Language Independent Statistical Models for on-Line Handwriting Recognition

Freddy Perraud, Christian Viard-Gaudin, Emmanuel Morin

Language Independent Statistical Models for on-Line Handwriting Recognition

Freddy Perraud
Vision Objects
Freddy.Perraud@visionobjects.com

Christian Viard-Gaudin
IRCCyN UMR CNRS 6597
Christian.Viard-Gaudin@univ-nantes.fr

Emmanuel Morin
LINA FRE CNRS 2729
Emmanuel.Morin@univ-nantes.fr

Abstract

This paper deals with a language modeling approach that is dedicated to an on-line handwriting recognition system. Three main goals are set: i) performances, ii) versatility, and iii) resources. To achieve these goals we propose a statistical word n-class approach, which uses a learning stage to cluster words in classes and defines an estimation of the probability distribution of sequences of classes.

Very large corpora from three different languages (English, French and Italian) have been used to train and test the language models. The efficiency of these models are evaluated not only from a linguistic point of view, using perplexity measurements, but also combined inside the recognition system on real ink signals corresponding to written sentences. Using a tri-class model allows a word error rate reduction ranging from to 50 to 60% according to the language.

Keywords: handwriting recognition, language modeling, n-gram, n-class, perplexity.

1. Introduction

Statistical language models are widely present in numerous applications. They are integrated in speech recognition systems, translation tools, predictive entry systems, indexing document facilities, grammatical correctors, etc. Clearly, they can also provide some convenient helps to increase the performances of handwriting recognition systems when contextual information can disambiguate poorly written texts.

Let us consider the following examples as shown in Figure 1. For every expression, one word has been hidden. Nevertheless, it is quite easy, even without any visual clue, to guess the correct missing word. Conversely, if we consider the word that is displayed in Figure 2, with no context available (Swedish word), it is very difficult to identify the correct spelling. Hence, it is possible to recognize without seeing, and to see without recognizing; these statements show that pattern recognition information and language knowledge are complementary information.

In this paper, we aim at taking advantage of the context by defining a language model that allows increasing the performances of a non-constraint on-line handwriting sentence recognition system. We would like this system to be versatile enough so that with little effort it can be adapted from one language to another. And third, the size of the resources requested by the language model should be compatible with small mobile devices, such as Smartphones and digital pens.

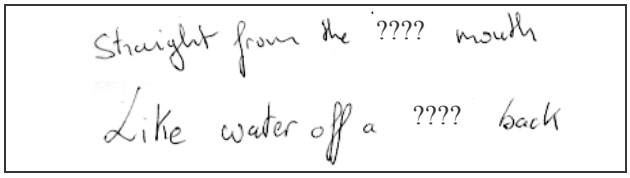

Figure 1. Recognizing without seeing: horse's, duck's.

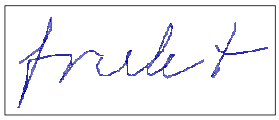

Figure 2. Seeing without recognizing.

The language model will be incorporated within the recognition system and will be linked to the pattern recognition results with the Bayes relation:

$$p(s|x) = \frac{p(x|s)p(s)}{p(x)} \quad (1)$$

$$\hat{s} = \arg\max_{s} \{ p(x \mid s)\, p(s) \} \quad (2)$$

Within these relations, x stands for the writing trace signal, s for a given sentence, and $\hat{s}$ for the recognized sentence using the MAP criteria. Eq. 2 implies the computation of $p(s)$ for every possible sentence. This is the main issue related to language modeling: how to produce a realistic estimation of $p(s)$ for a given sentence. It will be addressed by using a statistical approach where large corpora are used to train the language models. In this paper, we compare statistical n-gram models defined either at the word or at the class level, these latter being termed as n-class models in the following notations.

2. Stochastic language modeling

We can define a sentence, s, as a sequence of L words w_i or $s = w_1 \ldots w_i \ldots w_L$. A specific sentence can be interpreted as a realization of a discrete stochastic process. Moreover, if it is assumed that the evolution of

the process — the next word — depends only on its present state — the already written text —, the general framework of Markov processes is applicable. In addition, if the set of states is discrete, here it is defined by the lexicon, then the system can be modelled by a Markov chain [4].

$$p(s) = \prod_{i=1}^{L} p\left(w_i \middle| w_1 \cdots w_{i-1}\right) \quad (3)$$

In other words, for every position in the sentence $w_1 \ldots w_i \ldots w_L$, the language model is able to compute a probability value $p(w_i|w_1 \ldots w_{i-1})$ for every possible next word w_i, satisfying the normalization constraint:

$$\sum_{j=1}^{V} p\left(w_i = w^j \mid w_1 \cdots w_{i-1}\right) = 1 \quad (4)$$

where w_j stands for the j^{th} word of the lexicon (with V the size).

The longer the past word sequence, the less reliable the estimate of the conditional probability value since such a sequence is not likely to have been encountered during learning. To overcome this limitation, a reduction in the order of the Markov chain is used. It reduces the influence of the past words to only the *n-1* previous words. This allows defining the language model by using the so-called n-gram probabilities:

$$p\left(w_i \middle| w_{i-n+1} \ldots w_{i-1}\right) \approx p\left(w_i \middle| w_1 \ldots w_{i-1}\right) \quad (5)$$

By replacing in Eq. 3, we get:

$$p(s) = p\left(w_{1,L}\right) = \prod_{i=1}^{L} p\left(w_i \middle| w_{i-n+1} \ldots w_{i-1}\right) \quad (6)$$

We thus define the well know n-gram model where only the most recent words (one word for bigram, two words for trigram…) of the history are used to condition the probability of the next word. The computation of the model parameters requires very large learning corpora. With such corpora, it is possible with a basic counting approach to estimate these n-gram probabilities.

$$p\left(w_i \middle| w_{i-n+1} \ldots w_{i-1}\right) = \frac{c\left(w_{i-n+1} \ldots w_i\right)}{c\left(w_{i-n+1} \ldots w_{i-1}\right)} \quad (7)$$

where c(.) stands for the number of corresponding events.

A common problem with statistical language models is that the amount of training data is often insufficient, and as a consequence, estimations are not reliable due to sparse data. In order to have more reliable estimates and to deal with unseen n-grams, it is essential to incorporate smoothing techniques (such as additive smoothing, Good-Turing estimation or back-off smoothing). Based on the results of various experiments, we have selected the absolute discounting backing-off algorithm [2].

One severe limitation of the n-gram models is their size: n-gram models construct tables of conditional probabilities for each next word, in a large number of contexts, i.e. combinations of the last *n-1* words. Thus, the total number of parameters for these models is basically V^n, with V being the size of the lexicon. For large size vocabulary, i.e. several thousands of words – 400.000 words for the French lexicon – even with $n = 2$, there are already too many parameters to store in a mobile device. Putting the size issue aside, there is much more information to help predict the word in the sequence that immediately precedes the predicted word than in the identity of the previous words. Semantic and grammatical roles of these words are meaningful to generalize the computation to unseen situations.

To address these two issues, model size and generalization capability, the proposed approach is based on learning a clustering of words: each word is associated deterministically or probabilistically with a discrete class, and words in the same class are similar in some respect [7], [1].

Once a language model is available, you should be able to measure its performances. This allows optimization of the parameters settings of the model and to compare different models. Two main criteria have been considered here. The first is the precision of the model: it will be evaluated by the measurement of the perplexity. The second is the error rate of the handwriting recognition systems.

3. Statistical n-class models

The objective with statistical classification is to gather in the same class words that share the same lexical context. Every word w_i is associated to a class $g(w_i) = g_k$, with $k \in [1 \ldots K]$ (K being the number of classes). The conditional probability $p(w_i|w_{i-n+1} \ldots w_{i-1})$ is then given by:

$$p\left(w_i \middle| w_{i-n+1} \ldots w_{i-1}\right) = p\left(w_i \middle| g(w_i)\right) p\left(g(w_i) \middle| g(w_{i-n+1}) \ldots g(w_{i-1})\right) \quad (8)$$

For a bi-class model we use the following formula:

$$p\left(w_i \middle| w_{i-1}\right) = p\left(w_i \middle| g(w_i)\right) \cdot p\left(g(w_i) \middle| g(w_{i-1})\right) \quad (9)$$

It requires only $V + K^2$ parameters to be computed, compared with V^2 ($K << V$) for the bi-gram model. In this approach, the main difficulty becomes the definition of the *g()* function that assigns a class to a word.

With such a model, the main difficulty is to build relevant word classes: with two different classifications, the model will not have the same performance. Obtaining the best classification is not a straightforward result. When we have to cope with a corpus of several millions of words, gathering them in class according to their context turns out to be very costly. Most efficient methods proceed iteratively by finding a better classification at the $n+1^{th}$ iteration compared to the classification established at the n^{th} iteration, according to a given criterion. The criterion used to evaluate the quality of the current classification is directly linked to the quality of the n-gram model, i.e. its perplexity. Consequently, searching the best classification consists in determining the classification which provides the lowest perplexity (cf. Eq 10).

A primitive method to determine this optimum would be to test all possible classifications. If we take into account that for a 20 000 word lexicon split into 100 classes, there are already 100^{20000}, which is 10^{40000}, different classifications.

Sub-optimal but much more efficient methods exist. We used the method proposed in [6], based on the Expectation-Maximization algorithm.

First, words are sorted according to their decreasing number of occurrences, computed on a training corpus. The first *K*-1 words define the *K*-1 classes while the last class contains all other words of the ranked list. Afterwards, at each iteration, all words are moved one at a time in every class. Using this method, each neighbour solution is explored. With a 20 000 word lexicon split into 100 classes, the algorithm computes 100×20 000 = 2.10^6 configurations for each iteration. For every attempt, perplexity is measured and the classification reached at the end of an iteration is used to start the next. In most case, only 3 or 4 iterations are needed. Next iterations will provide an improvement less than 1%.

Many mathematic and algorithmic tricks are used to execute this algorithm in a reasonable time [6].

4. Experiments and results

4.1. Integrating language models into the recognition system

Most of the integration of language models in a recognition system first deals with the handwriting model alone and then reorders the list of *N*-best results by taking into account the output of the language model. The probability product of both models defines the overall likelihood function of a sentence. Once the re-scoring is achieved, only the top result is kept and proposed as the output of the recognition task [9].

When vocabulary becomes very large, building a complete list in real time is often too costly in terms of computation. To overcome this, an algorithm, such as *beam-search* algorithm, can be used to find the most likely sentences in the trellis of solutions where only the *N*-Best hypotheses are kept.

In these conditions, it is shrewd to integrate the language model as soon as possible in the recognition process, preferably at the construction of the trellis. Probabilities from the language model are involved in the re-scoring of the *beam-search* hypotheses as the handwriting signal becomes available. This is the procedure we have implemented here as it is more efficient than a post-recognition integration as the language model cannot recover 'out of the *beam-search'*- hypotheses.

4.2. Linguistic resources and handwriting databases

Table 1 shows the main features of linguistic resources involved in training and test of our language models. Novels from XIXth and XXth centuries, newspapers articles, and web pages are used to build these resources. They stand for corpora of general language in English, French and Italian.

We also built up an online handwriting ink database for these three languages (see Table 2) with the help of paper forms and digital pens. The samples of these databases are labelled at sentence level. In order to compute a word error rate at word level, a string-to-string distance measure [10] is calculated between the sequence of words outputted by the recognition system and the label of the given sample. Databases presented in Table 2 correspond to those used to test the recognition system integrating language models. They are independent of those used to train the neuro-markovian system.

Table 1. Linguistic resources description.

Language	Training corpus size	Test corpus size	Lexicon size
English	1 Gwords	15 Mwords	137 Kwords
French	150 Mwords	24 Mwords	452 Kwords
Italian	80 Mwords	13 Mwords	1,6 Mwords

We conducted experiments using language models of different orders: uni-gram, bi-class and tri-class. The number of classes of the bi-class and tri-class models was set to 256 to ensure that one class index corresponds to one byte. As shown in [6], the more the number of class is important, the more the models are performing. Moreover, n-gram models perform better than n-class models.

In the following section, we will investigate the impact in terms of perplexity, word error rate, model size and computation time depending on the order of the model, the number of classes and the pruning threshold of rare events. In the following graphs, language models are arranged in ascending order according to their size; uni-gram, bi-class, pruned tri-class and complete tri-class.

Table 2. Description of test handwriting databases.

Language	Number of sentences	Number of words	Number of scripters
English	541	3 861	320
French	719	10 044	270
Italian	697	8 866	353

4.3. Language model performances

Performances are studied based on two criteria. On one side, we compare measure of perplexity computed on test text databases (see Table 1), on the other side, we are interested in word error rates on handwriting databases (see Table 2). Perplexity is easier to compute than word error rates that involve the recognition system and a full integration of the language model. The impact of perplexity on recognition performances is not obvious [3]. Contrary to the field of speech recognition, only few advanced experiments were carried out in the handwriting recognition field [5], [8]. One of the aims of

our work is to empirically demonstrate that perplexity and word error rates are correlated.

Figure 3 shows the evolution of perplexity measure (see Eq. 10) on the three different languages depending on the different language models involved.

$$PP_M(T_{test}) = \left[\prod_{i=1}^{L} p\left(w_i \middle| w_{i-n+1} \ldots w_{i-1}\right) \right]^{-\frac{1}{L}} \quad (10)$$

It is important to keep in mind that the lower the perplexity, the better the language model. As its name indicates, the perplexity expresses the average uncertainty of the language model on a given text. Thus, in the case that the model hesitates equiprobably among all the words in the lexicon, the perplexity is equal to the lexicon size. This measure is strongly linked to cross-entropy between the model and the text (Perplexity = $2^{\text{Cross-Entropy}}$).

From Figure 3, we can observe a steady improvement of the language model performances when switching from a uni-gram model to a bi-class model to a tri-class model. This is also valid for a tri-class model itself: the less it is pruned, the better the model. Pruning values from 100 to 1 correspond to the absolute threshold of the tri-class counts under which events are not taken into account. For instance, with a cut-off threshold value equal to 3, all tri-class with an occurrence lower or equal to 3 are remove from the model.

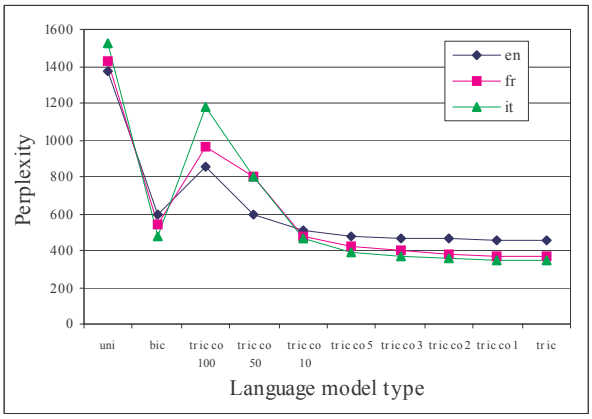

Figure 3. Perplexity measures on test corpora.

Recognition results on handwriting databases are shown in Figure 4. One can notice the strong correlation between the curves in Figure 3 and 4 even though some pruned tri-class models perform better than complete tri-class models. For instance, on English databases, the best model is the pruned model that corresponds to an absolute threshold of 10. Pruning provides a double positive effect. First, it reduces the size of language models, as shown in Figure 8. Moreover, it reduces the variance of the estimations given by the language model and lead to a better generalization of the samples in the handwriting databases.

The results in Table 3 are issued from the curves in Figure 4 and indicate the relative improvement of word error rates from one model to another. Integration of just a uni-gram model leads to a relative decrease of the word error rate from 32% to nearly 55%, depending on the language. With the help of bi-class model, word error rate is further reduced from 8% to 18%. In addition, switching from bi-class to tri-class provokes a small relative improvement from 3.1% to 4.9%. In conclusion, involving a tri-class model decreases significantly the word error rate to nearly 50% for French and Italian and more than 60% for English.

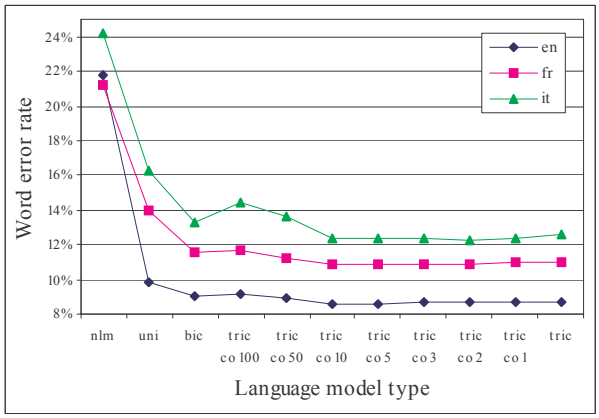

Figure 4. Word error rate evolution according to language model (nlm: no language model ; uni: uni-gram model ; bic: bi-clas model with 256 classes ; tric: tri-class model with 256 classes ; tric co-xxx: tri-class model with an absolute pruning threshold set to xxx).

Table 3. Relative decrease of word error rate.

Language	No model to Uni-gram	Uni-gram to Bi-class	Bi-class to Tri-class	No model to tri-class
English	54.8%	8.2%	3.1%	60.3%
French	34.4%	16.7%	4.9%	48.9%
Italian	32.6%	18.7%	4.7%	49.0%

Figures 5, 6 and 7 show 3 samples of sentences extracted from the French, English and Italian handwriting database. For each one of them, respectively Table 4, 5 and 6 give the recognition system output according to the model type. In order to compute word error rate, we use the string-to-string edition distance. Substitution and insertion costs are set to 1 whereas deletion cost is set to 0.

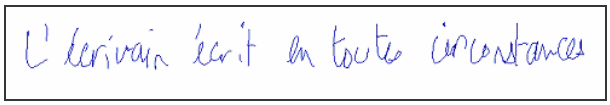

Figure 5.« L'écrivain écrit en toutes circonstances » (the writer writes in any circumstance).

Table 4. Evolution of Recognition.

Language model	Recognized sentence	Word error rate
None	L'écrivain écrit **ln** toutes **inconstancies**	2/6
Uni-gram	L'écrivain écrit en toutes circonstances	0/6

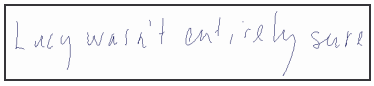

Figure 6. « Lucy wasn't entirely sure ».

Table 5. Evolution of Recognition.

Language model	Recognized sentence	Word error rate
None	Lucy **wash t** entirely sure	2/4
Uni-gram	Lucy **wash t** entirely sure	2/4
Bi-Clas	Lucy was **at** entirely sure	1/4
Tri-Class	Lucy wasn't entirely sure	0/4

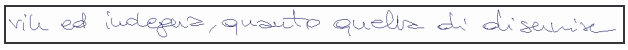

Figure 7. « vile ed indegna, quanto quella di disunire.» (vile and unworthy than that of separation).

Table 6. Evolution of Recognition.

Language model	Recognized sentence	Word error rate
None	**via** ed indegna, quanto quella di **disunisco**	2/8
Uni-gram	**via** ed indegna, quanto quella di **di smise**	2/8
Bi-Clas	vile ed indegna, quanto quella di **di servire**	1/8
Tri-Class	vile ed indegna, quanto quella di disunire	0/8

These samples clearly illustrate the progressive contribution of various language models. Before having a look at the memory cost and computational cost induced by these models, it is worth observing the content of the constructed classes.

4.4. Word classes features

Word classes based on grammatical criteria such as the part of speech tags or the morphological analysis of words contain homogeneous linguistic entities. For instance, we could find classes composed of a determiner, a conjunction, or a masculine noun.

In our approach, word classes are constructed through a contextual criterion. Therefore, word classes are not a priori significant and contain heterogeneous linguistic entities. Since word classes are produced by clustering techniques based on similar distributional contexts, they reveal words that share the same left and right lexical environments. Thus, we notice some similarities between words belonging to the same class. For instance, some word classes share syntactic (see classes 2 and 4) or semantic (see classes 1 and 3) features. Some other word classes seem to have no relevant meaning (see class 5).

The following word classes are induced from the French learning corpus:

Class 1 (country or area beginning with a vowel): Afghanistan *(Afghanistan),* Alaska *(Alaska),* Albanie *(Albania),* Alexanderplatz *(Alexander place),* Algérie *(Algeria),* Alsace *(Alsace),* Amazonie *(Amazonia),* Ameublement *(Furnishing),* Andalousie *(Andalusia),* Angleterre *(England)* ...

Class 2 (adverb): éminemment (*eminently*), affreusement (*terribly*), anormalement (*abnormally)*, désespérément (*desperately*), disco (*disco*), rock (*rock*), extremement (*extremely*), extraordinairement (*extraordinarily*), faussement (*falsely*), follement (*madly*)…

Class 3 (month): février (*february*), grès (*stoneware*), injectant (*injecting*), juillet (*july*), juin (*june*), mai (*may*), mars (*march*).

Class 4 (present participle): écartant (*diverting*), échangeant (*exchanging*), écoutant (*listening*), élargissant (*widening*), éliminant (*liquidating*), élisant (*esquiving*), éloignera (*will move away*), émettant (*emitting*), étouffant (*choking*), évitant (*avoiding*)…

Class 5: conquerront (*will conquer*), déballer (*to unpack*), écoute (*listening*), même (*even*), quadruplement (*quadruplet*), trotter (*to trot*).

We also notice that frequent words and specifically stop words (such as determiner or preposition) are isolated in classes with only a few words, even only one. Likewise, punctuation marks and tokens used to delimitate sentences are always isolated in separated classes.

4.5. Language model size

Figure 8 shows the language model size evolution.

Uni-gram and bi-class model sizes are reasonable even for mobile devices (nearly 1 Mo for English and French and less than 4 Mo for Italian which has the disadvantage of having a very large lexicon). However, tri-class models take up a large amount of memory space (nearly 13 Mo for English). Nevertheless, by pruning a very significant amount of memory space can be saved (less than 5 Mo with a pruning threshold set to 10). As shown in figure 4, recognition performances are hardly decreased or even improved.

There are some significant differences in model size according to the language. The main parameters that have an impact on the model size are:

- the size of the lexicon (which explains the large size of the Italian language model at order 1 and 2);
- the size of the training corpus (which is why the size of the complete tri-class model for English is important as most of the possible events are encountered).

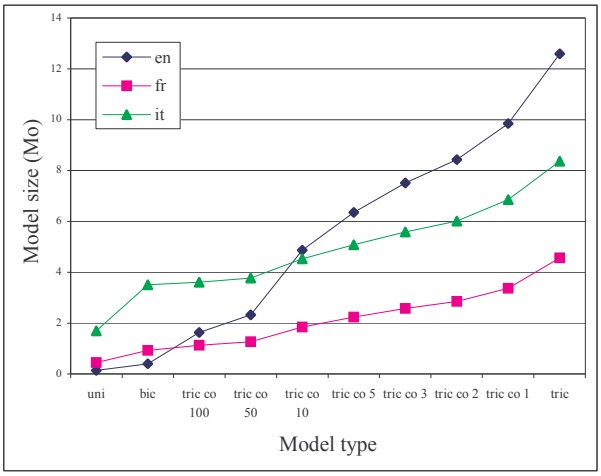

Figure 8. Model size evolution depending on model type.

4.6. Computation time

We have also studied the average recognition time by normalizing it by the average time of the bi-class model as shown in figure 9. According to the results, language models have a small influence on the recognition time. Compared to bi-class models, the extra computational time induced by tri-class models is augmented from 5 to 15 %. The drop gained with the help of mono-gram model varies from 7 to 15%, still compared to bi-class models. These small variations are the result of the structure of the recognition system. This system explores the recognition graph with the help of the *beam-search* algorithm whose depth is fixed, resulting in a certain independence of the size of the language model.

5. Conclusion

We have defined and experimented a statistical language model based on n-class probabilities. The classes are built with a non-supervised iterative algorithm by optimizing the perplexity on a training corpus. This scheme can easily be adapted from one language to another; the only requirement is an access to large text corpora in the new language. The number of classes being much smaller than the number of words, it downsizes considerably the size of the language model. Different orders for the model have been compared (uni, bi, and tri-class model), and it turns out that a pruned tri-class model remains of reasonable size when using 256 classes, and it exhibits a word error rate reduction ranging from to 50 to 60% according to the language. A next step would be to combine such statistical models with syntactical models which classes result from the grammatical nature of the words. A slight improvement can be awaited as these two kinds of models are rather uncorrelated.

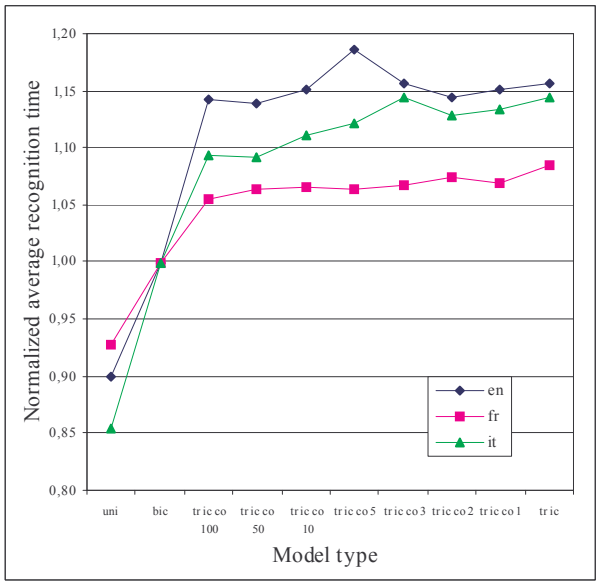

Figure 9. Normalized average recognition time.

References

[1] L.D. Baker, A.K. McCallum, "Distributional clustering of words for text classification", SIGIR, 21st ACM International Conf. on Research and Development in Information Retrieval, Australia, pp. 96–103, 1998.

[2] J. Goodman, S.F. Chen, "An empirical study of smoothing technique for language modelling", Vol. TR-10-98, 1998.

[3] D. Klakow, and J. Peters, "Testing the correlation of word error rate and perplexity", *Speech Communication*, Vol. 38, N°1, pp. 19–28, 2002.

[4] C. Manning, H. Schutze, Foundation of Statistical Natural Language Processing, The MIT Press, 2000.

[5] U.V. Marti, H. Bunke, "Unconstrained Handwriting Recognition: Language Models, Perplexity, and System Performance", IWFHR VII, pp. 463–468, 2000.

[6] S. Martin, H. Ney, J. Liermann, "Algorithms for bigram and trigram word clustering", Speech Communication, pp. 19–37, 1998.

[7] T. Niesler, E.W.D. Whittaker, P.C. Woodland, "Comparison of part-of-speech and automatically derived category-based language models for speech recognition", In International Conference on Acoustics, Speech and Signal Processing,, pp. 177–180, 1998.

[8] F. Perraud, C. Viard-Gaudin, E. Morin, P.M. Lallican, "Statistical Language Models for On-Line Handwriting Recognition", *IEICE Transactions on Information and Systems/Document Image Understanding and Digital Document,* Vol. E88-D, N°8 pp. 1807–1814, 2005.

[9] A. Vinciarelli, S. Bengio, H. Bunke, "Offline Recognition of Unconstrained Handwritten Texts Using HMMs and Statistical Language Models", Proc. IEEE Transactions on Transactions on Pattern Analysis and Machine Intelligence (PAMI), Vol. 26, N°6, pp. 709–720, 2004.

[10] R. Wagner, M. Fisher, "The string-to-string correction problem", Journal of the ACM, Vol. 21, N°1, pp. 168–173, 1974.