Internal Use Only（非公開）

TR－SLT－0018

Studies on word verification and on word correction using language information, time information, and N-best lists
言語・時間情報による単語照合・訂正に関する研究

Marion Dohen
マリオン ドーエン

August 30, 2002

This document will present the research work I conducted in ATR for five months from april to august 2002. It will focus on two main directions: word verification and word correction. The results which will be presented were achieved using the SPINE2 database. My approach to both problems is mainly based on the use of N-best lists outputted by the recognizer. Concerning word verification, I investigated several already known word scoring methods and I set a threshold on each one with the equal error rate method. As for word correction, I combined N-best lists with language information and other types of useful information. I then created a list of candidates among which a choice was made using several scoring methods.

（株）国際電気通信基礎技術研究所
音声言語コミュニケーション研究所
〒619－0288「けいはんな学研都市」光台二丁目２番地２ TEL：0774－95－1301

Advanced Telecommunication Research Institute International
Spoken Language Translation Research Laboratories
2-2-2 Hikaridai "Keihanna Science City" 619-0288,Japan
Telephone:+81-774-95-1301
Fax :+81-774-95-1308

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

AM: **A**coustic **M**odel.

ASJ: **A**coustic **S**ociety of **J**apan.

ASR: **A**utomatic **S**peech **R**ecognition.

ATR: **A**dvanced **T**elecommunications **R**esearch Institute International.

CLIPS: **C**ommunication **L**angagière et **I**nteraction **P**ersonne-**S**ystème (communication through language and human-machine interaction): French laboratory based in Grenoble and working in collaboration with ATR for the "C-Star project phase III" on the research topic of automatic speech translation.

CM: **C**onfidence **M**easure.

CMU: **C**arnegie **M**ellon **U**niversity (USA).

CNRS: **C**entre **N**ational pour la **R**echerche **S**cientifique (French national center for scientific research): France's biggest public institution for research.

ENSERG: **E**cole **N**ationale **S**upèrieure d'**E**léctronique et de **R**adioéléctricité de **G**renoble (National Engineering School of Electronics and Radioelectricity of Grenoble): this is the school in which I followed my studies for three years in order to become engineers.

HMM: **H**idden **M**arkov **M**odel: one of the main techniques used in ASR today.

HTK: **H**idden Markov model **T**ool**K**it: software developed by Cambridge University in order to build and manipulate HMMs for speech recognition procedures (see Appendix 1).

ICASSP: **I**nternational **C**onference on **A**coustics, **S**peech and **S**ignal **P**rocessing: this is a major conference on speech processing. It is held every year and groups researchers from all over the world.

ICSLP: **I**nternational **C**onference on **S**poken **L**anguage **P**rocessing: it is another major speech processing conference where many researchers present their state of the art achievements and work.

LM: **L**anguage **M**odel.

MFCC: **M**el-scale **F**requency **C**epstrum **C**oefficients: coefficients used in speech feature extraction (see Appendix 2).

NRL: **N**aval **R**esearch **L**aboratories: United States of America military defense research projects.

NIST: **N**ational **I**nstitute of **S**tandards and **T**echnology: American institution for standards and technologies.

R&D: **R**esearch and **D**evelopment.

ROVER: **R**ecognizer **O**utput **V**oting **E**rror **R**eduction: this system was developed by NIST and is used for re-estimation of the results of an automatic speech recognizer through the construction of word networks and multiple extraction of hypothesizes at a word level.

SLT: **S**poken **L**anguage **T**ranslation Research Laboratories.

SNR: **S**ignal to **N**oise **R**atio: this ratio measures the importance of the noise compared to the interesting signal.

SPINE: **SP**eech In **N**oisy **E**nvironments: speech recognition evaluation project conducted by the Naval Research Laboratories (NRL) of the United States of America. The 2nd SPINE evaluation took place in October 2001.

ACKNOWLEDGEMENT

I would like to thank **Dr. Satoshi Nakamura** (ATR-SLT dept1 head) head of SLT dept1 and **Dr. Tomoko Matsui** (ATR-SLT Senior Researcher) my supervisor, for their scientific and technical support as well as for their warm welcome in ATR and in Japan.

I would also like to thank **Dr. Jinsong Zhang** (ATR-SLT Researcher) for the building of the experimental setup and **Prof. Tatsuya Kawahara** (Kyoto University) for his helpful advice.

Thank you to **Yukiko Ishikawa** and **Makiko Tatsumi** (SHIEN group ATR) for their administrative help and everyday life support, their kindness and their friendliness.

Thank you to all the members of **ATR-SLT** for their warm welcome and especially to the members of the **Planning Section** and to **Megumi Kurita** who were always available to help.

Thank you to **Laurent Besacier** (CLIPS-France "Maître de conference") for his support from France.

Thank you to **Yves, Hélèna, Olivier, Cyril and Thomas** for making my stay in Japan so pleasant.

My research work was supported in part by a contract with the Telecommunications Advancement Organization of Japan entitled, "A Study of speech dialogue translation technology based on a large corpus".

ABSTRACT

This document will present the research work I conducted in ATR for five months from april to august 2002. It will focus on two main directions : word verification and word correction. The results which will be presented were achieved using the SPINE2 database. My approach to both problems is mainly based on the use of Nbest lists outputted by the recognizer. Concerning word verification, I investigated several already known word scoring methods and I set a threshold on each one with the equal error rate method. As for word correction, I combined N-best lists with language information and other types of useful information. I then created a list of candidates among which a choice was made using several scoring methods.

INTRODUCTION

Lately, research in the field of speech recognition has greatly evolved. Indeed, systems achieving greater and greater recognition accuracies are being developed. Some applications are starting to pop up through great public products such as cellular phones, human-computer interfaces.... Nevertheless, there remains a great problem that still requires the help of research: robustness. Actually, it is true that many systems show good recognition accuracies, nevertheless these good performances are achieved in laboratory conditions: clean data, no surrounding noise and stable environment. The problem is that for everyday applications such conditions are never gathered. Indeed, there is always some background noise (public places, people speaking behind the main speaker, reverberation of the speaker's voice...). Moreover, in spontaneous speech conditions, the speakers hesitate and may start over their sentences and use spontaneous speech expressions such as: "umh", "euh".... Of course, the system must not consider this as useful speech information.

The main issue in speech recognition today is therefore robustness. A few main guidelines direct the researchers' work all over the world and research considers many possibilities to improve robustness. Two main directions are being followed: either through recognition itself by noise adaptation and compensation or as post-processing methods focusing on a way to (re)estimate the reliability of the results by using more sophisticated language models or confidence measures.

I have only explored the second approach during my internship in ATR. This document will enable me to describe my work in ATR. I investigated word verification based word correction. The main issue was to find a post-processing method to improve the recognition accuracy of a traditional HMM automatic speech recognizer. Two problems mainly directed my studies. The first one was the estimation of the reliability of the results at a word level and the second one was the correction of the words that were considered as doubtful by the previous method. These two issues will be respectively referred to as *word verification* and *word correction.*

First, I will present the background of my research topic, then the objectives and the approaches that were fixed at the beginning of the internship. I will then describe into more details the methods I developed and some experimental results enabling me to justify the choice of such methods. At last, I will describe the overall achievements compared to the objectives fixed at the beginning and conclude on the work performed and the personal assessments.

As explained in the main introduction to this report, robustness to noise in real environment is a very important issue in research on ASR. Different approaches to the problem are tackled but the two main directions consist in either working on the recognition process directly or working on a post-processing method after recognition in order to improve the recognition accuracy. The former focuses on noise adaptation and compensation and the latter is based on (re)estimation of the reliability of the results. For the latter, many research projects have been conducted focusing either on the calculation of a confidence measure score (CM) or on the use of more sophisticated language models. Post-processing using CM has successfully been applied in [3, 4]. These methods consisted in globally maximizing the CM over the whole utterance. The problem is that CMs are often empirically defined and not regularized like probabilities. In those cases, the maximum criterion of the score product over the whole utterance is not always optimum. That is why, in my approach, I have applied such techniques locally (at a word level).

The global motivation of my project was therefore to consider the methods in a more local way. I considered each word on its own and with its surrounding context and not only the utterance as a whole. Indeed, each word can be important on its own and, in my opinion, the improvement of recognition accuracy comes through each word. What must be considered is that, on the basis of local post-processing, improvement can be performed even if the whole utterance is not entirely corrected. Moreover, it happens frequently that a single word in one sentence has a very important effect whereas others none. This is why considering each word may help for the recognition results. For example, in military environments, some words are crucial and even if the others are wrong the whole sentence can have the right meaning as long as this particular word is well recognized.

OBJECTIVE

My objective was to achieve word correction at a word level after traditional HMM recognition. This means that the methods concerning maximization of the CM were applied locally (at a word level) instead of over the whole utterance. This enabled me to get rid of the problem caused by the empirical designation of the CMs and their none regularization.

The first step was **word verification**. This consists in telling which words were probably well recognized and which ones are doubtful.

The second one was **word correction**. This is based on the previous step and consists in correcting the words that were said to be doubtful.

APPROACH

Global approach

My global approach was to study the two main issues separately and to develop several methods for each one in order to combine them afterwards in a global post-processing procedure, which I refer to as word verification based word correction.

Word verification

For word verification, my approach was quite simple. What I actually did was to use several already known CMs, combine them and compare their efficiencies. In order to do so, a threshold was set for each CM according to the EER criterion. The best CM gives the lowest EER. The aim is to make a binary decision: whether the word should, or not, be corrected. The method used for setting the threshold will be explained into more details in the word verification section.

Word correction

At first, I assumed that word verification was perfect (i.e. I knew exactly which words were well recognized and which were wrong after verification) and I tried to develop a correction procedure that improved the recognition accuracy. To do so, I first used only LM information but finding that this was not efficient enough, I tried to extract several types of information in order to combine them with the language information.

Combination

The second part of my job consisted in developing a procedure to combine the independently developed ones in order to design an entire post-processing procedure.

Moreover, I also tried to combine the techniques with the traditional rescoring methods [3, 4] in order to see if they could be of any use to improve the performances even after global 3-gram rescoring.

The first step, before developing any method, was to get some baseline recognition results to work on in order to improve them. The experimental setup I used was the one designed in [5]. I used special software designed especially for building hidden Markov model (HMM) recognizers; this software is called HTK which stands for Hidden Markov model ToolKit (for more information on this software please refer to Appendix1).

The feature extraction was made through 39 dimensional feature vectors with the 12 Mel-scale Frequency Cepstrum Coefficients (MFCC), log energy (sum of the squared time coefficients) and their 1st and 2nd order time derivatives. For more information on the MFCCs please refer to Appendix 2.

The acoustic model consisted of 15 mixtures of gaussians per state models.

The data used was the one from the SPINE2 database (Speech In Noisy Environments). The training data was collected over 20 speakers (10 males and 10 females) talking in noisy environments with varying SNRs from 5dB to 20 dB. These dialogs were recorded with 11 different types of noise. The test data I used was collected over 2 speakers (1 male and 1 female) different than the ones used for the training data with 4 types of noise: quiet, office, helo (helicopter) and Bradley (battleship like noise). The aim is to recognize continuous noisy speech with a vocabulary of 5720 words. For more information on the SPINE2 project please refer to Appendix 4.

The results are shown in *Table 1*. The "with rescoring" line refers to the case in which the 3-gram traditional rescoring method was directly applied to the recognition results. The results given are word accuracies given in %. The word accuracy corresponds to the following formula:

$$Acc = \frac{N - S - I - D}{N}$$

where Acc is the word accuracy, N the total number of words, S the number of substitution errors, I the number of insertion errors and D the number of deletion errors.

	Baseline word accuracy
Without rescoring	56.0%
With rescoring	62.1%

Table 1: Baseline recognition results.

The HTK recognizer tool used outputs N-best lists for each utterance. That is to say that, for each utterance, more than one hypothesis is outputted. The first N hypothesizes (considering their global likelihood value) are given along with the first best hypothesis. This is what is referred to by N-best list.

The main principle here was to find the most adapted CM score in order to have the best verification performance. This corresponds to the setting of a threshold on the CM that best reflects the chance of a recognized word to be well or mis-recognized. The aim is to set a threshold for which there are almost no words which are rejected whereas they were well-recognized and, on the other hand, almost no words which are accepted whereas they were mis-recognized. As I said before, I tried several CM.

CMs used

I mainly compared four types of CMs:

- 1st best
- phone-loop
- posterior probability
- 1st best/phone-loop

The **1st best** score corresponds to the score outputted by the recognizer for each word. This score corresponds to the likelihood of the word. The acoustic scores used for the likelihood calculation are first normalized by the duration of each word.

The **phone-loop** score is calculated for each word of each utterance. Actually, what is done is a separation of the global utterance MFCC file into several MFCC files, one for each word. The time information (start and end time stamps) outputted by the recognizer for each mis-recognized word is used in order to know where each word started and ended. The MFCC files can then be separated into MFCC word level files. These MFCC files are then input in a phone loop model recognizer consisting of all phones models. The recognizer outputs the scores for each recognized phoneme in each word and to get the global phone loop score of a word, the phoneme scores are simply added to each other. These acoustic scores for each phoneme are normalized by the duration of the phoneme.

The **posterior probability** score was calculated using an already made tool from ATR-SLT. This tool is described in [3].

The **1st best/phone loop** score corresponds to a ratio of the 1st best and phone loop scores both normalized by the duration of the word.

EER threshold setting

The goal here was to set a threshold on the CM that would determine whether the word considered was right or doubtful. This was done using the EER method (Equal Error Rate method) and a tool that was designed by other ATR members.

The EER method consists in setting a threshold that corresponds to equal false acceptance and rejection rates. Indeed, when setting a threshold, some words that have close scores to the threshold might be falsely accepted (they are said to be well-recognized whereas they are not) or falsely rejected (they are said to be doubtful whereas they have been well-recognized). A good CM score is a score for which the false acceptance and rejection rates are the lowest possible. The EER threshold setting method is illustrated in *Figure 1*.

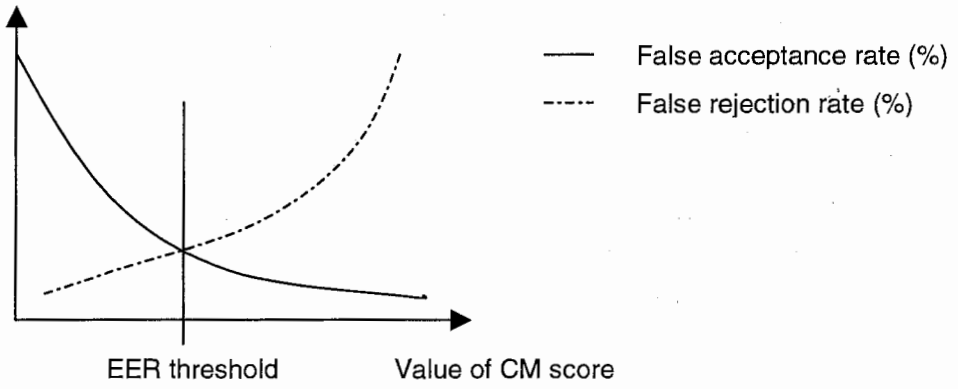

Figure 1: EER threshold setting method.

The principle I used in my approach to word correction was the following. At first, I had the idea to use 3-gram language information extracted from the 3-gram language model. This idea first came from the works described in [3, 4]. Nevertheless, as I said in the introduction section, these methods can show limits because they are applied over the whole utterance. The difference in my first idea was that I would apply this language information at a word level and not over the whole utterance. The results were quite poor because the language information is actually not enough. So, in order to improve the correction method, I decided to combine the language information with other types of useful information extracted from the N-best list of hypothesizes outputted by the baseline recognizer. The aim of such a method was to create a restricted list of the most probable candidates. The next main step was to manage to score the candidates in a way which best reflected their correctness (or not) in order to select the best one for correction.

The main steps of the correction method are shown on *Figure 2.*

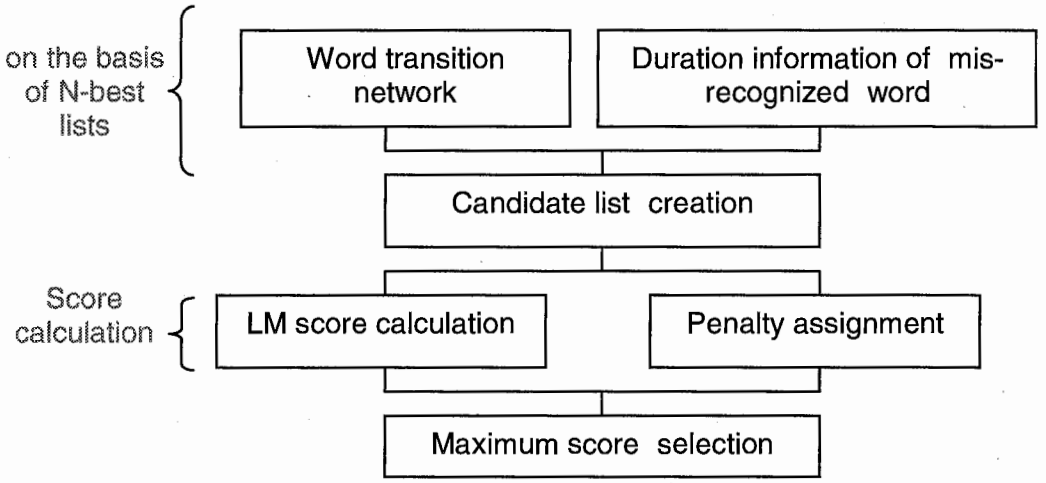

Figure 2: Global description of the word correction procedure.

As a global description of the procedure, I can say that, first of all, two steps are performed simultaneously and both consist in extracting some type of information from the N-best lists outputted by the baseline recognizer: word transition networks and duration information. These features are combined to create a list of candidates. Several scoring methods can then be used to give a score to each candidate of the list and after that a selection can be made among the candidates using the scoring features.

Word transition networks

The first features extracted from the N-best lists outputted by the recognizer are word transition networks. These are important for correction because they consider multiple hypothesizes at a word level whereas the output of the recognizer is a list of hypothesizes at an utterance level. My correction approach considers words on their own but inside a context, which is the utterance. Therefore, such word transition networks are interesting because they enabled me not only to get information on the

different alternatives for each word instead of only for each utterance, but they also enabled me to place these words in a context.

The tool used for the building of these word transition networks was the ROVER alignment tool developed by NIST. It is described in [6]. This tool takes N hypothesizes given from the recognition of a given utterance and combines them in order to get a word network. It also outputs time information (start and end time stamps) for each word of the network. If a given word appears several times in different hypothesizes at the same location the start and end times are averaged over all the appearances.

Duration information

The duration information corresponds to the duration of the mis-recognized words. Actually, the baseline recognizer does not only output the N-best lists but also outputs the time information for each recognized word: the start and end time stamps. This enabled me to define a more specific information pattern. Indeed, instead of just having the information: "the 3rd word is doubtful and should be corrected", I get the information: "all that is between t_{start} and t_{end} is doubtful and should be corrected". That is to say that, when there is a sequence of doubtful words following each other in an utterance as in *Figure 3*, they are grouped together as one case of correction. In *Figure 3*, we can see the information which is kept: start time of the first doubtful word: t_{start}, the end time of the last doubtful word: t_{end} and the duration of the whole doubtful sequence: $t_{dur} = t_{end} - t_{start}$.

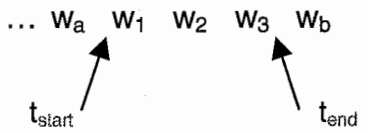

Figure 3: The duration extraction process.

Candidate list creation

The word transition networks and the duration information are combined to create a list of candidates for correction. Actually, the word transition network paths that fit the following criteria are selected for the list: their overall duration must not exceed the duration of the mis-recognized word or sequence of words (t_{dur}), the start time of the first word of the paths must not be smaller than the start time of the mis-recognized word or group of words (t_{start}) and its end time must not be bigger than that of the mis-recognized word or group of words (t_{end}). The advantage of this method is that there is no limitation in the number of words of a candidate as long as it fits the requirements. Moreover, doing so, this method enables me to correct not only substitution errors but also insertion and sometimes deletion errors. This advantage may help me to improve the recognition accuracy a little more.

LM score calculation

This step consists in calculating the LM score of each candidate taking into account its context inside the utterance. The i^{th} candidate is represented as follows:

$$C_i = C_{i1}C_{i2}\ldots C_{iNi}$$

where N_i is the number of words of the candidate and the c_{ij}, j=1..N_i, are the N_i words of the candidate c_i. The LM score corresponds to the following formula:

$$S_{LM}(c_i) = P(w_a, c_{i1}, \ldots c_{iNi}, w_b) = P(w_a)P(c_{i1}|w_a)P(c_{i2}|w_a, c_{i1}) \ldots P(c_{iNi}|c_{iNi-2}, c_{iNi-1})$$

This is actually the probability calculated with 3-gram LMs in consideration of the left and right context information.

"LMonly" method

This method consists in choosing the candidate with the highest LM score for correction.

Penalty assignment

In the previously explained "LMonly" method, the candidate with the highest LM is chosen for correction. Nevertheless, the main problem is that longer word sequences tend to have lower LM scores and such a method might then favor the candidates with shorter word sequences. Therefore, I considered the assignment of a word penalty for each candidate. This word penalty actually reflects the fact that longer word sequences have lower LM scores. To compensate this, I add a word penalty to the previously calculated LM score. This penalty depends on the number of words and grows as the number of words grows. In this manner, long word sequences will be attributed a large word penalty whereas sequences with very fewer words will have a small word penalty. The new score is calculated as follows:

$$S_{LM+WP} = S_{LMonly} + \lambda_i$$

where **i** is the number of words of the candidate and S_{LMonly} is the score of the "LMonly" method. The word penalty λ_i was set experimentally to:

$$\lambda_1 = 2.2 \quad \lambda_2 = 2.9 \quad \lambda_3 = 4.6 \quad \lambda_4 = 5.6 \quad \text{and for } \mathbf{i>4} \quad \lambda_i = 9.0.$$

"LM+WP" method

This method consists in choosing the candidate with the highest S_{LM+WP} for correction.

Various experiments were made to test the efficiency of our verification and correction methods. Nevertheless, the first thing to do was to design a baseline recognizer in order to have a baseline recognition accuracy to be compared to our results. That is what was already explained before. Then, separate experiments were performed in order to test separately the verification and the correction methods. The last experiments combine the two methods in order to test the efficiency of a global post recognition procedure.

Word verification

The results for the different scoring methods are shown in *Table 2.* These were achieved before any 3-gram rescoring was performed.

	1st best	Phone loop	1st best/phone loop	Posterior probability
EER (%)	49.1	64.4	35.4	59.3

Table 2: Results for word verification experiments.

As we can see these results are not so good. Actually several reasons can be put forward for this. The first one is of course the size of the vocabulary (more than 5000 words). With such a big number of words the variances over all the words are always very high. The phone loop score gives a bad result on its own and this was predictable since the acoustic features were extracted according to word considerations and not phoneme ones. Nevertheless, this score is useful to normalize the 1st best score. We can also see that we get better results for the 1st best, and 1st best/phone loop scores. The posterior probability score does not give any better results than what we would get by qualifying the words as right or doubtful by chance. This could be considered as quite strange since this scoring method is supposed to be a good measurement for confidence. This may be due to the fact that all the other scores where normalized by the duration of the words whereas posterior probability scores could not. Actually, there is some normalization effect when calculating posterior probability itself. Nevertheless, it is calculated as a ratio of the word and the whole word graph probabilities. When taking the ratio, the normalization effect is canceled because the length of the word is common. The tool used for posterior probability calculation may then not be so effective but unfortunately there was not enough time for me to explore the possibility of adapting it to this particular case.

Nevertheless, even though the 1st best and phone loop could be normalized they do not give very good results. An explanation for this may be that the threshold calculation algorithm used is based on an estimation of the curves of false acceptance and false rejection to estimate the EER and therefore the threshold. In this case the data is very spread and because of this, the estimation of the curves cannot be very good and the threshold setting is only approximate. Indeed, if we calculate the false acceptance (FA) and rejection (FR) rates obtained with the threshold giving the 35.4% EER we find FA=22.2% and FR=51.2% which is very bad. That is why I tried to adapt the threshold by hand in order to get closer FA and FR.

The best I could get was FA=29.2% and FR=44.0% with the 1st best/phone loop scoring method.

When 3-gram rescoring was performed after recognition and before verification, I obtained even worse EER (bigger ones): 36.5% for the 1st best/phone loop score, which gave the best results and by calculation of the FA and FR I found FA=18.5% and FR=59.6%. After adjusting the threshold by hand, I managed to achieve FA=27.2% and FR=55.9% but this is still very bad.

Word correction

For word correction, I used only 10-best lists. Indeed, experiments with a higher number of candidates were conducted but barely gave any improvement for a much longer computation time. The LMs I used are 2-gram and 3-gram LMs designed by CMU for the SPINE 2 database.

For the first experiments, corresponding to the results listed in *Table 3*, I considered that word verification was perfect (0% EER) that is to say that I knew exactly which words were wrong and which ones were right after the recognition procedure. *Table 3* lists the word accuracies for the "LMonly" and "LM+WP" methods with and without 3-gram rescoring.

	baseline	LMonly	LM+WP
Without rescoring	56.0%	62.9%	63.9%
With rescoring	62.1%	66.2%	66.2%

Table 3: Word accuracies after using the correction methods with perfect verification.

These results indicate that the correction methods can be effective with and without rescoring. The combination of rescoring and correction gives an improvement of more than 10% in word accuracy (from 56% to 66.2%). The performances of the "LMonly" method without rescoring and the baseline method with rescoring are almost the same (respectively 62.9% and 62.1%). It can therefore be assumed that 3-gram LMs that are first applied globally or locally as post-processing have a similar effect. With rescoring, the "LMonly" method outperformed the baseline method by 4.1% owing to the additional use of 3-gram LMs applied locally (after global application). The word penalty shows slight improvement when it is applied without rescoring but none with it.

In *Table 4*, we can see more detailed results. H stands for hits (well recognized words), D for deletion errors, S for substitution errors, I for insertion errors, N for the total number of words and Acc for the word accuracy.

		H	D	S	I	N	Acc(%)
baseline		739	141	319	68	1199	56.0
Without rescoring	LMonly	770	272	157	16		62.9
	LM+WP	794	223	182	28		63.9
With rescoring	LMonly	804	251	144	10		66.2
	LM+WP	809	222	168	17		66.2

Table 4: Detailed results of word correction with perfect verification.

We can see that, globally after word correction, there are more deletion errors but less substitution and insertion errors. There are more deletion errors because shorter word sequences, even with the word penalty, often replace long ones. This is because I have only set word penalties for the small number of words since the sequences with large numbers of words are rare and setting them by hand for all the possibilities would have been too long. Therefore, even if the long word sequences are rare, they cause this rise in the number of deletion errors. There are fewer substitution errors because of the local use of LMs for correction and fewer insertion errors exactly for the same reason as the rise in the number of deletion errors. We can also see that the "LM+WP" method gives more hits and stands for less deletion errors, more substitution errors and more insertion errors. Actually, this can be easily explained from the fact that when there is no word penalty the candidates with shorter word sequences are favored so more deletion errors are committed (since fewer words may replace a longer sequence), the same thing accounts for the diminishing of the insertion errors with the word penalty.

Combination of verification and correction

This section presents the results obtained from combination of the verification and the correction methods. First, the verification algorithm was applied directly to the recognition results either rescored or not. This algorithm gives each recognized word a qualifier (either probably right or probably wrong): this is the binary decision. Then I applied the correction algorithm taking into account the result of the binary decision process. That is to say that, every word that has been qualified as doubtful is corrected.

The results presented in *Table 5* were obtained considering the threshold set by hand that gave the closest FA and FR. They are the word accuracies obtained after verification and correction (except for the baseline of course). This was done for both methods: with and without 3-gram rescoring.

	baseline	LMonly	LM+WP
Without rescoring	56.0%	55.5%	56.2%
With rescoring	62.1%	58.6%	62.1%

Table 5: Word verification based word correction: results.

The bad results of word verification procedures did not enable me to estimate the efficiency of the correction method. A major priority would be to improve the performances of word verification, which unfortunately I did not have time to do. Nevertheless, in order to estimate the goal that should be achieved in word verification for the correction method to show efficiency, I simulated word verification.

The simulation principle I used is very basic. I actually designed an algorithm that, with a specified value of EER (the desired EER value input by the user), falsely rejected (or accepted) some words he knew were actually right (or wrong). The graph shown in *Figure 4* gives the improvement in word accuracy after verification and correction compared to the baseline word accuracy versus the EER. The improvement is calculated on the basis of the baseline determined with 3-gram rescoring (62.1%).

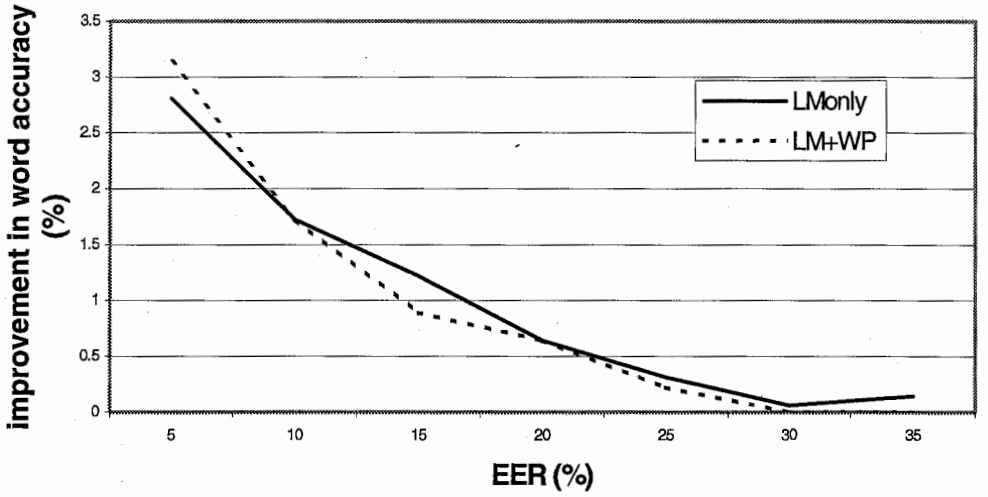

Figure 4: Word correction performance under verification simulation.

This graph enables me to say that if an EER of 20% could be achieved, our correction method would already show quite some improvement as far as word accuracy is concerned.

In this section, I will stress the points that could be improved if further work was done on this topic.

Of course, as I said before, word verification has to be improved for the global post processing method to be really effective. 20% EER should at least be achieved. This could be done through the use of the 2nd best scores (which I started doing at the end of my internship) and the next scores. The following formulas could be applied:

$$S_1 = \frac{P_{1stbest}}{P_{2ndbest}}$$

$$S_2 = \lambda_1 \frac{P_{1stbest}}{P_{1phloop}} + \lambda_2 \frac{P_{2ndbest}}{P_{2phloop}} + \ldots + \lambda_n \frac{P_{nthbest}}{P_{nphloop}}$$

Moreover, word correction could also be slightly improved in my sense. Indeed, the duration of the words inside a candidate could also be taken into account for setting another type of penalty. This would compensate the fact that longer words tend to have lower LM scores. Another thing to do could be to allow an overlap time in the criterions for the creation of the candidate list. Indeed, here I have limited the starting time of the possible path to be greater or equal to t_{start} but I could have allowed a small overlap in order to be more tolerant in the acceptations of paths from the word transition network to become candidates for correction. The same tolerance could be applied for the ending time of the paths.

Many other ideas could be applied and this topic really needs further work.

GLOBAL CONCLUSION: OVERALL REVIEW

As a global conclusion, I will say that the work achieved was quite useful since the correction methods designed showed some good improvement. Moreover, as I wrote at the beginning, the main idea was to apply language information, which had already been done in a local way (at a word level) but not in a global way (over the whole utterance). The experiments showed that this could be done successfully as a combination. That is to say that after recognition language information can be used in a global way to make overall corrections and then they can be used in a local way to make more precise corrections. Doing so, the word accuracies could be improved by more than 10%. This challenging research topic was therefore really rewarding since it managed to show improvement. Moreover, a patent will be deposited for the correction method.

This internship was my first experience in the field of research. It enabled me to understand the use of some of the things I had learned in my engineering school. It was a very pleasant thing for me to work in ATR and many people helped me during my stay. I saw how researchers from all over the world collaborated and I learned that research had no boundaries. I hope I can come again to Japan and to ATR in the future.

PUBLICATIONS

[1] Wessel (F.), Schluter (R.) and Ney (H.)
Using posterior probabilities for improved speech recognition
Proc. of ICASSP 2000, pp. 563-566, Istanbul (Turquey).

[2] Everman (G.) and Woodland (P. C.)
Large vocabulary decoding and confidence estimation using word posterior probabilities
Proc. of ICASSP 2000, pp. 2366-2369, Istanbul (Turquey).

[3] Zhang (J.) and al.
Developping robust baseline acoustic models for noisy speech recognition in SPINE2 project
Proc. of ASJ spring conference 2002, pp. 65-66, Japan.

[4] Fiscus (J. G.)
A Post-processing method to yield reduced word error rates: recognizer output voting error reduction (ROVER)

[5] Matsui (T.), Soong (F. K.) and Juang (B. –H.)
Classifier design for verification of multi-class recognition decision
Proc. of ICASSP 2002, vol. pp. 117-120, Orlando (USA).

[6] Singh (R.) and al.
Speech in noisy environments: robust automatic segmentation, feature extraction, and hypothesis combination
Proc. of ICASSP 2001, pp. 273-276, Salt Lake City (USA).

[7] Okimoto (Y.) and al.
Evaluation of mis-recognition detection using confidence measures
Proc. of ICSLP 2001, Daejon (Korea).

BOOKS

[8] Rabiner (L.) and Juang (B. –H.)
Fundamentals of speech recognition
Prentice Hall International Editions, 1993
Chap. 6: Theory and implementation of hidden Markov models (pp. 321-389).

[9] Young (S.) and al.
The HTK Book
Revised First Edition, Dec. 2001.

[10] Wall (L.), Christiansen (T.) and Schwartz (R. L.)
Programming Perl
O'Reilly Editions, Second Edition, Sept. 1996.

[11] Fukunaga (K.)
Statistical pattern recognition
Academic Press, second edition
Chap. 4: Parametric Classifiers (pp. 124-180).

WEB PAGES

SPINE2 information:

[12] http://elazar.itd.nrl.navy.mil/spine/sri2/sri1.html
[13] http://elazar.itd.nrl.navy.mil/spine/ibm2/ibm1.html

Information for the calculation of the overall cost of the project:

[14] http://www.sg.cnrs.fr/espaces/personnels.htm
[15] http://www.sg.cnrs.fr/drhstatus/remunerations/grille.pdf

APPENDIX

APPENDIX 1: HTK Hidden Markov model ToolKit

The information given here was extracted from the following web page: http://htk.eng.cam.ac.uk/.

This software was originally designed by the Speech vision and Robotics Group of the Cambridge University Enginneering department (CUED) in 1989 where it was used to build CUED's large vocabulary speech recognition systems. Then, the software was developed by Entropic Research Laboratory Inc. from 1993 and was licensed by Microsoft in 1999.

HTK is a portable toolkit for building and manipulating hidden Markov models. It consists of a set of library modules and tools available in C source form. The tools provide sophisticated facilities for speech analysis, HMM training, testing and result analysis. The software supports HMMs using both continuous density mixture gaussians and discrete distributions and can be used to build complex HMM systems. The HTK release contains extensive documentation and examples.

The HTK book [12] is very helpful to learn how to use HTK, it also gives information on how the software actually works and on the theoretical background concerning HMMs.

APPENDIX 2:MEL-Frequency Cepstral Coefficients (MFCC) [11, 12]

The MFCCs are one kind of parameters used in ASR to describe the speech signal. In order to calculate these coefficients, the procedure shown in *Figure 5* is applied to the speech signal.

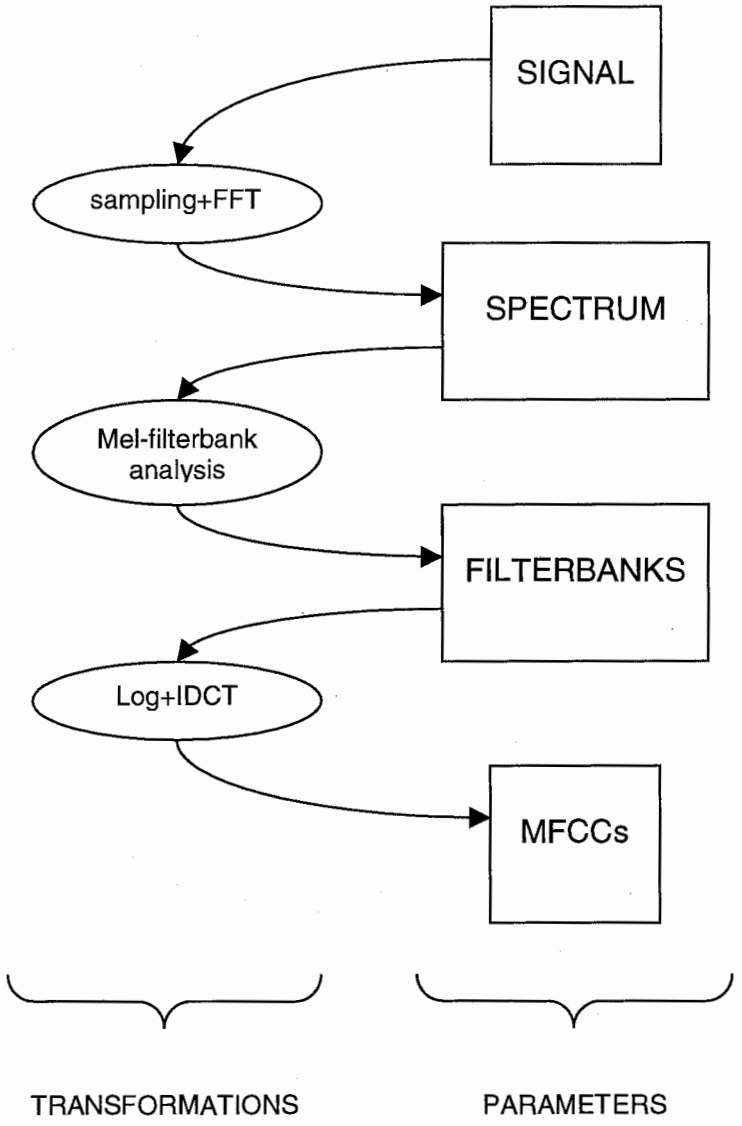

Figure 5: Procedure applied to the speech signal to calculate the MFCCs.

Some of the transformations, such as FFT and sampling, are quite traditional in signal processing but the Mel-filterbank analysis and the role of the log+inverse discrete cosine transform are more specific to speech processing. Therefore, their role will be detailed a little more in the next section.

Cepstrum analysis

A very widely spread model for human speech production is that the speech is the result of the filtering of a periodic signal, produced by the glottis, by the vocal tract (see *Figure 6*), the signals on *Figure 6* are shown in the frequency domain.

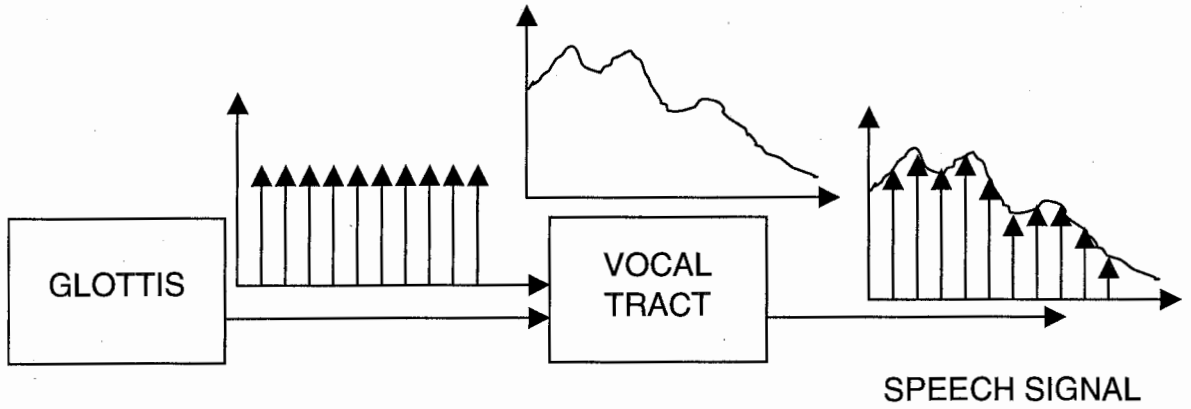

Figure 6: The speech production process.

In ASR, the information from the vocal tract is more useful than the information from the glottis. Therefore, it would be interesting to separate these two parts of the signal. This is done by using the log and the IDCT transform.

The first step is to take the log of the speech spectrum in order to transform the speech signal into a sum of two terms thanks to the following equation:

$$S^l = \log(S) = \log(S_{glottis}) + \log(S_{tract})$$

The second step is to apply the IDCT to S^l in order to go back to a kind of temporal domain, which is called cepstral domain (or quefrency domain). In this domain, $S^l_{glottis}$ and S^l_{tract} are located in two different regions on the quefrency axis. The information concerning the vocal tract can very easily be selected by keeping the low quefrency coefficients. These coefficients are called cepstrum coefficients. *Figure14* shows an example of this processing.

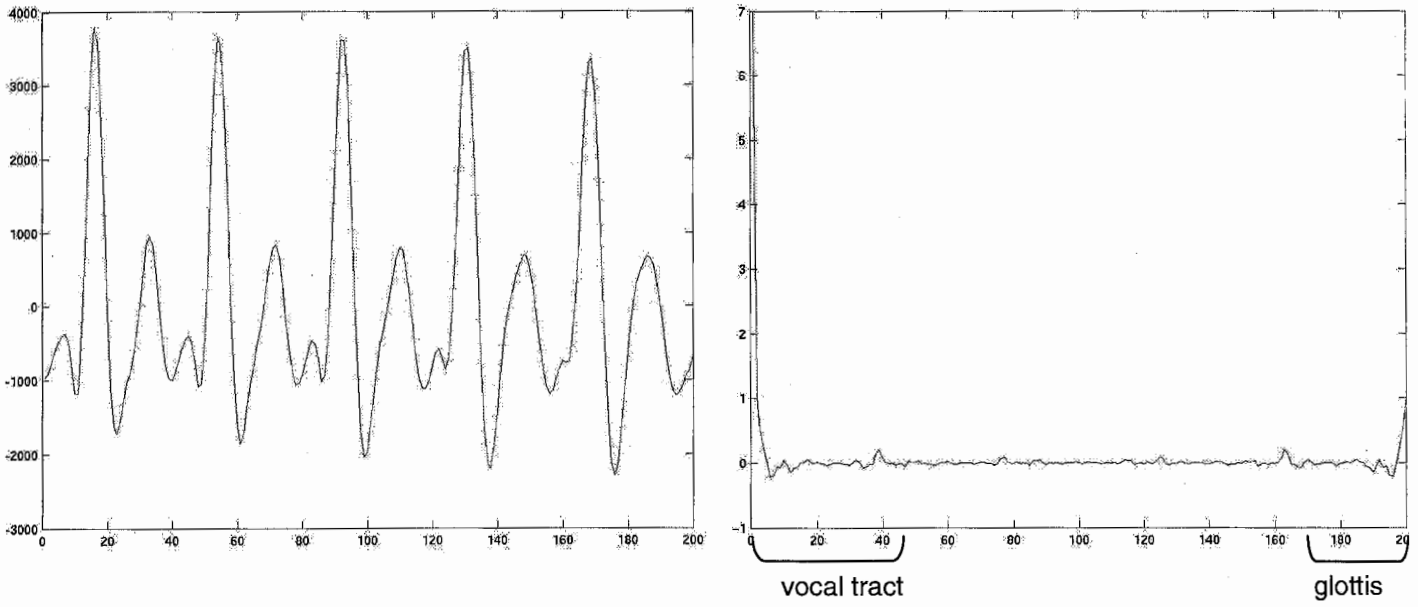

Figure 7: Extraction of the cepstrum coefficients.

Mel-filterbank analysis

The human ear resolves frequencies non-linearly across the audio spectrum and empirical evidence suggests that designing a front-end to operate in a similar non-linear manner improves recognition performance. A good way to achieve this is through the use of a filter bank analysis. The problem is to choose the spacing of filters on the frequency axis. Perceptual studies showed that one possibility is to space the filters along the critical band in order to choose bands that give equal contribution to speech articulation. The MEL scale is a variant on the critical band scale and is defined by:

$$Mel(f) = 2595 \log_{10}(1 + \frac{f}{700})$$

In order to implement this filter bank, the speech is transformed using a Fourier transform and the magnitude is extracted. The magnitude coefficients are then binned by correlating them with triangular filters equally spaced on the Mel-scale. Here binning means that each FFT magnitude coefficient is multiplied by the corresponding filter gain and the results are accumulated (see *Figure 8*).

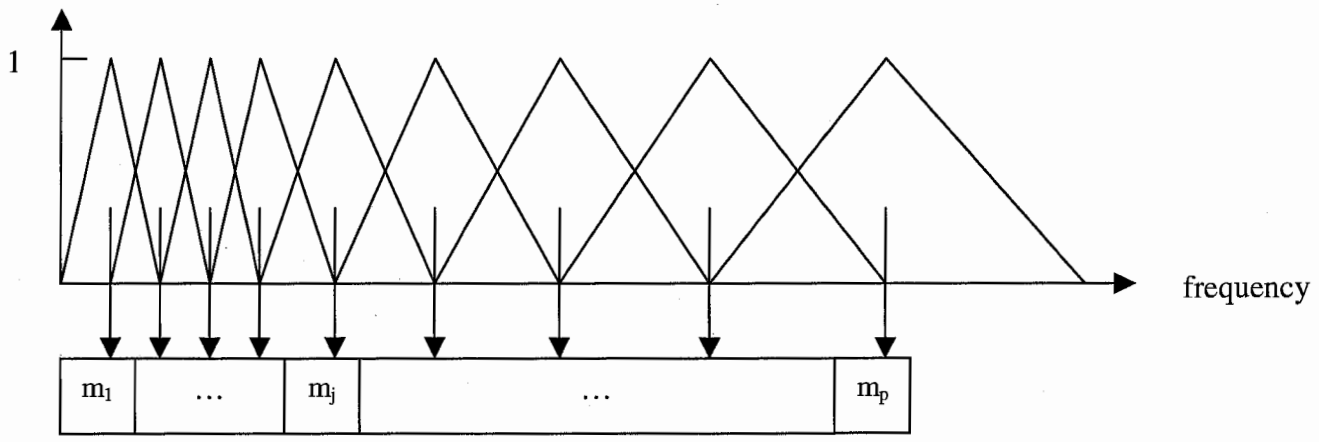

Figure 8: The Mel filter bank analysis.

APPENDIX 4: SPINE 2 project [9, 15, 16]

The 2nd SPINE (Speech in Noisy Environments) evaluation was conducted by the Naval Research Laboratories (NRL) of the United States of America in October 2001. For this evaluation, many robust automatic speech recognition systems were designed. The SPINE evaluation focuses on the task of transcribing speech produced in noisy environments with an emphasis on military environments. It concerns continuous speech recognition for a vocabulary of 5720 words. The task was consequently to transcribe speech produced in noisy environments. SPINE 2 provides a continuing forum for assessing the state of the art and practice in speech recognition technology for noisy military environments and for exchanging information on innovative speech recognition technology in the context of fully implemented systems that perform realistic tasks.

SPINE 2 data

The recorded signals consisted of a continuous background signal of noise produced by the recording equipment, with intermittent recordings of speech and reproduced military noises communicated through the channel.

Noise

In one's conversation side there may be anyone of 11 different types of noise including: quiet, office, aircraft, car, helicopter, tank, Bradley (battleship like noise) .... The level of the noise may also be varying in one utterance and from one utterance to the other.

Moreover, the free style speech also includes frequent dropouts, repairs and other spontaneous speech phenomena.

Data collection

The conversations corresponded to dialogs between talkers working on a collaborative battleship like task were they had to seek and shoot at targets (ARCON Communicability Exercise, ACE). Recordings were made through regular channels and through various vocoders. The speakers could talk freely (spontaneous speech) but the total vocabulary used was fairly limited. Each speaker was seated in a sound chamber were a previously recorded military noise background was accurately reproduced. The recordings were made through a microphone (headset). The speech data is presented as a sequence of turns where each turn is the period of time when one talker is speaking. Successive turns may be from the same speaker.

Training data

There are about 28 000 utterances in 324 dialogs by 20 speakers (10 males and 10 females) with varying SNRs from 5dB to 20 dB. These dialogs were recorded with 11 different types of noise.

Test data

This data was collected from 4 new speakers (2 males and 2 females). The noises environments considered were quiet, office, helo (helicopter) and Bradley (battleship like noise).

Aim of SPINE2 project

This project was mainly designed to provide researchers working on robust speech recognition and from all origins (university, industrial and commercial speech system developers) with noisy data very different from the clean data they used to use and very close to real environment conditions.