Figure 5 highlights the differences between the best

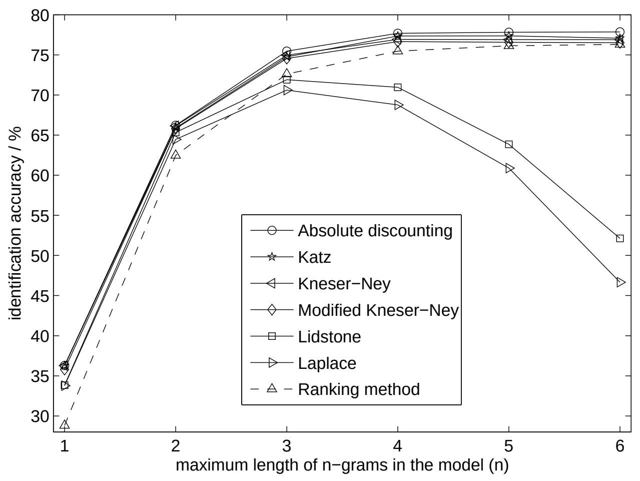

Figure 4: The average accuracy over all test samples for the methods as a function of n. For the ranking method, m is optimized for every n as described in Section 4.1.

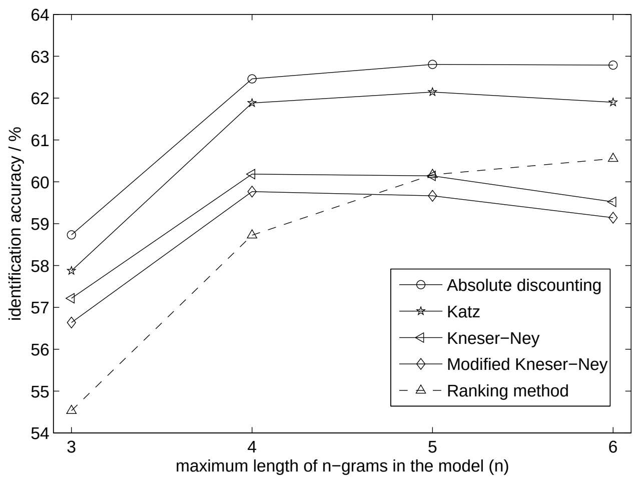

Figure 5: The average accuracy over short test samples of different classifiers as a function of n. For the ranking method, best possible m is chosen for every n as described in Section 4.1.

smoothing methods and the ranking method by showing the accuracies with $n \in \{3, 4, 5, 6\}$. The accuracies were calculated for short test samples to emphasize the differences. All differences between the methods were statistically significant, except the difference between the ranking method and the Kneser-Ney method at $n = 5$. Interestingly, the state-of-the-art smoothing technique, modified Kneser-Ney smoothing, obtains worse accuracies than absolute discounting or Katz smoothing. Furthermore, the best parameter for KN and modified KN is $n = 4$, whereas absolute discounting and Katz obtained the best results with $n = 5$.

4.3. Memory and CPU Time Usage

Figure 6 shows the model size of different methods as a function of n, the maximum length of n-grams in the model. The language model sizes are calculated by taking the sum over the sizes of files saved in plain text ARPA format. For the ranking method, n-grams were stored into a plain text file and the file sizes were summed up. The increase in

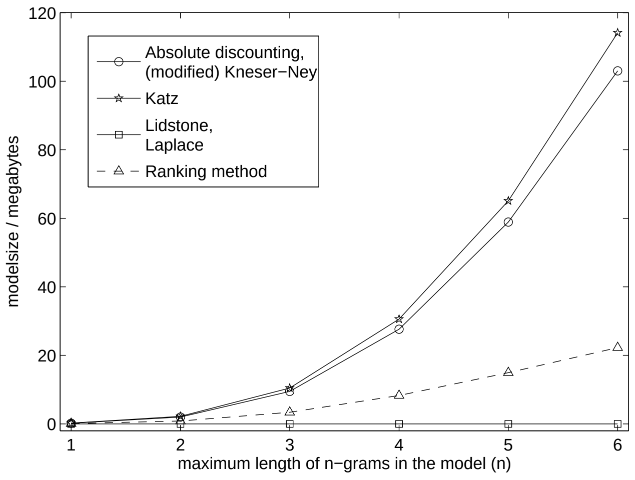

Figure 6: The model sizes of the methods used in Figures 4 and 5 as a function of maximum length of n-grams n.

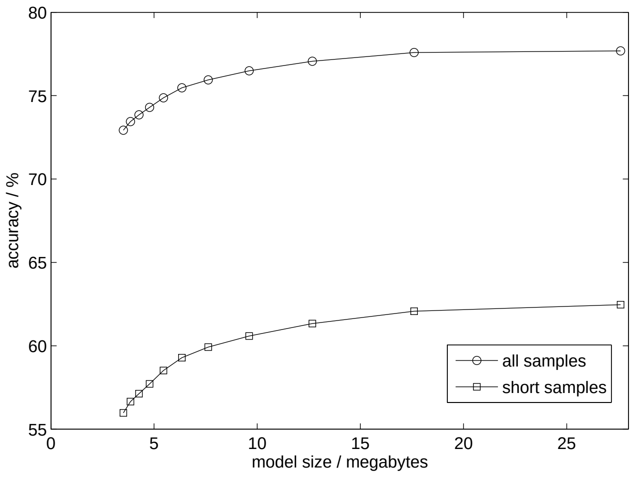

Figure 7: The effect of model pruning for identification accuracy of absolute discounting method ($n = 4$) with both short and all sample lengths.

model size is steepest with the smoothing methods that use backoff or interpolation. For the Katz smoothing, the models are slightly larger because the SRILM toolkit stores the model parameters with a higher precision.

We investigated the effect of language model pruning on the classifier size and the identification accuracy. The experiments were run for the best performing smoothing method, absolute discounting, with $n = 4$ and pruning parameter $\varepsilon \in \{0, 0.02, \ldots, 0.20\}$. Figure 7 shows results for both all test sample length and short test sample lengths. For all test sample lengths, the model size can be almost halved without a statistically significant decrease in the accuracy. For short test samples, the decrease is slightly larger and statistically significant even with the smallest tested amount of pruning.

In addition to the larger model sizes, the classification based on n-gram models was also slower than classification with the ranking method. With the optimal models, the difference was roughly tenfold. With our implementation, pruning of the n-gram models did not reduce the classification times. Also the training of the language profiles of the ranking method is considerably faster and simpler than training of the n-gram models.

4.4. Comparison to the Google API

Experiments with the Google AJAX language API were run with the smaller language set. The overall identification accuracy for the Google API was weak, ranging from 20.8 % to 59.9 % with different test sample lengths, compared to the accuracy that was obtained with the best n-gram models, ranging from 64.1 % to 90.7 %.

Table 3 shows the comparison between absolute discounting and the Google API with selected languages. As the average recall rates show, the Google API is probably not intended for language identification of short text segments. Even though the Google API had 100 % precision rates on some languages, such as Serbian and Chinese, it only indicates that no test samples of other languages were incorrectly classified to this language. As also the recall rates for these particular languages were very low, the Google API cannot reliably be used for identifying short texts in these languages. For English, the Google API had better recall (86.4 %) than obtained with the n-gram models (76.4 %). This is explained by the fact that the Google API probably had a very high emphasis on identifying English: 17.8 % of all and 29.2 % of short non-English test samples were identified as English. This is naturally at the cost of identifying the other languages.

There were some anomalies in the Google API results which affected the overall accuracy. It did not recognize Tagalog[3] or Portuguese at all, even though the list of supported languages included them. It was also unable to detect Greek with test sample lengths such as 19 and 21 characters whereas test samples of 15 characters were identified flawlessly.

5. Discussion

We have refrained from making any prior assumption on our corpus in the experiments. In real-word applications with specific language sets, language specific information can be exploited, e.g., in preprocessing. For instance, if one works with languages clearly consisting of words, combining word-based language models with character-based models could provide improved accuracy.

The widely used ranking method has two parameters (maximum n-gram length n and the number of n-grams m) that affect the model. Our experiments suggest that when the test samples are very short, the identification accuracy can be significantly improved by increasing the m far beyond the values suggested by Cavnar and Trenkle (1994). This can be explained by the fewer number of n-grams in the short input: unlike with long inputs, a short text input is not very likely to contain the most frequent n-grams in the language. Therefore, the model should contain also more infrequent n-grams.

Rather surprisingly, modified KN has the lowest identification accuracy of the advanced smoothing methods. However, by calculating average perplexities for the test data of the same language the model was trained, we found that

[3]Tagalog is spoken in the Philippines by 22 million people.

	Absolute discounting				Google API			
	Precision / %		Recall / %		Precision / %		Recall / %	
Language	short samples	all samples	short samples	all samples	short samples	all samples	short samples	all samples
English	59.6	74.9	62.5	76.4	4.8	9.0	71.3	86.4
French	62.4	77.8	60.6	77.6	21.4	34.8	50.7	75.6
German	78.0	89.3	75.1	87.4	42.2	61.0	41.3	68.6
Spanish	41.4	56.9	40.7	57.7	22.6	33.2	33.2	65.1
Portuguese	37.2	50.9	43.9	60.0	0.0	0.0	0.0	0.0
Maltese	82.5	91.3	81.0	91.3	73.5	87.4	26.8	37.7
Finnish	80.2	91.7	83.1	92.5	42.8	61.8	28.8	56.9
Greek	99.9	100.0	99.7	99.9	99.0	99.3	99.2	69.4
Slovenian	65.0	79.3	64.5	77.8	50.2	73.7	17.0	34.3
Serbian	61.5	76.2	62.7	76.2	100.0	100.0	10.7	20.6
Chinese	99.2	99.7	96.7	98.1	100.0	100.0	7.8	44.8
Russian	62.6	74.3	67.3	79.1	31.0	38.4	68.0	81.0
Uzbek	82.1	92.5	81.1	91.2	100.0	100.0	6.5	8.9
Malay	54.0	61.1	46.0	52.1	42.5	56.9	13.7	23.6
Tagalog	85.4	94.1	83.5	91.8	0.0	0.0	0.0	0.0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
Average	72.5	82.4	72.3	82.2	53.1	63.0	28.0	43.8

Table 3: Recall and precision for absolute discounting ($n = 5$) and Google AJAX Language API for selected languages with both short and all test sample lengths. Precision is the proportion of correctly classified test samples in all samples classified to the given language. Recall is the proportion of correctly classified test samples in all samples of the given language. Average values are calculated over all 50 languages in the smaller language set.

the modified KN improved the predictions over the other smoothing methods. This suggests that the performance of the language model as such does not necessarily reflect the classification accuracy in language identification with the naive Bayes classifier.

Combining different types of language models is one method of improving the prediction ability (Goodman, 2001). We made two initial experiments with linear combination. Backward n-gram models (Duchateau et al., 2002), where the probabilities of the characters are estimated conditioned on the following characters, improved the identification accuracy 0.9 % absolute with 5-gram models smoothed with absolute discounting. Another simple way of improving the results might be interpolating with a background model trained with the full (multilingual) data. As explained by Zhai and Lafferty (2001), the background model should reduce the influence of common characters, similar to the inverse document frequency weighting applied in vector space models. More extensive experiments with model combination is left to future work.

Considering the sizes of the n-gram models, there are some methods of creating more compact models that are not reported above. We did make initial experiments with variable length n-gram models trained with the growing algorithm by Siivola et al. (2007), but the models did not improve the results of 5-gram models (as did not full models using longer n-grams). However, we did not try using count cut-offs or clustering, which are reported to be more efficient than pruning alone in word-based n-gram models (Goodman and Gao, 2000). Overall, these methods should be more useful with larger training corpora.

6. Conclusions

Although many language identification methods work very well for documents or other long texts, our experiments confirmed that the identification of short text segments is not a solved problem. The accuracy of the studied methods was found to decrease significantly when the identified text gets shorter. Therefore, finding the best methods and optimizing the parameters is important with short text inputs.

We investigated n-gram language models with different smoothing methods, including the Laplace smoothing used in the language identification model by Dunning (1994) and advanced smoothing methods such as absolute discounting, Katz smoothing and Kneser-Ney smoothing. The best parameters and the respective results for all the tested methods are shown in Table 2.

The additive smoothing methods (i.e., Laplace used in Dunning (1994) and Lidstone) are clearly outperformed by the ranking method and all the advanced smoothing methods. With additive smoothing the accuracy has a peak at maximum n-gram length $n = 3$ and drops dramatically with longer n-grams. In contrast, all the advanced smoothing methods (absolute discounting, Katz, KN and modified KN) behave more systematically as n grows. We obtained highest accuracies with absolute discounting and Katz smoothing ($n = 5$ for both), with less than one percent difference. The next best results are obtained with the ranking method ($n = 6$, $m = 7\,000$) and the n-gram models with KN smoothing ($n = 4$). Their accuracies are worse than the best ones, especially with the short sample lengths. We studied reducing the model size with language model pruning. The more accurate models have larger model sizes which, however, can be almost halved without a significant decrease in identification accuracy. Pruning is clearly a good option for producing more compact models, but there is a natural trade-off between model size and accuracy.

We compared our best method, absolute discounting, with the language identification provided in the Google AJAX Language API. In overall, absolute discounting gave superior results, but the Google API was slightly better identifying some individual languages. The comparison might be unfair as it seems that the Google API is developed for longer text inputs.

To conclude, the advanced smoothing methods (e.g., absolute discounting) are more accurate than the ranking method and additive smoothing methods in identifying small text samples. However, the higher accuracy is obtained at the cost of larger models and slower classification speed.

7. References

Kenneth Beesley. 1988. Language identifier: A computer program for automatic natural-language identification of on-line text. In *Proceedings of the 29th ATA Annual Conference*, pages 47–54.

Gerrit Reinier Botha and Etienne Barnard. 2007. Factors that affect the accuracy of text-based language identification. In *Proceedings of PRASA 2007*, pages 7–10.

William B. Cavnar and John M. Trenkle. 1994. N-gram-based text categorization. In *Proceedings of SDAIR'94*, pages 161–175.

Stanley F. Chen and Joshua Goodman. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–393.

Ron Cole, Joseph Mariani, Hans Uszkoreit, Giovanni Varile, Annie Zaenen, Victor Zue, and Antonio Zampolli, editors. 1997. *Survey of the State of the Art in Human Language Technology*. Cambridge University Press. Web Edition in PDF.

Joaquim Ferreira da Silva and Gabriel Pereira Lopes. 2006. Identification of document language is not yet a completely solved problem. In *Proceedings of CIMCA'06*, pages 212–219.

Marc Damashek. 1995. Gauging similarity with n-grams: Language-independent categorization of text. *Science*, 267(5199):843–849.

Jacques Duchateau, Kris Demuynck, and Patrick Wambacq. 2002. Confidence scoring based on backward language models. In *Proceedings of ICASSP 2002*, volume 1, pages 221–224.

Ted Dunning. 1994. Statistical identification of language. Technical Report MCCS-94-273, Computing Research Lab, New Mexico State University.

I. J. Good. 1953. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40:237–264.

Joshua Goodman and Jianfeng Gao. 2000. Language model size reduction by pruning and clustering. In *Proceedings of ICSLP*, pages 16–20.

Joshua Goodman. 2001. A bit of progress in language modeling, extended version. Technical Report MSR-TR-2001-72, Microsoft Research.

Harald Hammarström. 2007. A fine-grained model for language identification. In *Proceedings of iNEWS-07 Workshop at SIGIR 2007*, pages 14–20.

Arthur S. House and Edward P. Neuburg. 1977. Toward automatic identification of the language of an utterance. I. Preliminary methodological considerations. *Journal of the Acoustical Society of America*, 62(3):708–713.

Slava M. Katz. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35:400–401.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of ICASSP-95*, pages 181–184.

Canasai Kruengkrai, Prapass Srichaivattana, Virach Sornlertlamvanich, and Hitoshi Isahara. 2005. Language identification based on string kernels. In *Proceedings of ISCIT-2005*, pages 896–899.

Paul McNamee. 2005. Language identification: a solved problem suitable for undergraduate instruction. *Journal of Computing Sciences in Colleges*, 20(3):94–101.

Hermann Ney, Ute Essen, and Reinhard Kneser. 1994. On structuring probabilistic dependence in stochastic language modelling. *Computer Speech and Language*, 8(1):1–38.

Radim Řehůřek and Milan Kolkus. 2009. Language identification on the web: Extending the dictionary method. In *Proceedings of CICLing 2009*, pages 357–368.

Penelope Sibun and Jeffrey C. Reynar. 1996. Language identification: Examining the issues. In *Proceedings of SDAIR'96*, pages 125–135.

Vesa Siivola, Teemu Hirsimäki, and Sami Virpioja. 2007. On growing and pruning Kneser-Ney smoothed n-gram models. *IEEE Transactions on Audio, Speech & Language Processing*, 15(5):1617–1624.

Andreas Stolcke. 1998. Entropy-based pruning of backoff language models. In *Proceedings of DARPA Broadcast News Transcription and Understanding Workshop*, pages 270–274.

Andreas Stolcke. 2002. SRILM – an extensible language modeling toolkit. In *Proceedings of ICSLP*, pages 901–904. `http://www.speech.sri.com/projects/srilm/`.

William John Teahan. 2000. Text classification and segmentation using minimum cross-entropy. In *Proceedings of RIAO'00*, pages 943–961.

Érika Valencia and François Yvon. 1997. Identification de l'origine linguistique: le cas des noms propres. In *Actes des Journes Scientifiques et Techniques du Réseau Francil (JST97)*, pages 473–480.

Chengxiang Zhai and John Lafferty. 2001. A study of smoothing methods for language models applied to Ad Hoc information retrieval. In *Proceedings of SIGIR'01*, pages 334–342.