N-gram models for language detection

Carlos Ramisch
M2R Informatique - Double diplôme ENSIMAG – UJF/UFRIMA
UE Ingéniérie des Langues et de la Parole
ramischc@ensimag.imag.fr

December 21, 2008

Abstract

In this document we report a set of experiments using *n-gram language models* for automatic *language detection* of text. We will start with a brief explanation of the concepts and of the mathematics behind n-gram language models and discuss some applications and domains in which they are widely used. We will also present an overview of related work in language detection. Then, we will describe the resources used in the experiments, namely a subset of the *Europarl* corpus and the `SRILM` toolkit. We will then perform a toy experiment in order to explain in detail our methodology. Afterwards, we will evaluate the performance of different language models and parameters through a precision measure based on the perplexity of a text according to a model. We conclude that n-gram models are indeed a simple and efficient tool for automatic language detection.

Contents

1 Introduction

In this report, we are interested in creating a system for *language identification*. We define it as special type of text categorisation where, given a sample of text, the system needs to define in which natural language this text was written. For instance, If the following sentences are input into the system, the expected output would be German, English and French:

Das Protokoll der gestrigen Sitzung wurde verteilt.
The Minutes of yesterday's sitting have been distributed.
Le procès-verbal d'hier a été distribué.

The intuition behind the solution of this problem is that we are able to build a model for each language that we would like to identify. Then, we can compare an input text with each one of the models and only keep the language for which the distance between the text and the model is minimal. For example, if a given sentence is very close to the model of French and far enough of the models for other languages, the system will say that this sentence was written in French. In this work, we use *n-gram language models* and our distance measure is *perplexity*. Formally, we can state our underlying hypothesis as follows:

Main Hypothesis Given a set of languages LS and a text T, we can detect the language of T by taking the language $ls_i \in LS$ for which the perplexity of T according to an n-gram model of ls_i is minimal.

An *n-gram* is any sequence of (contiguous) words in a text. N-gram language models are a class of models that assign a probability for each n-gram of a language, for a fixed value of n, based on the Markov assumption that the occurrence of a word only depends on a short history (the $n-1$ previous words). This probability may be estimated from a corpus (the *training corpus*), using the Maximum Likelihood Estimator (MLE), e.g. in section 4.1, the trigram *monsieur le président* appears 7,738 times in a corpus containing $N = 428,106$ trigrams, and therefore it has a probability of

$$P_{MLE}(w_i \dots w_n) = \frac{C(\text{monsieur le président})}{N} = \frac{7738}{428106} = 1.8\%$$

However, an n-gram that does not appear in the training corpus will have a probability of 0, under the assumption that the training corpus contains all possible n-grams for a given language. This assumption is not true, due to the sparseness of language and to the limited availability of corpora. Hence, we would like to assign a low (but non-zero) probability for unseen n-grams. Chen and Goodman [1996] present a set of statistical *smoothing* techniques for n-gram language models that can solve this problem. The technique used in this report is Good-Turing discounting with back-off smoothing [Katz, 1987], according to the inductive definition:

$$P_{BO}(wi \dots w_n) = \begin{cases} (1 - d_{w_1 \dots w_{n-1}}) P_{MLE}(w_i \dots w_n), \text{ for seen n-grams} \\ \alpha_{w_1 \dots w_{n-1}} P_{BO}(w_2 \dots w_n), \text{ for unseen n-grams} \end{cases}$$

The factor d is calculated using Good-Turing discounting and the normalisation factor α makes the probability mass left over in the discounting process to be distributed among unseen n-grams. This is explained in detail by Katz [1987] and by Manning and Schütze [1999].

The *perplexity* of a sequence of words w_1 through w_L (the *test set*) according to an n-gram model BO is defined as $2^{H(w_1 \dots w_L, BO)}$, where H is the *cross-entropy* between the text and the model, according to the information theory formula:

$$H(w_1 \dots w_L, BO) = \frac{1}{L-n+1} \sum_{l=0}^{L-n} log_2 P_{BO}(w_1 \dots w_{l+n}),$$

where the test data contains $L-n+1$ n-grams and the probabilities are estimated according to the back-off probabilities described above.

The initial motivation for n-gram models comes from Speech Processing: n-gram language models are especially useful for speech recognition. Nowadays, n-gram models are used in a wide range of NLP applications. Generative machine translation systems use it explicitly to verify the fluidity of the translation hypotheses, while descriptive translation models generally have one or more features corresponding to an n-gram language model. Reranking techniques for the parametrisation of the translation model in MERT-like learning algorithms often rely on a language model in order to distinguish the best among a list of the n-best translations. Another task that often uses n-gram models is Part-Of-Speech (POS) tagging. State-of-the-art POS taggers are trained over annotated corpora and learn some kind of conditional n-gram model in which the probability of a tag sequence $t_1 \dots t_m$ for a given sentence $w_1 \dots w_m$ depends on the current word but also on the n previous tags (Markov models):

$$P(t_1 \dots t_m | w_1 \dots w_m) = \prod_{i=1}^{m} P(w_i|t_i) P(t_i|t_{i-n+1} \dots t_{i-1})$$

The remainder of this report is structured as follows: section 2 discusses some related work on language identification, section 3 describes the resources used further in section 4, where we describe and discuss the results of each experiment. The overall conclusions and final comments are presented in section 5.

2 Related Work

Most of the literature in automatic language identification uses some variation of character-based n-gram model. Cavnar and Trenkle [1994] use a language *n-gram profile*, built upon the most frequent character n-grams in a text, with n running from 1 to 5. Then, they use an *ad hoc* "out-of-place" ranking distance measure to classify a newly arrived text into one of the existing profiles. They report 90% to 100% precision using a 300 n-grams profile, in detecting a text of 300 characters, and 95% to 100% for longer texts.

Dunning [1994] states that real-life applications need a precise language classification for texts up to 20 characters, showing the practical limitations of methods based on characteristic sequences and common words. He also criticises the approach of Cavnar and Trenkle [1994] since it depends on tokenisation and uses an empirical measure whose statistical properties are difficult to derive. Therefore, he proposes a method based on Markov chains and Bayesian models, similar to the one proposed here, that uses a log-probability sum as distance measure between a text and a language model.

A comparison of trigram-based and short-word-based techniques for language identification is described by Grefenstette [1995]. He uses a non-smoothed MLE probability distribution, and then compares one technique in which the distribution runs over trigrams and another in which the distribution runs over "common words" (5 letters or less, frequent words). The results show that trigrams are better for short sentences and that when larger sentences are considered, both methods perform equally well[1].

A short survey on language identification containing some pointers to other related work can be found in Muthusamy and Spitz [1997]. Given that language identification is an extremely simple problem compared to other NLP fields, it has the status of a "solved problem" that could be thought in undergraduate courses [McNamee, 2005]. The solution presented in this report aims at achieving state-of-the-art accuracy, that is to say correctly identify near to 100% of the sentences in a test set even for short sentences, using a relatively simple approach.

3 Resources

In order to verify our hypothesis, we will perform a series of experiments using n-gram language models. We will thus need two resources: a multilingual corpus for training and testing the models, and a language model toolkit. The following two sections present and describe these resources, that will be used throughout the remainder of this report.

3.1 Corpus

Data-driven, statistical or empirical methods on Natural Language Processing are different names for the same very active research field. During the last decades, the NLP community proposed many techniques that try to extract relevant information from corpora, mainly based on probabilistic tools. This report studies a stochastic approach to language detection and hence uses a multilingual corpus upon which the n-gram language models are built.

The *Europarl* corpus [Koehn, 1995] is a collection of parallel texts containing the transcription of official speeches held at the Parliament of the European

[1] This tool can be tested on `http://www.xrce.xerox.com/competencies/content-analysis/tools/guesser`

Union[2] from 1996 until 2006, along with their translations into 11 European languages. It is very popular in the Machine Translation community and most of the state-of-the-art statistical translation systems use it to train and evaluate translation (and language) models.[3]

In order to limit the effect of combinatorics, we focused our experiments on five languages: two Romanic languages – French (FR) and Portuguese (PT) – two Germanic languages – English (EN) and German (DE) – and Finnish (FI). Since each language contains approximately 1.5M sentences, we only used around 10% of the corpus. Two data sets were created: the *training set*[4] used for building the language models, and the *test set*[5], used to evaluate the language-detection models.

We started to clean the corpus by eliminating `XML` elements and blank lines. Then, we used the `perl` scripts provided with *Europarl* to split the paragraphs into sentences using simple punctuation and case heuristics and to tokenise the corpus separating punctuation marks from words (this was one of the most time-consuming steps of the whole experiments). Finally, we performed a language-dependent case homogenisation (convert all characters to lowercase letters).

Table 1 shows some characteristics of the resulting fragments of this pre-processing step. One could expect that the mean ratio between the size of the test set and the size of the training set would be around 1/12. However, for all data sets, this ratio is approximately 1/13. In short, we have a training set of 150K sentences (3M to 4M words according to the language) and a test set of 11K sentences (200K to 300K words) with sentences of 19 words for Finnish to 31 words for French, in average. This variability in mean sentence lengths is justified by the nature of the languages: Finnish has a small core lexicon and is highly synthetic whereas French is a verbose language. The characteristics of each data set are relevant because they allow a better understanding of the results obtained in section 4.

3.2 Language Model Tool

As shown by Manning and Schütze [1999], a large number of language models exist with applications in several NLP tasks. Some of these language models are implemented by freely available NLP packages, for instance the `CMU-Cambridge` Statistical Language Modelling package [Clarkson and Rosenfeld, 1997], and the `SRILM` toolkit [Stolcke, 2002]. The former no longer receiving support, the latter has become one of the most popular software packages in the NLP community, specially for Machine Translation and Speech Processing.

In the following experiments, we will use the version 1.5.7 of `SRILM`, available on `http://www.speech.sri.com/projects/srilm/`. We installed the package by compiling the source C++ files and tested it with the provided scripts. The

[2] `http://www.europarl.europa.eu/news/public/default_en.htm`

[3] The *Europarl* corpus is freely available at `http://www.statmt.org/europarl/`. In this report we use the *source* release, which is not sentence-aligned.

[4] Archives of the year of 2001

[5] Archives of December 2000

Table 1: Characterisation of the *Europarl* corpus fragment used in our experiments. Both data sets are described in terms of the number of sentences (#Sent), the number of words (#Words) and the average number of words per sentence (Length).

	Training set			**Test set**		
Lang	#Sent	#Words	Length	#Sent	#Words	Length
FR	147,979	4,590,539	31	11,020	329,861	29
PT	145,384	4,273,277	29	10,847	311,234	28
EN	149,371	4,163,389	27	10,961	304,073	27
DE	151,685	3,856,874	25	11,526	287,058	24
FI	149,887	2,965,049	19	11,181	218,919	19

package provides a large number of utilities to generate sophisticated language models, but we will only use a simple n-gram model with Good-Turing discounting and back-off probabilities. The command `ngram-count` generates and counts the n-grams of a given corpus. It also computes the language model log-probabilities, stored in the language model file (`.BO`), which correspond to the log-probability of an n-gram estimated from the given corpus. The option `-write-vocab` can be used to define a file in which the vocabulary is written, and the option `-order` n is used to define the order of the created n-gram model. The option `-gt`n`min` defines a threshold under which the n-grams of order n will not be considered. This avoids the computational overhead generated by the *long tail* phenomenon which characterises Zipfian distributions like words and n-grams in corpora.

Finally, the command `ngram` will be used to calculate the perplexity of a given text according to a model. Using a `-debug 1` option, we will be able to retrieve the perplexity of each sentence of the corpus and evaluate the model in a sentence-by-sentence basis.

4 Experimental Setup and Results

The experiments we made are subdivided in two steps: first, we perform a "toy" experiment in which we study the feasibility of a language identification tool using n-gram language models and perplexity. Then, we will make an extensive study of the influence of the different model parameters (model type, size of training corpus, order of the model, size of test sentences)on the overall performance of language identification. All the experiments described in the next session are temporarily available on `http://www.inf.ufrgs.br/~ceramisch/TP_SRILM.zip`.

4.1 Toy Experiment

The toy experiment consists of two steps: Firstly, we will build a 3-gram language model for each language of the *Europarl* training fragment described in session 3.1 using the `ngram-count` program of the `SRILM` package described in section 3.2. Secondly, we will calculate the perplexity of the test fragments of the *Europarl* corpus according to the model of each language using the `ngram` command described in section 3.2. In this section, all the intermediary steps and files will be explained in detail for the French data sets.

The creation of the language model is done through the following commands:

```
ngram-count -write-vocab fr.voc -text ep-fr.txt -write fr.bo
ngram-count -read fr.bo -lm model-fr.BO -gt3min 2 -gt2min 1
```

The first command creates the vocabulary (`fr.voc`) and the n-gram counts file (`fr.bo`) of the *Europarl*(ep) French(fr) training corpus (`ep-fr.txt` file). The order of the model may be explicitly defined through the option `-order`, but we used the default value (3-gram model). We ignore all *hapax* bigrams and all the trigrams that appear twice or less in the corpus, using the `-gt`n`min` option. Table 2 shows the top-10 most frequent trigrams in the French training data, obtained from the `fr.bo` file, most of them composed by recurrent expressions in political speeches and random combination of function words and punctuation. The n-grams generated in the `fr.bo` file contain special markers `<s>` and `</s>`, used as sentence delimiters. This is in conformity with the assumption that sentences are independent and that the beginning of one sentence is not influenced by the last words of the previous sentence.

Table 2: Most frequent trigrams in the French part of the training corpus.

Trigram	Frequency
monsieur le président	7738
de l '	7645
le président ,	7114
de la commission	6809
<s> monsieur le	6162
l' union européenne	4451
<s> c' est	3828
de l' union	3500
à l '	3287
, monsieur le	2728

We calculated the perplexity, as described in section 1, of each training set according to each language model, using the command:

```
ngram -lm fr.BO -ppl ep-fr-test.txt
```

We compared each test set with each language model using the previous command. The results are shown in table 3. We consider that a test text

Table 3: Perplexity of each test set according to each language model. The minimum values of each line are emphasised, a minimum on the diagonal is a correctly identified test set. The proportion of sentences correctly identified is shown in the last column and the size of the vocabulary (in number of words) for each model figures on the last line.

	Lang. Model					
Test set	FR	PT	EN	DE	FI	Precision
FR	**78**	13,931	29,350	28,449	26,062	98.27%
PT	2,558	**109**	3,579	10,985	9,201	88.35%
EN	42,037	35,707	**96**	34,355	31,200	99.52%
DE	43,830	16,286	13,820	**168**	17,008	93.32%
FI	740	497	**221**	760	543	15.66%
Voc. size	39,569	47,958	29,882	67,746	169,261	

is classified as being in the language of the model for which the perplexity is minimal. Therefore, we emphasised the minimal values in the table, and minimal values in the diagonal represent a correctly identified language. We observe that for all but Finnish, the language was correctly identified.

However, this is a simplistic binary evaluation measure for the models, since it only allows us to say whether the language was correctly identified or not. Therefore, we output the perplexity for each sentence of the test corpus using the `-debug 1` option and then calculated the minimal perplexity for each sentence. Then, we define the precision of a given model as the number of sentences correctly identified over the number of sentences in the test set. The last column of table 3 shows this precision measure for each test set. We observe that, for all languages but Finnish, the models perform relatively well, especially for English. According to Cavnar and Trenkle [1994], Portuguese is one of the most difficult languages to identify. This confirms that it has a low precision (88.35%) compared to English (99.52%). Finnish is the exception in this table, having a surprisingly low precision (lower than a random uniform assignment of languages to sentences). We underline that it also has a vocabulary that is 2 to 3 times bigger than the other languages, suggesting that there is a correlation between the size of the vocabulary and the performance of the model. Two solutions to this problem are possible: either we use a bigger training set in order to obtain better estimations for the n-gram probabilities, either we diminish the size of the alphabet used in the construction of the model. Since the former considerably increases the computational cost of the method, we will investigate the latter option in the next section.

4.2 Extensive Experiments

In this section, we will run four experiments in order to study some parameters of the language models and the impact of these parameters on the performance

Table 4: Perplexity of each test set according to each character language model. The minimum values of each line are emphasised, a minimum on the diagonal is a correctly identified test set. The proportion of sentences correctly identified is shown in the last column and the size of the vocabulary (in number of symbols) for each model figures on the last line.

	Char. Lang. Model					
Test set	FR	PT	EN	DE	FI	Precision
FR	**5.78**	22.73	17.85	24.67	37.28	99.88%
PT	24.14	**6.02**	22.15	31.65	37.69	99.89%
EN	20.71	23.87	**6.29**	20.31	34.1	99.91%
DE	36.25	40.8	26.1	**6.11**	38.16	99.91%
FI	84.08	96.71	75.36	47.69	**6.49**	99.94%
Voc. size	90	90	86	94	88	

of the models. First, we compare a character-based language model to the previously built word-based language model (§ 4.2.1), then we study the impact of the size of the training corpus (§ 4.2.2) and of the order of the model (§ 4.2.3) on the overall performance. Finally, we will validate the models on different sentence lengths, showing that it performs well even if test sentences are quite small (§ 4.2.4).

4.2.1 Experiment 1: Model Type

As described in section 4.1, we are interested in reducing the size of our vocabulary. The literature of language detection suggests that a character language model could be used instead of a word-based model. In this case, the vocabulary will be restricted to the alphabet, i.e. the number of letters, punctuation symbols, accentuated letters, whitespace and special symbols in the training set. Therefore, we used the same approach of the previous section on the preprocessed training and test corpora. We separated each character from each other by a whitespace and replaced the whitespace by a special underline ("_") symbol. Sentences are still considered independent from each other but the last character of one word is considered to influence the first character of the next.

The results of this experiment can be seen in table 4, following the same convention of the previous section. Now, we have a vocabulary that ranges from 86 to 94 letters and all the minimal values are in the diagonal, meaning that all languages were correctly identified. Word-based models were also able to identify similar languages: the second smallest perplexity of the French test set is Portuguese and vice versa, and the same can be observed between English and German, correctly grouping Latin and Germanic languages. This does not seem to be the case for character language models, since they do not take the lexicon of each language into account.

If we look at the per-sentence precision of the method, we verify that we

Table 5: Precision (% of correctly identified sentences) of each test set according to each character language model built on different fragments of the original training set. The last column symbolises the precision increase of multiplying the training set size by 1000.

	Proportion of original training set					
Test set	0.1%	1%	10%	50%	100%	Gain
FR	99.62%	99.79%	99.86%	99.88%	99.88%	.26%
PT	99.65%	99.86%	99.86%	99.89%	99.89%	.24%
EN	99.76%	99.87%	99.89%	99.90%	99.91%	.15%
DE	99.82%	99.9 %	99.91%	99.92%	99.91%	.09%
FI	99.84%	99.90%	99.95%	99.94%	99.94%	.10%

reach very high precision values, around 99.9% for all test sets. This shows us that a simpler model, i.e. a character model instead of a word model, is better fit to the task of language detection. We will thus use character models for all the following experiments, where we would like to investigate whether even simpler models could still reach such high precision.

4.2.2 Experiment 2: Size of Training Set

One could ask himself whether the good performance of the models created for the first experiment are dependent of the size of the corpus on which the character model was trained. We try to answer this question by training models on fragments of the original training set and comparing the performance with the model for the whole training data. We used fragments of 0.1%, 1%, 10% and 50%, e.g. for the French training set, who originally contains 147,979 sentences, we built a model using only 147 sentences, then using 1479 sentences, and so on.

The precision of each model is shown in table 5. With a training set that is 1000 times smaller than the original set, we obtained precisions between 99.6% and 99.8%, i.e. an extremely gracefull degradation compared to the 100 % model. Indeed, multiplying the corpus size by 1000 does not lead to a 1000 times better performance: the performance gain, shown in the last column, is neglectful, reaching at most 0.26%. This shows that character n-gram models could also be applied to resource-poor languages that do not dispose of large corpora, as a model built upon some hundred sentences is already very accurate.

4.2.3 Experiment 3: Order of the Model

Building n-gram models may be computationally expensive if we have a large language set for which we need to build character models. The time taken to build a model is exponential to the order of that model, so it would be of great value if we could obtain comparable results using a lower order model. In this

Table 6: Precision (% of correctly identified sentences) of each test set according to each character language model built considering different values for n, i.e. models of order 1 to 5.

	Order of n-gram model				
Test set	$n = 1$	$n = 2$	$n = 3$	$n = 4$	$n = 5$
FR	93.21%	99.61%	99.88%	99.91%	99.92%
PT	93.47%	99.71%	99.89%	99.94%	99.96%
EN	94.02%	99.67%	99.91%	99.89%	99.91%
DE	92.45%	99.81%	99.91%	99.96%	99.97%
FI	98.21%	99.88%	99.94%	99.97%	99.97%

experiment, we compared the performance of unigram, bigram, trigram, 4-gram and 5-gram models in language identification.

Table 6 shows that a precision above 90% may be obtained with a model as simple as counting the letters of the text. This is quite surprising and possibly dependent of the number of languages that need to be identified: if we had a large language set, perhaps we would find more similarities between different languages and we would mistake ourselves more frequently. However, we cannot observe a real performance increase when going from bigrams to 5-grams, suggesting that a bigram model could be used in this task, considerably reducing the processing time and the memory used to store the models. Regardless of the results obtained here, Manning and Schütze [1999] affirm that a trigram model performs as well as a human in predicting the next word of a text and Dunning [1994] also uses trigrams for language identification. A further investigation with a bigger language set would be necessary in order to investigate whether bigram models would be as efficient as trigram models in language identification.

4.2.4 Experiment 4: Size of Test Sentences

Dunning [1994] criticises the related work on language identification because they evaluate their models using large sentences as input. He gives examples of applications in which sentences containing as few as 20 characters need to be correctly identified. In this section, we slice the test corpus according to the number of characters in the sentences and compare inter-slice performances. Therefore, we considered three different slices: slice 1 contains only sentences that have 75 characters or less, slice 2 contains the sentences that have more than 75 characters and 150 or less characters, and slice 3 contains big sentences with more than 150 characters. The slice boundaries were established empirically so that slices have comparable sizes.

The last line of table 7 shows the minimum and maximum size for each slice, confirming that they have comparable sizes thus avoiding a bias in the evaluation. Small sentences are recognised with a precision above 99%, while we obtained 100% precision for longer sentences in Portuguese, German and Finnish. Average sentences are correctly identified more than 99.9% of the

Table 7: Precision (% of correctly identified sentences) of each slice of the test set. Test set was sliced according to the number of characters in each sentence. Maximum and minimum size of each slice are shown in the last line.

	Size of sentences in test set		
Test set	$0 - 75$ chars	$75 - 150$ chars	> 150 chars
FR	99.35%	99.94%	99.98%
PT	99.68%	99.97%	100%
EN	99.64%	99.97%	99.97%
DE	99.53%	99.97%	100%
FI	99.68%	99.97%	100%
#Sentences	1,549 – 1,933	3,175 – 3,490	3,715 – 4,462

times. This shows us that even sentences smaller than 75 characters may be correctly identified using n-gram models.

When we created our training and test corpora, no manual verification was made to eliminate sentences in a foreign language. Therefore, a precision of 100% in the whole test set is nearly impossible, since one can find some pathological cases among the test sentences. For example, the Spanish sentence *hay unas palabras españolas muy bien conocidas en mi país que son las palabras " no pasarán ! "* . is found in the French test corpus and is systematically identified as Portuguese, instead of French, because it is the nearest language in our set corresponding to Spanish.

5 Conclusions

Language identification has the status of a "solved problem" in NLP. We made an extensive study of this task using n-gram language models. Our first hypothesis, based on related work on language identification, was that n-gram models can be used to detect the language in which a texts was written. Therefore, we proposed a simple detection method where we compute models for different languages and then compare a test text to each model using the cross-entropy (actually, perplexity) between the text and the models. We classify the text into the language for which the perplexity was minimal.

In order to try to prove our hypothesis, we used a fragment of the *Europarl* corpus and the `SRILM` toolkit, evaluating the models both globally and sentence-by-sentence. The first attempt to use a word-based trigram model showed that it is possible to detect the language given that it has a relatively small vocabulary, which is not the case for Finnish. Then, we proceeded our experiments and showed that character-based models are better fit for the task of language identification. According to our results, a few hundred sentences are enough to achieve good performances. We also suggest that bigram models could also be used without dramatically decreasing precision, and that using 4-gram and 5-

gram models does not improve the accuracy of the method. Finally, we showed that our method is efficient in identifying long sentences as well as short sentences of less than 75 characters. Although the heterogeneity of the languages used in the experiments, these results confirm our initial hypothesis.

It is important to realise that the methods described in this report are simple, easy to implement and totally stochastic thus language-independent. Since we did not use any hand-built dictionary, it is possible to apply this method for resource-poor languages, in particular because it works surprisingly well even with quite small corpora. Additionally, no prior knowledge of the languages is needed, as the author of this report does not speak a single word of Finnish. Even though several sophisticated NLP problems need some form of linguistic knowledge, simple problems like language identification clearly take advantage of statistical methods over knowledge-based methods.

A number of free and commercial NLP packages implement a language identification module: TextCat, LingPipe, Xerox MLTT Language Identifier, Google's AJAX API for Translation and Detection, etc. Most of them, if not all, use variations of the technique presented here. Orthography verification software generally uses language detection, as well as some Machine Translation systems. With this extensive study, we hope to have shown the importance of this simple but very important "solved problem" that has many practical applications in NLP systems.

References

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

Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. In *Proceedings of ACL*, pages 310–318, 1996.

Philip Clarkson and Ronald Rosenfeld. Statistical language modeling using the cmu-cambridge toolkit. In *Proceedings of ESCA Eurospeech*, pages 2707–2710, 1997.

Ted Dunning. Statistical identification of language. Technical report, New Mexico State University, 1994.

Gregory Grefenstette. Comparing two langage identification schemes. In *JADT*, 1995.

Slava M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(3):400–401, 1987.

Philipp Koehn. Europarl: A parallel corpus for statistical machine translation. 1995.

Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, USA, 1999.

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

Yeshwant K. Muthusamy and A. Lawrence Spitz. Automatic language identification. pages 273–276, 1997.

Andreas Stolcke. Srilm – an extensible language modeling toolkit. In *Proceedings of ICSLP Interspeech*, pages 901–904, 2002.