Internal Use Only
002

TR-IT-0026

Recent issues in Meaning Recognition

Didier Verna

1993.11

本論文では、特殊な形の統計的神経回路網を用いてＡＴ＆Ｔベル研究所において行われている意味認識に関する研究の要約とその評価とを行っている。また、その認識システムの改良のためのいくつかの考えも提案する。

©ＡＴＲ音声翻訳通信研究所
©ATR Interpreting Telecommunications Research Laboratories

1 Introduction

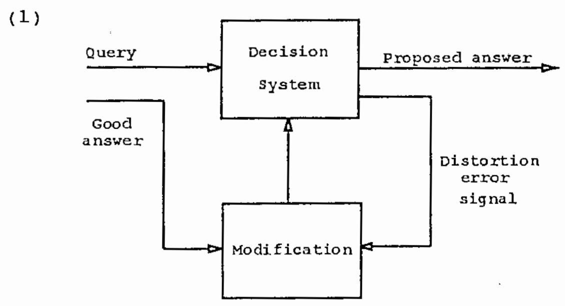
Recently, some experiments were done in a new way of understanding the concept of speech recognition. These experiments, performed at the AT&T Bell Laboratories by A.L.Gorin, L.G.Miller, S.E.Levinson and A.N.Gertner ([Go1], [Go2], [Go3], [Go4]), were based on the following main ideas: First, they focused on *meaning understanding*, rather than on any grammatical detection purpose. Their device was then supposed to respond in a particular way, according to its understanding of the meaning of the query. Secondly, the system was build in order to *learn its skills during the curse of performing its task*. That is, there was no separation between the training period and the functioning period, all tasks acomplished in a single step. Those principles conduced them to build a device based on connectionist methods, and providing a conversational-mode algorithm, highly in interaction with the speaker. We think that this new approach in speech recognition is quite challenging, and may lead to important results in the future. Consequently, we will remember at any time, the two main concepts that motivated their work. However, we feel not totaly satisfied with their system, which remains highly dedicated to its original task, and which we think is not efficiently improvable. In this paper, we would like first to give a more detailed summary of this recent work, and then, present new ideas that would permit to build a more powerful and more general system. Those ideas will be presented as five new concepts that will permit to eliminate some weaknesses of the preceeding system, and that we think could even be a new way of aproaching more general tasks in speech recognition. In order to give concrete illustrations to all the problems an proposed solutions, we will take the example of a telephonic autocommutator (example already used by A.L.Gorin) in all our considerations. The system will answer customers phone queries, and will be able to connect the speaker to some departements of a shop, such as the food department, the clothing department, or the HiFi/video department.

2 Summary of previous work

We now go on the description of the work performed at the AT&T Bell Laboratories. We would like first to present the general concepts of the system, that will still be relevant in our own work, and then describe the device itself, and the algorithm that runs it.

2.1 General concepts

As we said before, the system is based on the two following principles [Go4]:



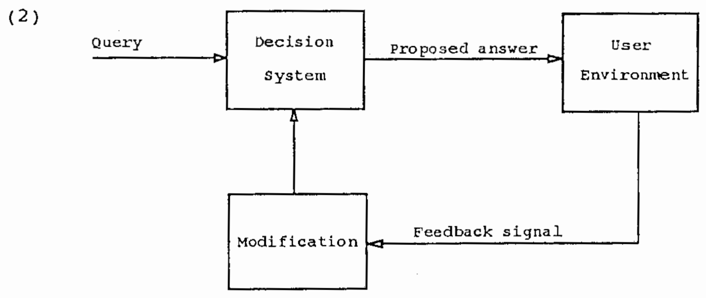


図 1:
(1) A traditionnal system with two separate periods.
(2) A system that "learns by doing".

- *"The primary function of language is to convey meaning"*, which implies that the device must learn associations between meaningful responses and input stimuli. Then, we will say that the system *understands* if it learns how to answer correctly the input query [Go3]. This underlies an investigation of connectionists methods, namely, the use of a neural network.

- *"The language is acquired by interacting with a complex environment"*, which implies the existence of a feedback signal mesuring the appropriateness of the output for the given input stimuli. This underlies a system which takes into account this signal in its learning procedure. We call that *learning by doing* instead of *learning by example.* This principles implies that the device will be ruled by an "conversational" algorithm which will ask confirmations or corrections to the speaker.

We can now compare this kind of system to the classical ones, that provide two separate periods, one for learning, and one for running. Fig 1 permits to visualise the difference.

2.2 The device

Mainly, we saw that the first principle lead us to use connectionist methods (that is, neural networks) for buildinig the system. Actually, the network used here is based on an *Information-Theoretic* approach [Go1], as described below. We first describe the basic, two layered network, and then mention the possible improvements, by adding hidden layers.

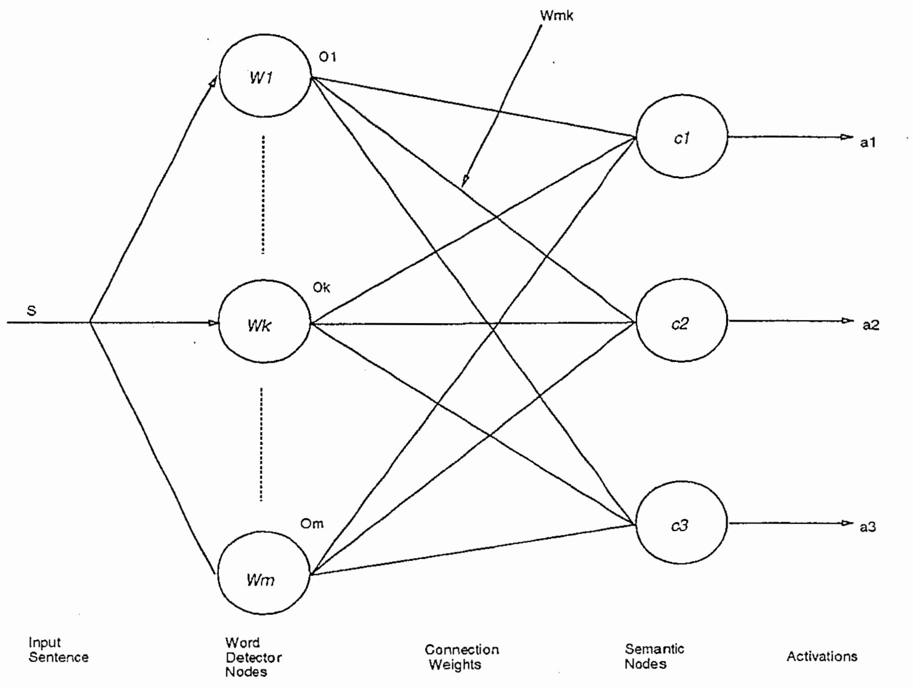


図 2: A two layered network for associating words and actions.

The input layer (See figure 2) is composed of nodes representing the words known by the system. Its size N will then increase as the number of known words increases. When a sentence is given to the system, the input nodes will produce an output O_n between 0 and 1, according to the presence of their word in the sentence. Basicaly, a node is a set of templates of the same word (different speakers, differents prononciations...), that we call M_n. For each word w in the sentence, the system computes the DTW distance [Go2] between this word and all the templates of the node n, and then, keeps the minimum. This will be written as

$$D(w, M_n) = \min_{v \in M_n} d(w, v)$$

which is obviously the "distance between the word and the node". For detecting a known word, the node which gives the smallest D will be activated, and its output will be $O_n = e^{-D^2}$. A word will be considered as *unknown* if its distance to the "closest" node (i.e. the one that would be activated) is still superior to some threshold [Go4]. In that case, a new node is added to the input layer.

The connection weights (figure 2) in this network [1] are defined as the *mutual information* [Go1] between the words and the actions. For instance, if we considere the input node n and the output node k, the connection weight w_{nk} will be defined as $w_{nk} = I(c_k, w_n) =$

[1]Here is the "Information-Theoretic" approach.

$log(P(c_k|w_n)/P(c_k))$ where $P(c_k)$ is the probability of the action k, and $P(c_k|w_n)$ is the probability of the action k, knowing that the word n appeared in the sentence.

The output layer (figure 2) is composed of nodes representing the possible actions the system can perform. (In our example, we will have a node representing the action "Connect to the food departmtent", etc.). Each output node is connected to the whole input layer, and adds a bias $w_k = log(P(c_k))$ to its output. Thus, for the k^{th} action node, we define its output as

$$a_k = \sum_{m=1}^{M} (O_m w_{mk}) + w_k$$

which will be called its "activation". When the system is given a sentence, it computes the activations of all the output nodes, and the action corresponding to the biggest activation is then proposed to the speaker. It can be shown [Gol] that under certain restrictions, this network performs a *maximum a posteriori decision rule.* Here is a proposition that details this property:

*Proposition:*If the words in the sentence $s =< w_1, w_2, ..., w_m >$ are both independant and semantic-conditionnally independant, and if the outputs O_m of the first layer are restricted to 0 or 1 (which is the case for written input), then the activations $a_k = P(c_k|s)$.

Obviously, no natural language satisfies these restrictions, but the results are still satisfying, and such a process is actually similar to grammar detection based on unigram models. A more reliable system can be obtained by adding a hidden layer to the network [Gol]. This layer will be a rudimentary grammar understanding system, dealing with adjacent word pairs instead of single words (figure 3), that is, such a node $w_m w_n$ will detect that the word w_m is present in the sentence, and immediately followed by the word w_n. In a similar way, we define the connection weight between such a node $w_m w_n$ and an output node c_k by $w_{mnk} = I(c_k, w_m w_n) - I(c_k, w_m) - I(c_k, w_n)$. The activation of the output node a_k will then become as follows:

$$a_k = w_k + \sum_{m=1}^{M} (O_m w_{mk}) + \sum_{m=1}^{M} \sum_{n=1}^{M} (O_{mn} w_{mnk})$$

One can in a similar way implement other hidden layers, dealing with n-uples of adjacent words, or even with non adjacent words.

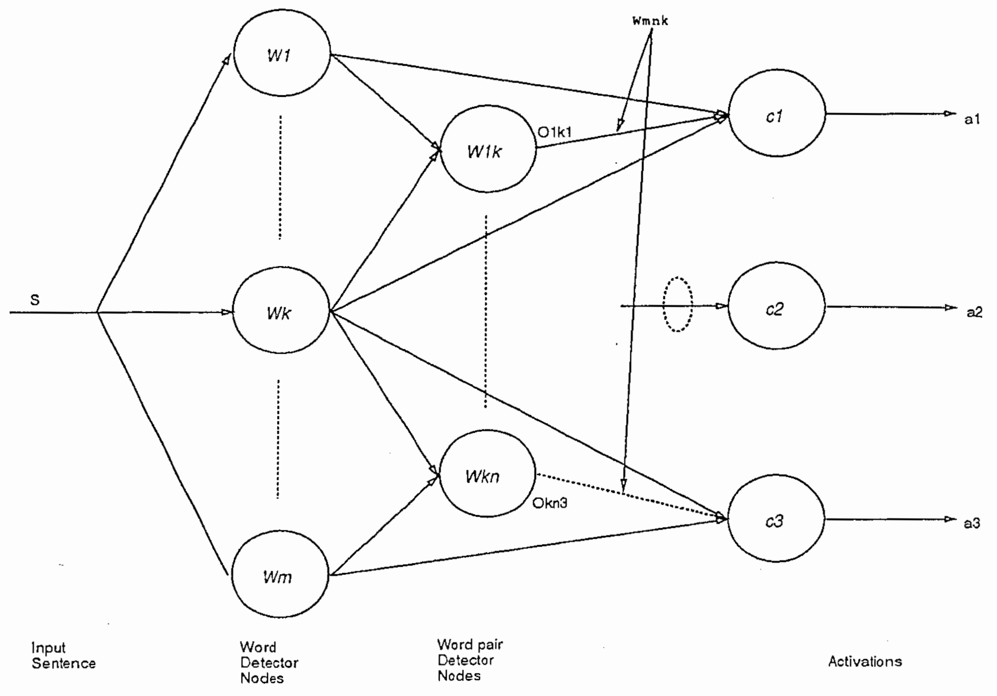


図 3: Adding hidden layers to the network.

Now that the outlines of the device are drawn, we can study how it works. We will now describe the algorithm that rules the system, and some main aspects of the computation.

2.3 Algorithm and Computation

As we said before, the system must be highly in interaction with the speaker, in order to integrate the "feedback signal" in its learning procedure. We said too that since the speaker is supposed [2] to require an action, *understanding* means *learning to respond correctly to the query*. Consequently, the feedback signal must give information to the system about the appropriateness of its response. Namely, the speaker will be asked if the proposed action is the good one, and while he is not satisfied, the network will compute his answers, and propose the most probable action among those that are still possible. Here is a more formal description of the algorithm:
At each step l of the conversation, we will call s_l the sentence given by the speaker, and $a(s_l)$ the array composed of the output of the network, when acrtivated by the sentence s_l. We then define a *total activation array* A_l at each step of the conversation:

$$A_1 = a(s_1)$$
$$A_l = (l - \alpha)A_{l-1} + \alpha a(s_l) + \varepsilon_l$$

where α is a gain parameter ($1/l$ for instance) and ε_l an array containing $-\infty$ at the position of the previous action, and 0 elsewhere. Since the dialogue keeps on going at the step l, the preceding proposed action was wrong, and thus ε_l permits to eliminate it for ever from the left possible actions. The conversation will then go on until the good action is proposed, or until no action satisfies the speaker, in which case he would be connected to some human being...

[2]That's the whole point !!

Now we would like to illustrate this algorithm with two examples of dialogue, one that gives immediately the good answer, and one that needs two steps. This will permit to introduce rapidly some of the computation problems. At each step, the speaker will be asked to begin his answer with "No" if he is not satisfied. We use `S` to adress the speaker, and `D` to adress the device.

- EXAMPLE 1:

  S: Hello ! I'd like to buy a red shirt with green circles on it...

  D: Shall I connect you to the clothing departement ?

  S: OK, that sounds fine.

  Assuming that all the words are known, the network computes its output $a(s_1)$, and proposes the most probable action. After getting the confirmation from the speaker, the network is able to increase the probabilities $P(c_k|w_m)$ and $P(c_k)$ for the k^{th} action (which is the good one), and for each word w_m encountered in the sentence. In other words, the system updates the concerned weights and biases. We decide that there is no point in computing the last sentence, since we usually face sentences like "OK, good.", which are meaningless for the system.

- EXAMPLE 2:

  S: Hello ! I'd like to buy a red shirt with green circles on it...

  D: Shall I connect you to the food departement ?

  S: No, a shirt is something that you wear.

  D: Shall I connect you to the clothing departement ?

  S: Yes, that sounds better.

  Assuming again that all the words are known in this conversation, the network first computes $a(s_1)$. Since the speaker is not satisfied, the network computes the activation $a(s_2)$, and the total activation array A_2 (in which the activation for the action "Connect to the food departement" is $-\infty$). The most probable action is then proposed, it is the good one. Now the system knows the good answer to the query, and is able to update its parameters for *all the words in the conversation* (not only in the last sentence).

If some words are unknown to the system, it simply computes its activations without considering these words. At the end, when the good action is encountered, it will be able to integrate the words in the network, and update its parameters related the good action. But this underlies the problem of initialization, both of the network, and of the new words. This point will be discussed in the next paragraph. Consider now the second example, and assume that the word "shirt" is unknown. If the system knows what "wear" means, it is clear that this supply of information [3] will help the network finding the good action rather quickly. This is how the system "learns" new words.

[3] The speaker has to be REALLY kind...

So far, we talked about *updating probabilities*, *initializing new words*, but what does that mean ? Obviously, the system never knows the exact prbabilities that it uses, and thus computes *estimates* of its parameters. As an example, take the case of $P(c_k)$. A primary estimate of this probability is $N(c_k)/N_T$ where $N(c_k)$ is the total number of times the action k was the good one, and N_T is the total number of observed sentences. More clever (non linear) estimates can then be built, by adding *thresholds* and *prior beliefs* to the preceding estimate [Gol]. Thresholds are used to switch back an estimation to the prior belief, if the difference between the estimation and the prior belief is not so important. Simultaneously, new estimates are built by using barycenters between prior beliefs and preceeding estimates, decreasing little by little the importance of prior belief. As far as initialization is concerned, the actions will first be considered as equiprobable, and the unknown words will first be considered as meaningless (which means that all the weights will be originally null).

We have described here the main aspects of the system developped at the AT&T Bell Laboratories. As a conclusion, we will briefly review the positive aspects of such a system:

- First, this system is extremely easy to implement, since its main aspects are all meaningfull for the conciever. There is no abstract concepts or variables, no non-terminal symbols that would be difficult to understand and thus to program. All the connection weights, for instance, refere to some concrete probability which is easily understandable.
- Secondly, the statistical approach of the connectionists methods makes the system really efficient on the field of computation: No gradient propagation is needed for updating the network, only one pass through the variables is necessary. The updating procedure is reduced to increment variables such as $N(c_k)$ or $N(c_k, w_m)$ (see the paragraph above), and then, compute again the probability estimates concerned with the changes. This makes the system very quick.
- Thirdly, the use of a "conversational-mode" algorithm seems to be the major interest of such a system. This leads to absolutely unsupervised learning. More exactly, each user supervises "unconsciously" the learning of the field he is interested in. At the begining, the network knows only the two words "no", and "ok". But it is immediately put on service, and it is up to the user to train the system *while it is running*.

Nevertheless, one can find important inconvenients in this device, and try to improve the negative aspects, while keeping in mind all the intersting ideas that were developped in this section. That will be the object of the next section.

3 Possible improvements, new concepts.

In this section, we will examine some important weaknesses of the previous system, and try to find a way to eliminate them. This will lead us to draw the outlines of a new system based on some identical ideas, but with new concepts as well. We hope that our system will not only be an improvement of the previous one, but will provide more possibilities and new functions as well. We will start this section by studying some possible conversations, and find out why their queries are unproperly understood by the system.

3.1 Some unefficient dialogues

We would like here to illustrate some weaknesses of the system by considering (non unprobable !) dialogues, where the user's query would be misunderstood, or non efficiently taken into account. Basically, our speaker will not be here as kind as he was in the previous section...

- EXAMPLE 1:

  S: Hello ! I'd like to buy a red shirt with green circles on it...

  D: Shall I connect you to the food departement ?

  S: No, a shirt is something that you wear.

  D: Shall I connect you to the clothing departement ?

  S: Yes, that sounds better.

  This dialogue, already given as example in the previous section, seems to work well, but we will demonstrate here that it works unefficiently. Assume again that the word "shirt" is unknown, and that the word "wear" is known by the system. The *real* meaning of the speaker's sentence "A shirt is something that you wear.", is that there exist a strong relation between the word "shirt", and the word "wear". If the system knows the word "wear" (That is, if there is a heavy connection between this word and the action "connect to the clothing departement"), the network will propose the good action, and update its parameters. But the new word "shirt" will be considered as occuring only once with this action, which is not an efficient learning. Actually, the system does not really understand what the speaker means. The supply of information is used only for finding the good action. To the contrary, a human being would know for ever that "shirt" is related to the clothing departement, since he was told that it is something to "wear". In that case, the supply of information is used not only to find the good action, but to aquire rapidly a strong knowledge of a new word. Consequently, we will try to find a more reliable way of aquiring new vocabulary, based not only on knowing the good response to the query, but on the reference to previous knowledge as well.

- EXAMPLE 2:

  S: Hello ! I'd like to buy a red shirt with green circles on it...

  D: Shall I connect you to the food departement ?

  S: No, I don't think one could eat that !!

  D: Shall I connect you to the clothing departement ?

  S: Yes, that sounds better.

Here, the system finds the good answer only by eliminating progressively the bad ones, but with no logical deduction from the speaker's answers. Moreover, at the end of this dialogue, the updating procedure will find that the word "eat" occured once with the action "connect to the clothing departement", and thus will increase the concerned connection weight. Actually, the system will have learned exactly the opposite of the meaning of the sentence. This pathological behavior comes directly from the fact that the system is based only on a statistical detection of the presence of such word in the query, regardless of any consideration of word order, or grammatical syntaxe. In our research, we will then try to improve the understanding procedure, by computing more than a simple "presence detection", possibly some kind of grammar understanding procedure.

So far, we already have two purposes in the improvement of the system: Adding a special grammar detection module, and giving the system a mean to update its knowledge not only in the traditionnal statistical way described above, but by refering to its previous knowledge too. As far as the grammatical aspect is concerned, one can object that it is already taken into account in the previous system, by adding hidden layers to the network. We are not satisfied with this solution because of the following reason: Adding a hidden layer leads to incerease considerably the amount of computation, since the network works directly on the words themselves. In the example of hidden layer described in the previous section, we see that the complexity of this layer is in the range of M^2, M being the total number of known words. One can then foresee an exponential increase of the complexity, as the number of hidden layers grows. Moreover, for such an increase of computation, only restricted aspects of the grammar are taken into account, such as the word pairs in our example. Thus, we find this system not evolutive enough to keep on investigating that way. To the contrary, we will rather use another type of statistical network: The bayesian networks [Bal]. Those networks are tentative in many ways: First, their mathematical background is stronger, since they are based on conditional independances between random variables. Consequently, they are much more evolutive. Secondly, more than one type of computation is possible: Information can be computed in any sense, which was not the case in the previous network. By giving precise values to some variables and computing the others, the network could be told "THIS is not the good action", and then compute the second most probable, instead of using an iterative algorithm.
The second purpose that we found in this section (Using the previous knowledge) will be detailed later, as it needs some clarifications on the notion of "meaning". This is the topic of the next section.

3.2 What do you mean by "mean" ?

In this section, we will describe more precisely some notions we have already refered to, such as *meaning understanding* or *grammar detection*. This will bring us to the principles we announced in the introduction. It is important to realise that in such a system, any consideration of *meaning* is focused on the tasks the system is designed for, that is, no kind of "universal understanding" is reached, but a meaning understanding, in the context of the possible tasks. This will be a great help to design the system.

In the previous system, *meaning uderstanding* meant respond correctly to the query. This means understanding a whole sentence, and not the words themselves. Moreover, we saw that with a simple statistical detection of word presence, sentences like "I don't want to buy a shirt" could not be understood. We would like here to improve the notion of meaning understanding, by introducing a way to understand the words themselves, and not only the sentences. We claim that a word is not semanticaly reffering to an action, but to *the target of an action.* For instance, consider the word "wear". In the previous system, it was highly related to the ACTION "connect to the clothing departement", and this is partly why negative sentences could not be understood. Here, we say that this word is not related to an action, but to the *clothing departement itself.* This is what we call a "target" of an action. Thus, we define a measure $P_r(k, w_m)$ that gives the probability that the word w_m is reffering to the target of the k^{th} action. That brings the first of our principles:

> FIRST PRINCIPLE: A word is "understood" when the system knows its state of reference to all the possible targets of actions.

In our example, a word is understood if the system has an array of three measures giving the probability that the word refers to the food departement, the clothing departement, and the HiFi/Video departement. For instance, at a certain time, the array of the word "wear" could be something like [0.1,0.7,0.2], and the one for the word "want" [0.3,0.3,0.4]. According to the exactitude of the estimates, a word will be "well" or "badly" understood. Here, the way of using any previous knowledge appears clearly: If the system was told "A shirt is to wear", it would link the two corresponding arrays, or start the new one with no prior belief, but the values from the other one. This is quicker than a statistical updating, and more efficient to learn new words in their context.

Another point we discussed earlier is the grammatical detection. We said that a grammar detection as performed by the hidden layer of the previous system was not satisfying because it dealed with words. As a matter of fact, consider the two following sentences:

"I want to buy a shirt."
"I want to buy a tape-recorder."

Both sentences are obviously built on the same grammatical structure, and so it would be unefficient for the system to learn two different grammar structures from these sentences, just because they refer to two different actions. Actually, a good grammar detection device would not deal with words themselves, but with *grammatical classes* of words. In that case, learning would be much faster, since there are obviously far less classes than words. (In recent word classification works, a 10% reduction was approached [Jarl] [Neyl].) This brings us to the second principle.

> SECOND PRINCIPLE: The grammatical structure of a sentence does not depend on the words of the sentence, but on the grammatical classes of those words.

With this second principle, a new problem comes out: If the device detects the same grammatical structure for the two preceding sentences, how will it choose the good action, or how will it simply choose an action ? If we compare the two sentences, we find out that both of them require an action "connect to somewhere", and that the target of the action is given only by the last word. Actually, the grammatical structure defines a certain kind of action (connect...), and the target is precised by something that has nothing to do with grammar (a single word here). We understand now that the grammar detection device does not even have to take into account the last word. To detect the grammatical structure, having something like "I want to buy a XXX" is sufficient to understand that the speaker wants to be connected somewhere. Here, we find a way to simplify once more the grammar detection module, by substituing "well-choosen" words [4] by a single symbol (XXX), which decreases again the number of classes needed to understand the grammar. In a similar way, those well-choosen words will be sufficient by themselves to define the target of the action, without any grammatical consideration. This new way of understanding the concept of grammar can be modelized by the notion of *function and argument*: The grammatical structure defines a certain *function* (for instance connect somewhere...), and the words that were not used in the grammar detection define the *arguments* of the function (for instance "shirt" defines the argument "clothing departement"). This conception of meaning understanding brings to more principles, perhaps the most important ones:

> THIRD PRINCIPLE: The grammatical structure of a sentence does not define an action, but a *type* of action, which we call "function".

> FOURTH PRINCIPLE: The words in the sentence then define the targets, which we call "arguments" of the functions.

Finaly, those principles introduce a parrallel between our conception of words, and of classes. We said that the words influenced the arguments (the targets), and that the grammar (that is the classes of words) influenced the functions themselves. Thus, we introduce here the last principle which constitutes the equivalent of the first one, but for the classes:

> FIFTH PRINCIPLE: A class of words is "understood" when the system knows its state of reference to all the possible functions.'

To illustrate concretly the consequence of these principles, we give now some examples of queries, and their interpretations:

"Hello ! I want to buy a shirt."

> Here, the function detected will be "connect", and the argument will be defined by "shirt", that is the clothing departement.

"A shirt is something to wear."

> Here, the function would be "Make a link", and the arguments would be "shirt" and "wear".

"No, I don't want to buy food !!"

[4] How to get "well-choosen" words will be discussed in the next section.

Here, the function is "Eliminate a possibility", and the argument is given by food: The food departement.

We can imagine that such a device will be able to detect more than one grammatical structure per sentence, and thus that a complex sentence like "I don't want to eat my shirt !! It's a cloth !!" would be no problem for it. Moreover, it is clear now that this sytem is far more evolutive that the preceding one, since it is up to the conceiver to add functions, and so to make the system more clever, more precise, and more general. Whereas in the previous one, improving the grammar detection meant adding a whole hidden layer to the network, here, to improve the understanding of the system, it is only necessary to add one "function node" to the output layer.
Two other aspects of this system have to be emphasized now: The first one is that here, the possible actions are not any more restricted to the original extern tasks, but can include other tasks, and even "intern" ones consisting in modifying intern parameters (the Link function for instance), which was before done only by the updating procedure, and thus uncontrollable by the user. The second one is that since the system is still based on a (more powerfull) statistical network, it can be trained, as before, with a conversational algorithm, and so it preserves the attractive properties of the "learning by doing" concept.

We are now able to draw the outlines of the new system, define the different modules that are needed, and give basic ideas about their structure. This is the purpose of the next section.

3.3 Outlines of the device

We give here a general schema of the device, with the differents modules it is built from. Nevertheless one must not expect a precise description of each part of the system, since most of the ideas are still under reflexion, and many parts remain unclear. We will here propose some general ideas on each part of the system, and emphasize the different problems we are faced with, and that we will have to solve in our future work.

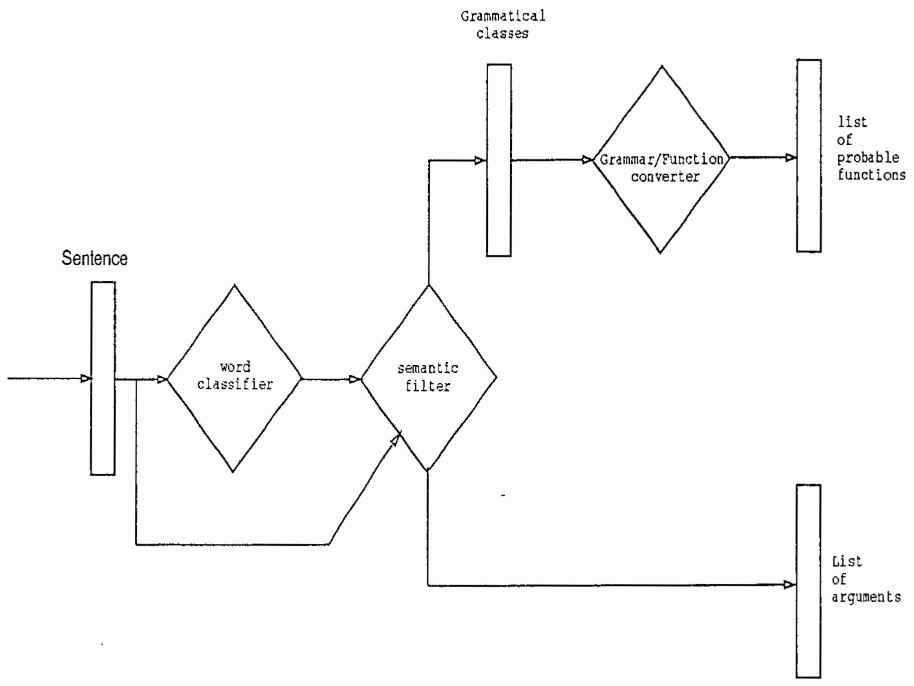


図 4: General schema of the system and its modules

- The classifier:

  Recently, researches were done on automatic word classification, and gave interesting results [Jar1] [Ney1]: A number of classes approaching 10% less that the number of words, classes *grammaticaly* significant such as articles, subjetc etc., and classes *semanticaly* significant, for instance verbs related to some kind of movement. The automatic classification concept is attractive but might not be implemented exactly in the same way as in [Jar1] or [Ney1]. As a matter of fact, the "random" aspect is too important in those algorithms, to permit an efficient classification in our case. For instance, the initial repartition of the words should not be a single class, as described in [Jar1], but rather with one class per word, or with some previous basic knowledge. In fact, a wrong classification is dangerous because it can introduce "noise" in the grammar understanding module, and thus, make the learning procedure weak. We shall rather investigate an algorithm that does not accept wrong classification at the begining, but that finds the most appropriate one at each step.
  Moreover, we think that the algorithm must not be based only on statistical and probabilistic considerations, like in bigram or trigram optimisation models, but must take into account the fact that our "meaning understanding" is focused, and not universal. Thus, the progressive classification of words would depend partly on the meaning caracteristics of the words. To illustrate this problem, considere the word "wear". In traditionnal systems, it may be classified with other transitive verbs, such as "want", or "buy"... but in our system, "wear" is semanticaly significant, since it referes only to the clothing departement, whereas "want" can be encountered in any of the three actions. Then we probably would have to build differents classes for those verbs, *because the semantic aspect has to be considered.*

- The semantic filter:

This is the most clear of the modules in the system. Its purpose is both to give an ordered list of classes on which the grammar detection module could work, and to determine the arguments of the detected functions as well. Its functionement is based on a corrolar to the principles detailed in the last section: We claim that *if our classification is done correctly, a word (or a class of words) cannot be both semanticaly and grammaticaly significant*. To illustrate this, we take some examples: The word "shirt" is semanticaly significant, since it can only refere to the clothing departement. The word classifier might put it in a class such as a direct objet complement class, maybe with other words like "trousers", "hat" etc... And it happens here that that kind of words is not grammaticaly significant (in the sense defined in the fith principle), since they does not refere to any function in particular. So, that word can be used as an argument, and the semantic filter will just send a special "class A" information to the grammar detection module. To the contrary, considere the word "want". This word is not refering to any target of the system (One can want a cloth, or an apple-pie), but is highly refering to the function "Connect". This is exactly what we call "being grammaticaly significant". Then, the semantic filter will not send this word as an argument, but will send a class highly refering to the action "connect" to the grammar detection module. In that way, there cannot be any redundancy in using the words of the sentence. We give now a more synthetic formulation of the preceding corrolar, and add a schema that permits to visualise the action of the semantic filter, with a particular sentence.

`CORROLAR: A word which is semanticaly significant for the targets of the system will be classified in a grammaticaly meaningless class. Identiquely, a grammaticaly significant class contains only words that are not refering to any target in particular, that is, meaningless words.`

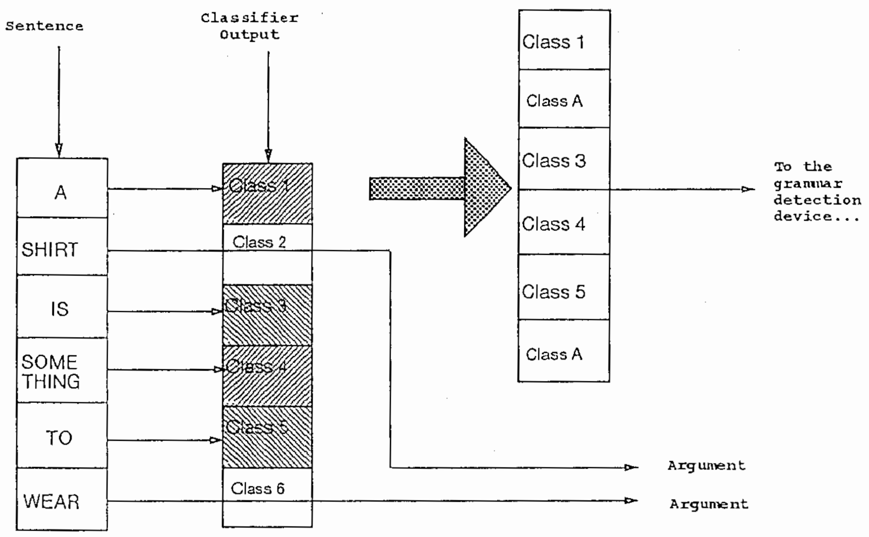


図 5: The semantic filter can be seen as a mask...

- `The grammar/Function converter:`

This is the "heart" of the system, the network that will detect the grammar issued from the list of the classes, and give for each possible function, the probability that this function be meant by the speaker. The type of bayesian network to be built there remains unclear, and is subject to future research. However, it will surely have to take into account considerations of word order (multi-gram models ?) more than simple statistical presence detection. One point has to be clarified here: We already talked about the possibility of decoding more than one function in each sentence. But then we are faced with the problem of the number of arguments. If more than one function is activated, how will the system decide which argument(s) goes to which function? Note that in any probable conversation, one given grammatical structure referes to only one function. Consequently, if more than one function is probable in a sentence, then more than one known grammatical structure is present. So, the system can detect *sequentially* the differents structures, and associate *sequentially* the arguments to their function. The kind of grammar learning developped by Helmut Lucke [Lu1] could be an interesting way of investigation, since it is based on a statistical learning of grammar with a bayesian network.

- `The algorithm:`

  Very similar to the one used in the previous system, it will feature a conversationnal-functionning mode. Before, the algorithm asked for confirmation of the action it found the good one. Here, more generally, it will ask confirmation for all the functions it finds probable. That will be part of the training procedure. We would like to give some examples of typical conversations that we hope the system will handle.

S: Hello, I would like to buy a 501[5].

> Here, the device detects the function "Connect", but the argument (501) is unknown. It computes the sentence without this word, and finds (We can do that thanks to the bayesian networks) that the most probable word knowing the others is "tape-recorder"...

D: Shall I connect you to the HiFi departement ?

S: No!! 501 is a mark of trousers[6]!!

> The device computes this sentence, detects the function "Link" with an unknown argument, and with "trousers" (that we assume is already known) and ask confirmation...

D: Do you mean that 501 as something to do with the clothing departement ?

S: Yes, I'm glad you understood that !

> The system links the two words, by starting the word "501" with the parapeters of the other one, and executes the good action.

D: I connect you to the clothing departement.

[5]You know what that is, don't you ?!

[6]Now, you know...

Here is another example that shows how the system can resist to "clever" users:

S: Well, I'm not interested in food, I want a shirt.

The first function detected is "eliminate", with the argument given by "food". According to wether the system is confident or not, it can (or not) ask the following question:

D: Do you mean that you don't want to be connected to the food departement ?

S: Yes, exactly.

The system computes the second part of the sentence, and find the action "connect", with the argument "shirt":

D: I connect you to the clothing departement.

4 Conclusion

In this paper, we have presented a recent work based on a special way of understanding the notion of speech recognition. The two most attractive aspects of this work were *focusing on meaning understanding*, and *featuring a conversational-mode algorithm*, that permited to integrate both training and functionning in the same, unique period. This work motivated then a new reflexion on the notion of meaning understanding, and some conclusion were found, that would permit to improve the previous system, and even to generalize it to a more extended field of understanding. Those conclusions were mainly formalized in five principles about the notions of meaning, grammar, and understanding. After reading this paper and in addition to those principles, one should keep in mind the following points:

The fact that our conception of meaning is focused on several precise points, and not at all a universal meaning recognition is very helpfull for building a system and for simplifying its components, and should be exploited as much as possible.

The "learning by doing" concept seems quite challenging for the future since it provides constantly evoluting, less supervised systems, actually, more close to the human being's behavior.

We would like to emphasize on the new concepts that were introduced here, particularly on the way to separate datas according to their semantic, or grammatical inference, and to treat them in parrallel, in two different ways. Altough each part of such a system is not totaly defined, we find this idea very hopefull, and we think that those general concepts could be succesfully used in more general systems, to make their learning procedure stronger, and their abilities far more evolutive.

REFERENCES

Go1 A.L. Gorin, S.E. Levinson, A.N. Gertner and E.R. Goldman, "Adaptative acquisition of Language", Computers, Speech and Language, vol. 5, no. 2, pp. 101-132, April 1991.

Go2 A.L. Gorin, S.E. Levinson, A.N. Gertner, "Adaptative Acquisition of Spoken Language", Proc. of ICASSP 91, pp.805-808, May 1991.

Go3 A.L. Gorin and L.G. Miller, "Structured Networks for Adaptative Language Acquisition", Proc. of ICASSP 92, pp.201-204, March 1992.

Go4 A.L. Gorin and L.G. Miller and S.E. Levinson, "Some Experiments in Spoken Language Acquisition", Proc. of ICASSP 93, pp. 505-508, April 1993.

Jar1 M. Jardino and G. Adda, "Language Modeling for CSR of Large Corpus using Automatic Classification of Words", Proc. EUROSPEECH'93, vol.2, pp.1191-1194.

Ney1 R. Kneser and H. Ney, "Improved Clustering Techniques for Class-Based Statistical Language Modelling", Proc. EUROSPEECH'93, vol.2, pp.973-976.

Lu1 H. Lucke, "Inference of Stochastic Context-Free Grammar Rules from Example Data using the Theory of Bayesian Belief Propagation", Proc. EUROSPEECH'93, vol.2, pp.1195-1198.