Republic of Iraq
Ministry of Higher Education
Al-Nahrain University
College of Science

Symbolic Learning System for Natural Language Understanding

A THESIS
SUBMITTED TO THE
COLLEGE OF SCIENCE, AL-NAHRAIN UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE IN
COMPUTER SCIENCE

By

Zeina Abdul Razzaq Al-Jassani
(B.Sc. 2002)

SUPERVISORS

Dr. Moaid Abdul Razzaq Fadhil **Dr. Taha Saadon Bashaga**

2005 **1426**

بسم الله الرحمن الرحيم

(وقُل اعمَلوُا فَسَيَرىَ اللهُ عَمَلَكُم وَرَسُولُهُ و المُؤمِنُونَ وَ سَتُرَدّونَ إلى عالِمِ الغَيبِ و الشَّهادَةِ فيُنَبِئُكُم بِما كُنتُم تَعمَلُونَ)

صدق الله العظيم

(التوبة:105)

الإهداء

إلى نور عيني و نبع الحنان الذي لا ينضب ... والدتي الحبيبة

إلى من أضاء لي طريقي ... والدي الحبيب

إلى سندي و عوني ... أختي و أخوي الأحبة

و إلى كل من أحب قلبي...

زينة

Acknowledgments

Many people deserve thanks in helping me to finishing my dissertation. I'll say one big generic thanks to all who deserve to be thanked but aren't mentioned here by name.

I would like to express my sincere gratitude and appreciation to my supervisors Dr. Moaid Abdul Razzaq Fadhil and Dr. Taha S. Bashaga for their able to guidance, supervision.

Special thanks to the College of Science, Dean of the College for the continuous support and encouragement during the period of my studies.

Grateful thanks for the Head of Department of Computer Science Dr. Taha S. Bashaga, staff and employees.

Special thanks to my friends and especially Abeer Khalid for their help.

Finally, my special thanks to my family for their continuous support and encouragement during the period of my studies.

Abstract

The present work is an attempt of designing a symbolic based learning system for natural language understanding. The field selected to be the domain of application is the subject of data structure.

The technique of learning used is Symbolic Learning which is a combination of two learning strategies Rote Learning and Learning by Instruction. Symbolic learning methods are being developed for aiding the construction of systems requiring extraction of information from natural language documents and subsequent natural language querying of the resulting database.

The system consists of the following modules, Process Query (Information Source) input to the learning system represented by the query entered by the end user, The Learning Engine carries out the learning task and produce knowledge for the knowledge base, Performance Engine make sure that the knowledge produced is useful .

Visual Prolog Version 5.1 was used for building the system and its interface which provides Visual Development Environment (VDE).

Contents

List of Figures

Contents

Chapter One

Overview

1.1. Introduction

Learning is the hallmark of intelligence; many would argue that system cannot learn is not intelligent. Without learning, everything is new; a system that cannot learn is not efficient because it re-derives each solution and repeatedly makes the same mistakes, if any.

Machine Learning is a branch of Artificial Intelligence (AI). The field is currently undergoing a significant period of growth, with many new areas of research and development being explored [1].

Machine Learning is the domain of Artificial Intelligence which is concerned with building adaptive computer systems that are able to improve their competence and/or efficiency through learning from input data or from their own problem solving experience.

The basic premise of the Inferential Theory of Learning as stated by Michalski is that in order to learn an agent has to be able to perform inference and to have memory that both stores the background knowledge needed for performing the inference and records 'useful' results of inference. Without either of the two components - the ability to reason and the ability to memorize and retrieve information from memory - no learning can be accomplished. Thus one can write an equation:

Learning = Inference + Memorizing ...……. (1.1)

Memory is seen here to be performing a dual role, one as a supplier of background knowledge and the other as a storage area or depository of results. This dual function of memory is one that is often seen in the organization of learning systems [2].

There are several basic learning situations. These include rote learning or learning by induction. In this process, the learning system obtains knowledge from the environment in a form that it can use directly, this information is memorized for later use. Learning by taking advice or explanation-based learning is another example of a learning process. In this situation, the learning system is given general-purpose advice that it must then transform into a format that can be used by the system to improve its performance. Learning by discovery speaks to the concept of making use of the large amount of available data, discovering patterns and relationships among these data sets, and learning from these patterns and associations [3].

This research employs a combination of two methods rote learning and learning by instruction methods.

1.2. Literature Survey

There are several research approaches that are concerned with natural language and machine learning. Some of them are summarized below:

- Brill Eric, PhD. Thesis, <u>A Corpus Based Approach to Language Learning,</u> Department of Computer and Information Science, University of Pennsylvania 1993.

  The goal of this dissertation is to make significant progress toward designing automata that are able to learn some structural aspects of human language with little human guidance. In particular the thesis describe a learning algorithm that takes a small structurally annotated corpus of text and a larger un annotated corpus as input and automatically learns how to assign accurate structural descriptions to sentences not in the training corpus. The main tool used to automatically discover structural information about language from corpora is transformation based error driven learning. The distribution of errors produced by an imperfect annotator is examined to learn an ordered list of transformations that can be applied to provide an accurate structural annotation. The application of this learning algorithm is demonstrated to part of speech tagging and parsing. Successfully applying this technique to create systems that learn could lead to robust trainable and accurate natural language processing systems [4].

- Sunglian William Kung, <u>Exploiting Equity Momentum with Symbolic Machine Learning,</u> University of Oxford, 1998

  The thesis investigates the use of machine learning to identify regularities in the phenomena of momentum within the equity market. Empirical results obtained here suggest that is possible to obtain some degree of predictability in stock return movement. Results also suggest that investment strategies derived from machine learning. This study take first step towards incorporation of first order machine learning techniques as embodied by inductive logic programming into stock market prediction [5].

- Cynthia Ann Thompson, PhD. Thesis, ”Semantic Lexicon Acquisition for Learning Natural Language Interfaces”, University of Texas, 1998
  This dissertation presents a lexicon acquisition system that learns a mapping of words to their meanings. First, the only background knowledge needed for lexicon learning is in the training examples themselves. Second, interaction with a system that learns to parse is demonstrated. Third, a simple, algorithm is used for efficiency to acquire word meanings. Fourth, the system is adaptable to different sentence representations. Finally, the mapping problem is eased by making a compositionality assumption that states that the meaning of a sentence is composed from the meanings of the individual words and phrases in that sentence, in addition, perhaps to some connecting information specific to the representation at hand [6].

- Kazakov Dimitar, PhD. Thesis, Natural Language Processing Applications of Machine Learning, Czech Technical University, Prague 1999.
  This thesis offers an overview of some of the machine learning techniques used for the purposes of natural language processing.The thesis describes two original projects. The first one introduces the system at the inductive learning of parsers of natural language. The second suggests a bias for unsupervised word segmentation. A genetic algorithm is applied to reduce the search space and draw the first draft of the word segmentations. Then a set of rules for word segmentation are found and expressed in high order formalism (first-order-logic) by the means of inductive logic programming

techniques; the application of those rules to the training data produces the final word segmentations [1].

- Fakhri Sawsan, M.Sc. Thesis, Machine Learning a Multistrategy Approach, National Computer Center 2001.

  This research sheds on the prominent monostrategy learning types within a symbol based framework as this will help in having a clear understanding of the role and the applicability conditions of different learning strategies. But the focus will be upon the foundational ideas and theoretical framework of multistrategy learning that typifies such features in a proposed multistrategy medical learning system. This work devoted to the food and drug guide for patients with diabetes mellitus [7].

- Muaid Esraa, M.Sc. Thesis, Database Communication using Arabic Natural Language, Department of Computer Science, Al-Nahrain University, 2001

  This project is an attempt to develop the process of retrieving information using Arabic natural language as an inference, by merging semantic analysis with syntax analysis through the conversation from the surface structure, which is the form in which the query was entered, to the structure, which is an internal logical form and it simulate the way that we express as an idea by it.

  The system was evaluated by performing several evaluation processes, these evaluations gave a good results because the system was able to understand most of users queries in a correct way and to give accurate solution to it, and because it provides abilities to help the user to perform searching and updating processes to the database contents and in flexible manner [27].

1.3. Aim of the Thesis

The aim of the thesis is to design and implement a symbolic machine learning system for understanding natural language. This system interacts with computer system using the English natural language, from the very primitive language tokens to build learning knowledge base. The knowledge base is the basis for the system in understanding the requirements and the queries of the users which are using natural language.

1.4. Thesis Layout

- Chapter one presents the introduction to this work, the literature survey, and the aim of the work.

- Chapter two focus on related theoretical concepts about machine learning, its strategies, why should machine have to learn, benefits and limitations of natural language understanding.

- In chapter three, a detailed description of the learning engine modules and other modules and algorithms are presented.

- Chapter four discuss the conclusions of this work which are given together with some recommendation and future work.

Chapter Two

Machine Learning and Natural Language Understanding

2.1 Introduction

The concept of learning may be regarded as any process through which a system utilizes knowledge to improve its performance. Machine learning is one of the central areas of Artificial Intelligence/Computational Intelligence. The Essence of Learning is to construct or improve knowledge by exploring input information and prior knowledge. The concept of the "learning strategy" is loosely defined and is used to characterize different aspects of the learning process [2].

The application of learning techniques to natural language processing has grown dramatically. Machine learning can provide natural language processing a range of alternative learning algorithms as well as additional general approaches and methodologies. On the other hand, natural language can provide machine learning with a variety of interesting, important, and challenging problems [8].

In this chapter several aspects will be discussed, they are, the definition of idea of machine learning, its strategies, the reason of using machine learning and where to be used, and finally natural language understanding and its characteristics also presented.

2.2 Machine Learning

Sometime humans may work to learn from their mistakes. In contrast, programs always work in the same way – if there is an error in the code that causes a certain mistake to occur, then that mistake will be repeated every time that part of the code is executed. In order to add some ‘intelligence’ into our programs one need them to be able to learn from their mistakes & adapt. Many AI programs do not do this – knowledge is built in at the start and remains the same throughout – such systems are therefore limited – especially in terms of exhibiting intelligent behaviour, but nevertheless they still do a useful job. Programs that learn enable more complex jobs to be done – either by improving their performance based on experience, or as a mean of acquiring knowledge for expert systems. [9]

One needs to know what an intelligent system is. The concept of "intelligent system" needs to be defined in terms of capabilities that it should process, regardless of whether it is a human, an animal, or a machine.
A system is called intelligent if it has the ability to:

- **Perceive**, that is, it is able to collect information about its environment, this mean that it must be equipped with some sensors and be able to interpret their output.
- **Learn**, that is, it is able to create knowledge from the perceive information and to memorize it.
- **Reason**, that is, it is able to conduct inference employing perceived information and previously acquired knowledge to accomplish its goals [2].

Simon (1993) defines learning as **‘**changes in the system that are adaptive in the sense that they enable the system to do the same tasks drawn from the same population more efficiently and more effectively next time’ [10].

There are two basic forms of learning: Knowledge acquisition and skill refinement.

Knowledge acquisition is defined as learning new symbolic information and coupled with the ability to apply that information in an effective manner. When we say someone learned physics, understood their meaning, and understood their relationship to each other and to the physical world. The essence of learning in this case is the acquisition of new knowledge, including descriptions and models of physical system and behaviors. A person is said to learn more if his knowledge explains a broader scope of situations, is more accurate, and is better able to predict the behavior of the physical world.

A second kind of learning is the gradual improvement of motor and cognitive skills through practice, such as learning to ride a bicycle or to play the piano. The machine learning focuses on the knowledge acquisition aspect of learning [11].

- *Learning Declarative Knowledge* (Knowledge generation) e.g. "The largest proven prime number is 2 multiplied by itself 765,839 times minus 1", i.e. one learn the largest proven prime number but in the future one may discover and learn a new number largest than first one.
- *Learning Procedural Knowledge (skill learning, performance improvement)* e.g. "Since August, Elizabeth has improved her typing speed by 30% and decrease her error by 5%", i.e. Elizabeth is increase her skill in typing [2].

Learning can be described as normally a relatively permanent change that occurs in behavior as a result of experience. Learning occurs in various regimes. For example, it is possible to learn to open a lock as a result of trial and error.

Machine learning (ML) is the study of computer programs that improve through experience their performance at a certain class of tasks, as measured by a given performance measure. In more detail, the learning system aims at determining a description of a given concept from a set of concept examples provided by the trainer and from the background knowledge [1].

2.2.1 The Necessity of Machines to Learn

There are several reasons why machine learning is important some of these are: [12, 13]

- Some tasks cannot be well defined by experts example; that is one might be able to specify input output pairs but not a concise relationship between inputs and desired outputs. Machines required being able to adjust their internal structure to produce correct outputs for a large number of sample inputs and thus suitably constrain their input/output function to approximate the relationship implicit in the examples.

- It is possible that there are an important relationships and correlations hidden among large piles of data. Machine learning methods can often be used to extract these relationships (data mining).
- Human designers often produce machines that do not work as well as desired in the environments in which they are used. In fact certain characteristics of the working environment might not be completely known at design time. Machine learning methods can be used for job improvement of existing machine designs.

- The amount of knowledge available about certain tasks might be too large for explicit encoding by humans Machines that learn this knowledge gradually might be able to capture more of it than humans would want to write down.

- Environments change over time. Machines that can adapt to a changing environment would reduce the need for constant redesign.
- Recent progress in algorithms and theories.

2.2.2 The Architecture of a Learning System

The learning system consists of the following components:

1. **Problem solving Engine**: Implements a general problem solving method that uses the knowledge from the knowledge base to interpret the input and provide an appropriate output.

2. **Learning Engine**: Implements a learning method for extending and refining the knowledge base to improve agent's competence and/or efficiency in problem solving.
3. **Knowledge Base**: Data structures that represent the objects from the application domain, general laws governing them, actions that can be performed with them, etc.

4. **User / Environment**: informs the learner of how well the agent is doing provides training examples to the learning element [14, 15].

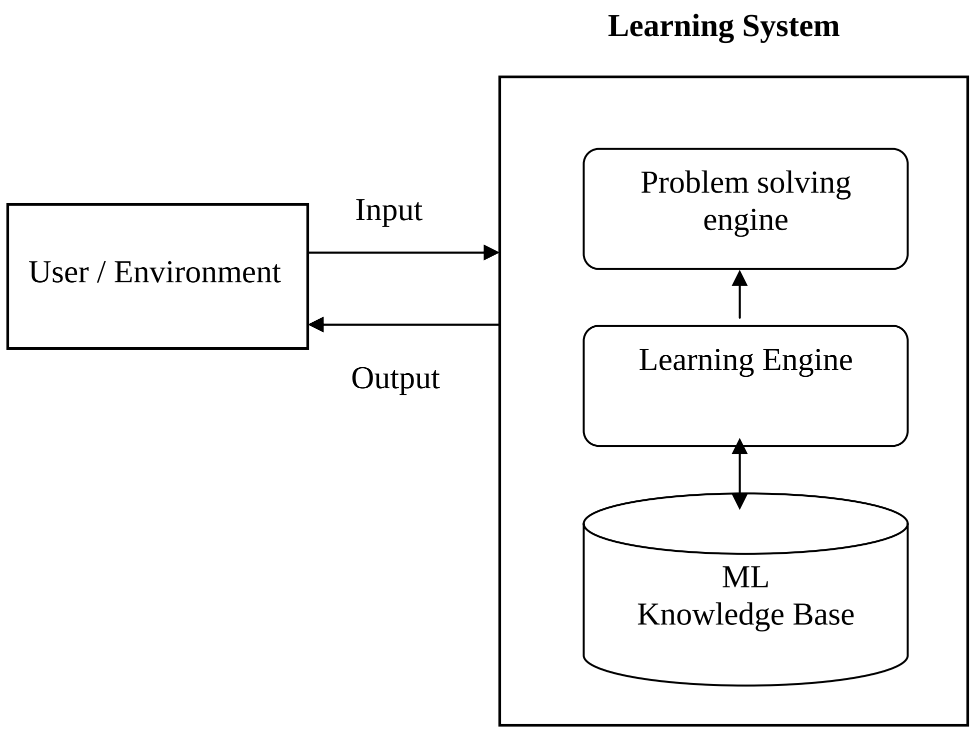


Fig 2.1 The Architecture of Learning System

2.2.3 Models of learning

There are several elements that any "model of learning" must specify [15]:

1. <u>Learner</u>*:* "Who" is doing the learning? Typically, it means a computer program. The learner may be restricted.

2. <u>Domain</u>*:* "What" is being learned? For example, the learner may be trying to learn an unknown concept, such as a chair. Concept learning, where the learner is trying to come up with a "rule" to separate positive examples from negative examples, is one of the most studied domains. There are many other types of things that can be learned: an unknown device (e.g., an unknown technique (e.g., how to juggle), an unknown function, an unknown environment (e.g., a new city), an unknown language, an unknown family of similar phenomena (e.g., speech recognition, face recognition, or character recognition) …etc.

3. Information Source: "From what" is the learner learning? How is the learner informed about the domain? There are many ways the learner can be informed by:

(a) Examples: The learner is given positive and negative examples. These examples can be chosen in a variety of ways. They can be chosen at random, from some known or unknown distribution. They can be chosen arbitrarily. They can be chosen maliciously, by some adversary who wants to know the worst-case behavior of a learning algorithm. Or, the examples can be carefully chosen by a helpful teacher who wants to facilitate the learning process.
An important issue to address here is how the examples are described. That is, how are the features of the example specified?

(*b) Queries*: The learner may get information about the domain by asking questions to a teacher. For example, the learner may ask `` is a stool an example of a chair?" Or, it may ask ``Is this a correct floor plan for the third floor?" The first type of question is known as a membership query, where the learner asks for the label of an example. The second type of question is known as an equivalence query, where the learner provides a rule and asks if that rule is correct.

(c) *Experimentation:* The learner may get information about the domain by actively experimenting with it. For example, a learner may learn how to draw a map of a new city by walking around it.

2.2.4 Machine Learning Techniques

There are two most frequently used learning techniques symbolic learning technique and neural networks technique. These two techniques will be discussed below:

2.2.4.1 Symbolic Learning

Symbolic machine learning techniques, acquire non numerical knowledge and can be classified, based on such underlying learning strategies as rote learning, learning by analogy, and learning from examples, among these techniques, learning from examples, a special case of inductive learning, appears to be the most promising technique for knowledge discovery in real databases. It induces a general concept description that best describes the examples [16, 5].

Symbolic learning methods are being developed for aiding the construction of systems requiring extraction of information from natural-language documents and subsequent natural language querying of the resulting database [17].

2.2.4.2 Neural Networks

The foundation of the neural networks paradigm was laid in the 1950s and has attracted significant attention in the past decade due to the development of more powerful hardware and neural network algorithms. Nearly all connectionist algorithms have a strong learning component.

In symbolic machine learning, knowledge is represented in the form of symbolic descriptions of the learned concepts. In connectionist learning, on the other hand, knowledge is learned and remembered by a network of interconnected neurons, weighted synapses, and threshold logic units [16].

2.2.5 Classification of Learning Strategies

There have been a number of different approaches to machine learning that can be classified based on primary inferential strategy as: [2]

2.2.5.1 Rote Learning

Storing data is a basic form of learning. More data generally leads to better performance. Can also use - data caching - where the computed values is stored so that they do not have to be re-computed later. This can save time in certain situations. Such caching is known as rote learning [9].

Rote learning consists of memorizing the solutions of the solved problems so that the system needs not to solve them again:

$$\begin{tikzcd}(X_1, \dots , X_n) \arrow[r, "f"] & (Y1, \dots , Y_p) \arrow[r, "\text{Store}"] & {[(X1, \dots , X_n), (Y_1, \dots , Y_p)]}\end{tikzcd}$$

Input Patterns   Performance Function   Output value of computation   Associated Pair

During subsequent computations of $f(X_1,..., X_n)$, the performance element can simply retrieve $(Y_1, ... , Y_p)$ from memory rather than re-computing it such as shown above

Where

X: input patterns

Y: Output value of computation

f: Performance function

There are several issues in the design of rote learning systems:

- **Memory organizations**

  Rote learning requires useful organization of the memory so that the retrieval of the desired information will be very fast.

- **Store-versus-compute trade-off**

  The cost of storing and retrieving the memorized information should be smaller than the cost of re-computing it.

Rote learning needs no prior knowledge. In this case of learning, examples of correct behavior are stored and when a new situation is encountered it is matched with the learnt examples. If one of the examples matches, the relevant response is given. Training consists simply of memorization, and the output of the training set is just the stored training set. For example Samuel's CHECKERS program, it is one of the earliest game-playing programs, this program learned to play checkers well enough to beat its creator/design, this program used minimax search procedure to explore checkers game trees. Rote learning was employed to save the results of previous game tree searches in order to speed up subsequent search. This is the extreme case in which the learner does not have to perform any inference on information provided. The knowledge supplied by the source is directly accepted by the learner [10, 16].

2.2.5.2 Learning by instruction

In this type of learning, the learner acquires concepts from a teacher or other organized source, such as publication or textbook [10]. The challenge to learning from instruction is the translation of the instruction from the natural language that is convenient to the human teacher into knowledge structures and operations suitable for use by the program [18]. The person giving natural language advice is often unaware of the internal structure of the system receiving advice to the internal structure by itself [19].

Early in the history of artificial intelligence coined the term ‘advice taker’ to describe a program that could learn from helpful teacher. But progress towards actually implementing this dream has been slow. The reason is that it is hard to device widely enough channels of communications from the teacher to the advise taker. Ideally, communication should be in English. This raises all the problems. Most natural language programs have dealt with simple stories, not general rules. The problem of ambiguity, pronoun preference, and so forth can only get worse as the sentence to be understood get more complex, for example in chess “fight for control of the center of the board” is useless unless the player can translate the advise into concrete moves and plans [7].

2.2.5.3 Learning by deduction

Deductive learning works on existing facts and knowledge and deduces new knowledge from the old. This is best illustrated by giving an example. For example, assume:

A = B

B = C

Then we an deduce with much confidence that

C = A

Arguably, deductive learning does not generate "new" knowledge at all; it simply memorizes the logical consequences of what is known already [20].

A form of deduction is Explanation Based Learning (EBL), which require a very large number of examples in order to ensure that the rules learnt are reliable. Explanation-based learning addresses this problem by taking a single example and attempting to use detailed knowledge in order to explain the example. Thus the attributes required in the explanation are taken as defining the concept [21]. The explanation-based programs are thought to be taking the following as input:

1. *Target Concept*: This learner's task is to determine an effective definition of this concept depending upon the specific application. The target concept

may be classification, theorem to be proven, a plan for achieving a goal, a heuristic for a problem solver.

2. *Training example*: An instance of the target.
3. *Domain Theory*: A set of rules and facts that are used to explain how the training example is an instance of the goal concept.
4. *Operationality Criteria*: Some means of describing the form that concept definition may take.

From the above, EBL computes a generalization of the training example that is sufficient to describe the goal concept, and also specifies the operationality criterion. An EBL program seeks to operationalize the goal concept by expressing it in terms that a problem-solving program can understand. These terms are given by the operationality criterion. [21]

'An EBL system attempts to learn from a single example X by explaining why X is an example of the target concept. The explanation is then generalized, and the system's performance is improved through the availability of this knowledge.'[18] The following is an example that illustrates the idea of explanation based learning:-

$$\text{premise}(X) \longrightarrow \text{cup}(X)$$

Where a premise is a conjunctive expressions containing the variable X, Assume domain theory that includes the following rules about cups:

$$\text{liftable}(X) \wedge \text{hold\_liquids}(X) \longrightarrow \text{cup}(X)$$

$$\text{Part}\ (Z, W) \wedge \text{concave}\ (W) \wedge \text{points\_up}\ (W) \longrightarrow \text{holds\_liquids}\ (Z)$$

Light(Y) ^ part(Y, handle) $\longrightarrow$ liftable(Y)

Small (A) $\longrightarrow$ light (A)

Made of (A, feathers) $\longrightarrow$ light (A)

The training example is an instance of the target concept. That is given:

Cup (obj1)

Small (obj1)

Part (obj1, handle)

Owns (bob, obj1)

Part (obj1, bottom)

Part (obj1, bowl)

points_up (bowl)

Concave (bowl)

Color (obj1, red)

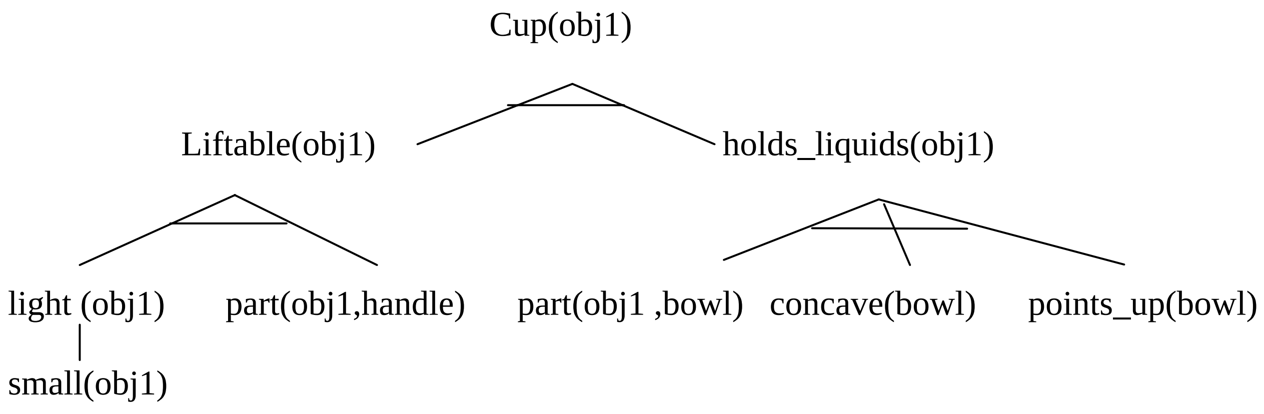
Operationality criteria requires that target concept be defined in terms of observable, structural properties of objects, such as parts and points_up.

The next stage of explanation based learning generalize the explanation to produce a concept definition that may be used to recognize other cups. EBL accomplish this by substituting variables for those constants in the proof tree that depend on a particular training instance as in fig 2.2. Based on the generalized tree, EBL define a new rule whose conclusion is the root of the tree and whose premise is the conjunction of the leaves:

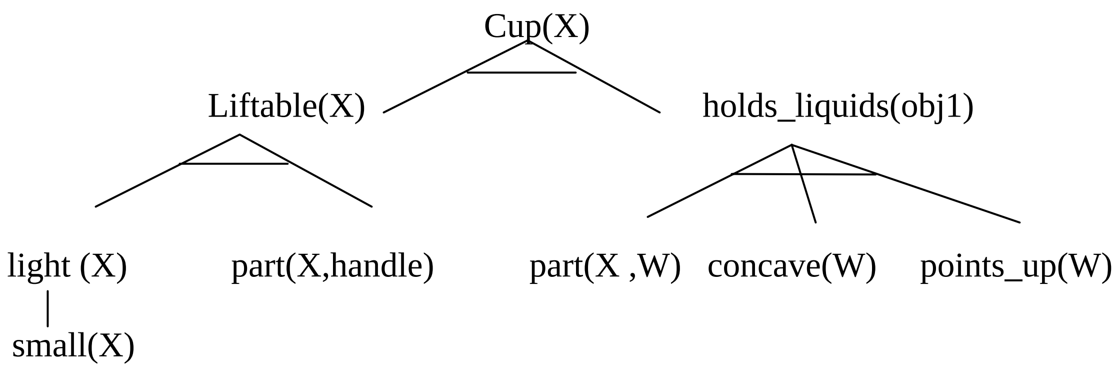
small(X) ^ part(X, handle) ^ part(X,W) ^ concave(W) ^ points_up(W) $\longrightarrow$ cup(X)

In constructing generalized proof tree, our goal is to substitute variable for those constants that are part of the training instance.

If the explanation is complete, then one can guarantee that the description is correct [18].



(a) Proof obj1 is a cup



(b) Generalized proof that X is a cup

Fig.2.2 Specific and generalized rule that an object X, is a cup

2.2.5.4 Learning by Analogy

A mapping of knowledge from one domain into another which conveys that a system of relations holding in the former also holds in the latter i.e. transfer knowledge from one database into that of different domain.

This type of learning applies existing knowledge to a new problem instance on the basis of similarities between them to acquire the concept. Analogy allows great flexibility in applying existing knowledge in a new situation, for example assume that a learner is trying to learn about the behavior of electricity, and assume that the teacher tells that "*electricity is analogous to water*", with voltage corresponding to pressure, amperage to the amount of flow, and resistance to the capacity of a pipe. Using analogical reasoning, the learners may easily grasp such concepts as Ohm's Law [18].

If a system has available to it a knowledge base for a related performance task, it may be able to improve its own performance by recognizing analogies and transferring the relevant knowledge from the other knowledge base. Suppose, for example, that a program has available to it a knowledge base describing how to diagnose disease in human beings and someone wants to use the same program to diagnose computer-system failures. By finding the proper analogies, the program can develop classes of computer failures (diseases) and possible solutions (therapies). Diagnostic procedures can be transferred as the analogy is developed (e.g., x-ray, can be analogized to core dumps) [23].

Learning by analogy is referred to as *reformulation,* because the new learnt statement is different in content but logically equivalent to the original statement. It is this logical equivalence that makes analogy a truth-and-falsity preserving means of inference [10].

2.2.5.5 Learning by induction

Learning from incomplete set of examples is called inductive learning. Inductive learning is based on the following assumption, known as inductive learning hypothesis:

"Any hypothesis found to approximate the target [concept] well over a sufficiently large set of training examples will also approximate the target [concept] well over other unobserved examples".

The concept definition learned by inductive learning is evaluated on a set of test examples. Inductive Learning takes examples and generalizes rather than starting with existing knowledge in i.e. is the process of reaching a general conclusion from specific examples, such as shown fig 2.3.

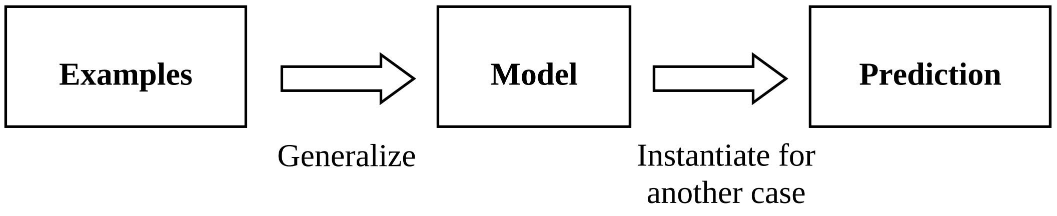


Fig 2.3 Learning by Induction

For example, having seen many cats, all of which have tails, one might conclude that all cats have tails. This is unsound step of reasoning but it would be impossible to function without using induction to some extent. In many areas it is an explicit assumption. There is scope of error in inductive reasoning, but still it is a useful technique that has been used as the basis of several successful systems [21].

Inductive learning is reasoning from specific to general in the sense that a cat is assumed to have a tail from observed experience of cats [20].

One major subclass of inductive learning is concept learning. This takes examples of a concept and tries to build a general description of the concept. Very often, the examples are described using attribute-value pairs. ID3 Iterative Dichotemiser, and Version Space are inductive learning algorithms [20]. ID3 is an algorithm for decision tree construction developed by Ross Quinlan, 1987 it is top-down construction of the decision tree by recursively selecting the "best attribute" to use at the current node in the tree. Version Space illustrates the implementation of inductive learning as search through a concept space. It take the advantage of the fact that generalization operations impose an ordering on concepts in a space, and then use this ordering to guide the search [22].

2.3. Natural Language Understanding [22]

The goal of the Natural Language Processing (NLP) group is to design and build software that will analyze, understand, and generate languages that humans use naturally, so that eventually one will be able to address his computer as though he was addressing another person.

This goal is not easy to reach. "Understanding" language means, among other things, knowing what concepts a word or phrase stands for and knowing how to link those concepts together in a meaningful way. It's ironic that natural language, the symbol system that is easiest for humans to learn and use, is hardest for a computer to master. Long after machines have proven capable of inverting large matrices with speed and grace, they still fail to master the basics of the spoken and written languages.

To *understand* something is to transform it from one representation into another, where this latter representation is chosen to correspond to a set of available actions that could be performed, and for which a mapping is designed so that for each event an *appropriate* action will be performed.

2.3.1 Uses of Natural Language

There are several uses of natural language such as: [17]

- ***User interfaces***: better than unclear command languages. It would be nice if one could just tell the computer what one want it to do, these about a textual interface not speech.
- ***Knowledge-Acquisition***: programs that could read books and manuals or newspaper. So one don't have to explicitly encode all of the knowledge need to solve problems or do whatever they do.
- ***Information Retrieval***: find articles about a given topic. Program has to be able somehow to determine whether the articles match a given query.

- ***Translation***: machines could automatically translate from one language to another. This was one of the first tasks they tried applying computers to. It is very hard.

2.3.2 Advantage of Natural Language Interface

Natural language is only one medium for human-machine interaction, but has several obvious and desirable properties: [20, 21]

1. It provides an immediate vocabulary for talking about the contents of the computer.
2. It provides a means of accessing information in the computer independently of its structure and encodings.
3. It shields the user from the formal access language of the underlying system.
4. It is available with a minimum of training.

2.3.3 The Hardness of Natural Language

There are several major reasons why natural language understanding is a difficult problem. They include: [22, 24]

The complexity of the target representation into which the matching is being done. Extracting meaningful information often requires the use of additional knowledge.

1. The level of interaction of the components of the source representation. In many natural language sentences, changing a single word can alter the interpretation of the entire structure. As the number of interactions increases, so does the complexity of the mapping.

2. Elliptical utterances. The interpretation of a query may depend on previous queries and their interpretations. E.g., asking *Who is the manager of the automobile division* and then saying, *of aircraft?*

2.4 Learning and Natural Language [25]

Why study Natural Language? Natural Language is the main channel of communication and the main channel through which we acquire knowledge. Both these aspects are important both for the study of language from a cognitive point of view and from an engineering perspective. A grand application to keep in mind that embodied all aspects and difficulties of natural language is that of human computer interaction. In particular, the problems of language understanding and generation. Specific tasks could include: Question answering, summarization, translation.

Learning Perspective: Why study learning in natural language? Natural Language is a great example and for studying and developing theories in learning. Any non trivial task in natural language involves both the study of learning and inference. The tasks involved, e.g., language comprehension are large scale phenomena in terms of both knowledge and computation.

Natural Language Perspective: Why study learning in NLP? There isn't any significant aspect of language that can be studied without giving learning a principle role. As a consequence, the role of learning in the study of natural language is significant both from the engineering point of view as it is from the cognitive point of view. Learning is important in language acquisition, language change, language variation, language generation and language comprehension.

Chapter Three

The Proposed Symbolic Learning System

3.1 Introduction

As mentioned in chapter one the aim of the project is to design and implement a machine learning system (as an adaptive tool). This system interacts with computer system using the English natural language, from the very primitive language tokens to build learning knowledge base. The knowledge base helps the system in understanding the requirements and the queries of the users which are using natural language.

The proposed system interacts with two users the first user is the **end user** and the second one is the **human expert.** The end user asks the system queries related to the domain of data structure (selected to be the domain in this work) and the system will answer the queries after reaching a specific point of learning. The learning process is done by the expert who learns the system gradually through the module "*Learning Engine*" which will be discussed in details later in this chapter.

This system is implemented using the facilities of **Visual Prolog** language since it is a very important in programming artificial intelligence applications, and it supports many advanced programming tasks such as *searching, patterns matching, database programming,* and *the creation of the user interface component.*

3.2 System Modules

The system consists of the following modules as shown in fig. 3.1, **Process Query (Information Source)** input to the learning system represented by the query entered by the end user, The **Learning Engine** carries out the learning task and produce knowledge for the knowledge base, **Performance Engine** make sure that the knowledge produced is useful, and the **Feedback** from the performance engine produces feedback to the learning engine decides between two actions either continue the learning process by asking for more information or stop.

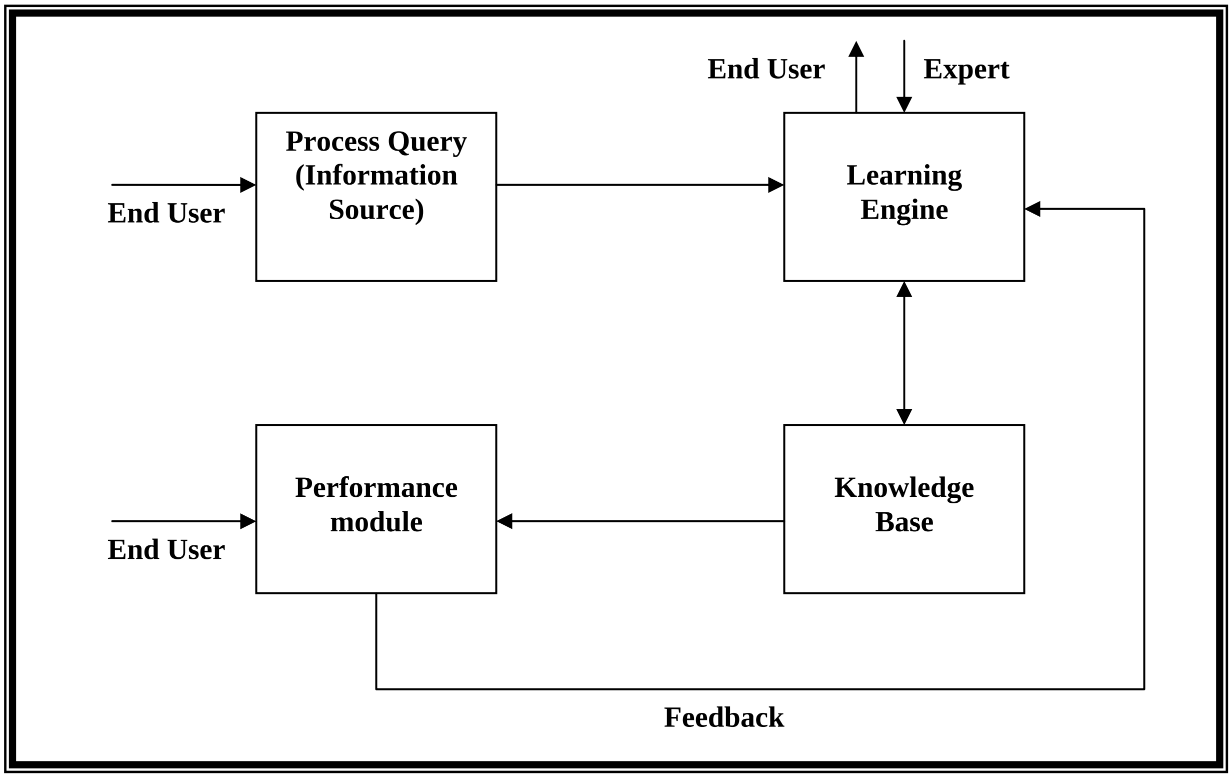


Fig 3.1 System modules

3.3. Process Query (Information Source)

Input to the learning system which is represented by the query entered by the *End User*, the input query is entered as a string. This module deals with the *dictionary* which will be discussed in the *Knowledge Base* module later on (See section 3.5.1.) This module process the query entered by the end user as shown in the following steps:

- Remove any unnecessary spaces by using "*Space eater*" which will remove any unnecessary spaces from the sentence entered by the end user.
- The input sentence will be changed to small letter case so that it will be case insensitive.
- Convert the input sentence which is string to a list of string.
- Remove the duplicate words so that it will not ask about it more than once.
- Management of synonym words by replacing each word in the list by its synonym.
- Filtering any noise words which is already exist in the initial knowledge.
- Check the word if it is not exist in the *dictionary* then add it to another list called "uwl" unknown word list

This process is illustrated in algorithm 3.1

3.4. Learning Engine

Carry out the learning task and produce knowledge for the knowledge base. This module is performed by the expert and only the expert has the authority to do the learning process.

Algorithm 3.1: Process Query

```
Input: The query entered by the end user as string (Str)
Output: Unknown word list (Uwl), Filtered list (Fl).

Step 1: Remove any unnecessary spaces by using the sub modules "Space eater"
Step 2: Convert Str to List of string L
Step 3: Remove any duplicate words in L
Step 4: for each word in L do

        Step 4.1: If the word has a synonym then replace the word with its synonym
        Step 4.2: Filter any noise word in L.
        Step 4.3: Check if the word is not exist in the dictionary then add it to the Uwl.

Step 5: Save Str, Uwl, Fl

Step 6: Stop.
```

The authorization of the expert is checked by a password that must be entered by the expert and this password will be verified with the saved password , The expert has the right to enter the password three times if non of these three passwords match the saved password, the expert will not have the right to do the learning process as shown in algorithm 3.2:

```
Algorithm 3.2: Password Verification
Input: Password from Expert
Output: True or False
Step 1:  get password (pass) from expert
Step 2:  K=1
Step 3:  While K <= 3
         Step 3.1: K=K+1
         Step 3.2: if pass= saved_password then
                                 return True
                                 goto Step 4
                   Else return False
         Step 3.3: End while
Step 4: Stop.
```

The learning engine gets its information from the knowledge base (from the dictionary) and from the entered query after processing it. There are **two phases** of learning the first phase of learning by learning each unknown word and save it in the knowledge base in the dictionary. In the beginning the system asks the expert about the field of the word, if it is concerning data structure, data type, noise word or other, if it is one of the mentioned fields the system will save the word and the selected field in the dictionary, if it is not the expert will choose other and then enter a new field that the unknown word belong to it. After each word is learned the second phase starts by learning the answer for the questions, the answer will be entered by the expert and the question and the answer will be saved in the knowledge base in the file (ques_ans.edb), After saving the question and its answer the question will be deleted from the file (question.idb) so that the answered question will not be displayed again for the expert. The question and its answer will be saved so that if the end user asks the same question again there is no need to repeat the learning process again just get the answer from the knowledge base. In the beginning the following knowledge base will be created and during the learning process this knowledge base will be updated accordingly:

- An internal database called "question.idb" which contains the questions asked by the user and it is not answered yet by the expert, and contains also a question that is already answered by the expert but during the performance module it is marked as incomplete answer and asks about more details. When the expert answers any question this question will be deleted from this file.
- An external database called "ques_ans.edb" which contain the questions asked by the end user after processing it using sub module *Process Query* and the answer for that question e.g. If the question asked by the end user

"What is Stack?", it will be saved in the file as Q_A (["what", "stack"], "An ordered list in which all insertions and deletions are made at one end").

The algorithm of learning engine which is performed on one query and it is repeated each time the expert wants to answer another question is, Algorithm 3.3 represents the learning engine.

Algorithm 3.3 Learning Engine

```
Input: Question from (question.idb)

Output: Update the existing knowledge base

Step 1: For each unknown word ( Uw ) in Uwl do
            Step 1.1: Ask the expert to determine the field of the word if it is
            one of the mentioned fields then
                Choose the mentioned field
            Else
                Choose Other
                Get the new field from the expert
                Add the new field to the file (field.idb)
            Step 1.2:  Store the word and its field in the dictionary

Step 2: If (question in "question.idb") and (marked as incomplete answer)   then
            Enter more details about the question
            Add the details to the old answer
            Save the new answer
        Else
            Enter the answer
            Save the answer

Step 3: Delete Question from Question.idb

Step 4: Stop.
```

3.5 knowledge Bases

The purpose of the knowledge base module is to store the initial primitive keywords, and to store the formulated query entered by the end user and to store the questions and its answer. The knowledge base is gradually grow and updated as long as the learning process is performed. The knowledge base can be divided as shown:

3.5.1. Dictionary

The dictionary consist of several files that save the keyword of the specific domain (as the domain in this work is data structure) and it also contain the synonym for these words if any, the field of each word, and the noise words. The dictionary is important during the module *Process Query* because all the steps such as filtering (removing noise word), replacing the words with its synonym, and creating the unknown word list all of these steps depends on the dictionary. The Dictionary consists of the following files:

- Finite number of keywords which is represented as internal database called (key_word.idb), and declared as:

  Database – Key_Word
  KW (Word)
  Word is string.

- A number of words and its synonym which is represented as external data base called (synonym.edb) by building a B+ tree for it where the key of the B+ tree is the word to find its synonym, and declared as:

  Synonym (Word, Word_Synonym),
  Word & Word Synonym are String

- A number of noise word which will be filtered during the process of the query which is represented as external database called (noise.edb) by building a B+ tree for it where the key of the B+ tree is the noise word itself, and declared as:

  Noise (Word)
  Word is String

- An external database called (general.edb) which contain a word and the word belong to which field e.g. ("stack", "data structure"), ("integer", "data type"), by building a B+ tree for it and it's key is the word, and declared as:

  General (Word, Field) ,
  Word & Field are String

3.5.2. Question Database

An internal database called "question.idb" which contains the questions asked by the user and it is not answered yet by the expert, and contains also a question that is already answered by the expert but during the performance module it is marked as incomplete answer and asks about more details. When the expert answers any question this question will be deleted from the file, and declared as:

```
Database – Question
        Question (Oq, Uwl, Fl)  /* Oq String, Uwl, Fl List of String */
        Question2 (Oq)             /* Oq String */
```

Where:

Oq is the original question entered by the end user before any processing using *Process Query* module. The original question is saved to display it for the expert.

Uwl is the Unknown word list.
Fl is the filtered list.

3.5.3. Questions and Answers Database

An external database called "ques_ans.edb" which contain the questions asked by the end user after processing it using sub module *Process Query* and the answer for that question e.g. If the question asked by the end user "What is Stack?", it will be saved in the file as Q_A (["what", "stack"], "An ordered list in which all insertions and deletions are made at one end"), and declared as:

```
    Q_A (Oq, Fl, Answer)   /* Fl List of string,
                             Answer & Oq are String */
```

Where

Fl: is the filtered list

Answer: is the answer for the question.

Oq: is the original question entered by the end user before any processing using *Process Query* module.

3.5.4. Other Databases

There are many other databases such as the file (fields.idb) which we save in it the fields of the words i.e. data structure, data type, noise word and if the word doesn’t belong to any of these fields then another field will be added to these fields, and is declared as:

Database – Fields

Fields (ListField)

ListField is list of String

The second file is (neg_q_f.idb) neglected questions file, which we save in it the questions that is asked by the end user and these questions is not reasonable or if the question is not of the expert concern

Database - Neg_ques_db

Neg_q (Q, Fl, Y, M, Y)

Where

Q is string represent the neglected question

Fl is list of string the filtered question

Y is integer represent the year

M is integer represent the month

D is integer represent the day

The last three fields, year, month and day represent the date when the end user enter its question. The date is saved so we can delete this unreasonable questions after a specific period e.g. a week. These questions will be deleted because it is not related to the subject so; it will be deleted to reduce the size of the knowledge base.

3.6. Performance Module

Make sure that the knowledge produced is useful. When the end user get the answer from the knowledge base after learning , a message will appear to him/her If the answer is "Yes" then the knowledge produced is useful, else if he/ she choose "No" then the same question will be saved again in the file (Question.idb) and marked as incomplete answer so that when the expert begin to learn the system this question will appear to the expert again with the old answer for that question and ask the expert more details for the question, The new answer will be added to the old one and saved in the file again replacing the old information. Algorithm 3.4 represents the Performance Module is:

Algorithm 3.4 Performance Module

```
Input: Input from End User

Output: Updating existing knowledge base

Step 1: Display if the answer is Ok with user
            If Answer = “yes” then do nothing
             Else
                    add the question to "question.idb" file and mark it as
                    incomplete   answer.

Step 2: Stop.
```

3.7 Implementation

This learning system is successfully implemented using Visual Prolog Programming Language. Prolog is an efficient declarative programming Language. This means that given the necessary facts and rules prolog will use deductive reasoning to solve your programming problems. It is provided with a powerful built in searching technique and database programming features that reduce the program size and save the design efforts of searching for the appropriate sorting and searching algorithm.

3.7.1 User Interface

To Show how the system works an example is considered to be run to show the windows that appear to the user and how to make use of these windows. The main window is shown in fig 3.2. From this main menu the user can choose either **End User** or **Expert**.

Example: Define the stack please.

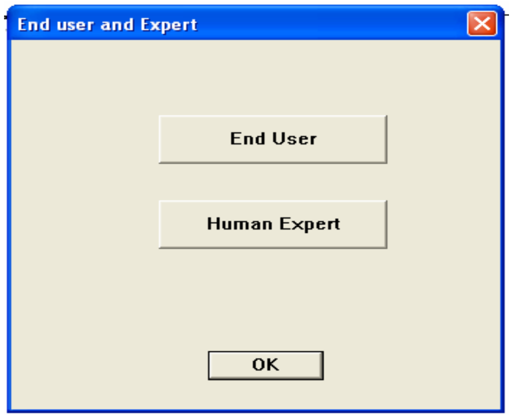
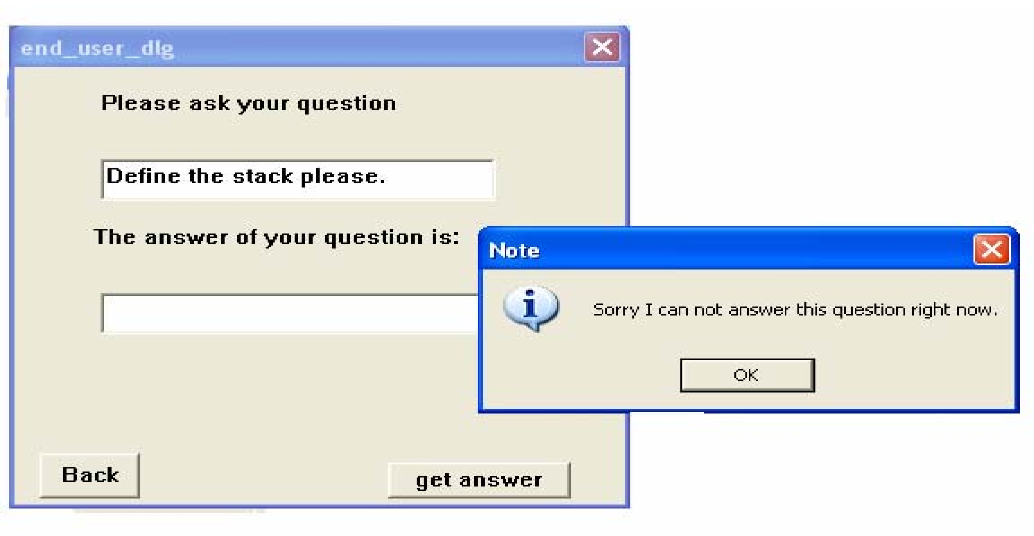


Fig 3.2 Main Menu

3.7.1.1 End User

If the end user wants to ask a question he/she will choose **End User,** then it will appear to the end user the window. The end user writes its question in the question field if the answer is not available message will appear to the end user "Sorry we can not answer this question right now." Such as shown in fig 3.3:



Please ask your question

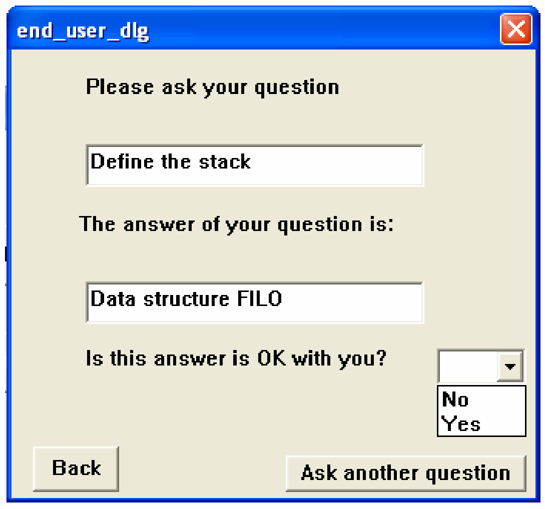
Define the stack please.

The answer of your question is:

Sorry I can not answer this question right now.

Fig 3.3 End User, answer is not available

If the answer is available then the answer will displayed and the system will ask the user if this answer is OK with him such as shown in fig 3.4



Please ask your question

Define the stack

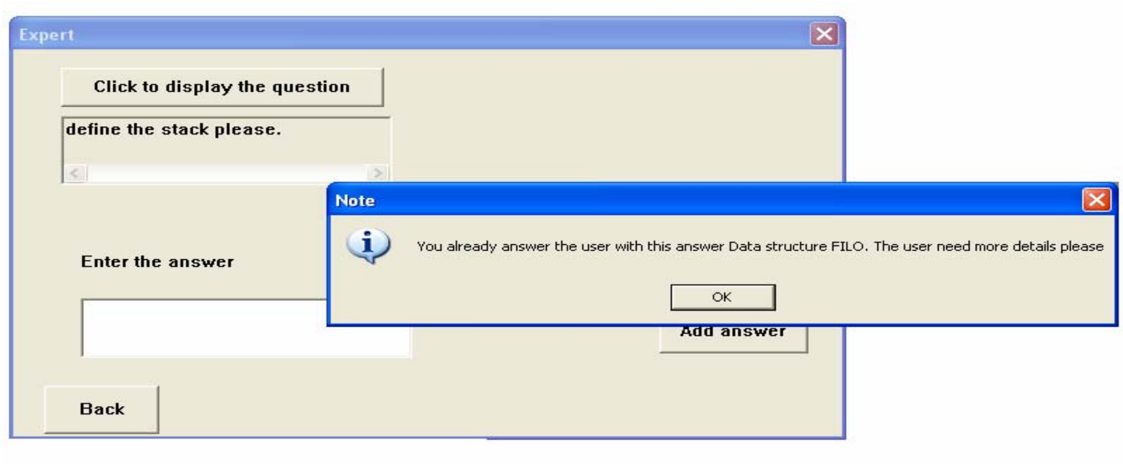
The answer of your question is:

Data structure FILO

Is this answer is OK with you?

Fig 3.4 End User, answer available

If the end user chooses **Yes** then the answer is saves in the knowledge base as complete answer. If end user chooses **No** then the answer is saves in the knowledge base as incomplete answer and when the expert enter to check if there is not answered question a message will appear to the expert "You already answer the user with this answer the user want more details such as shown in fig 3.5:



define the stack please.

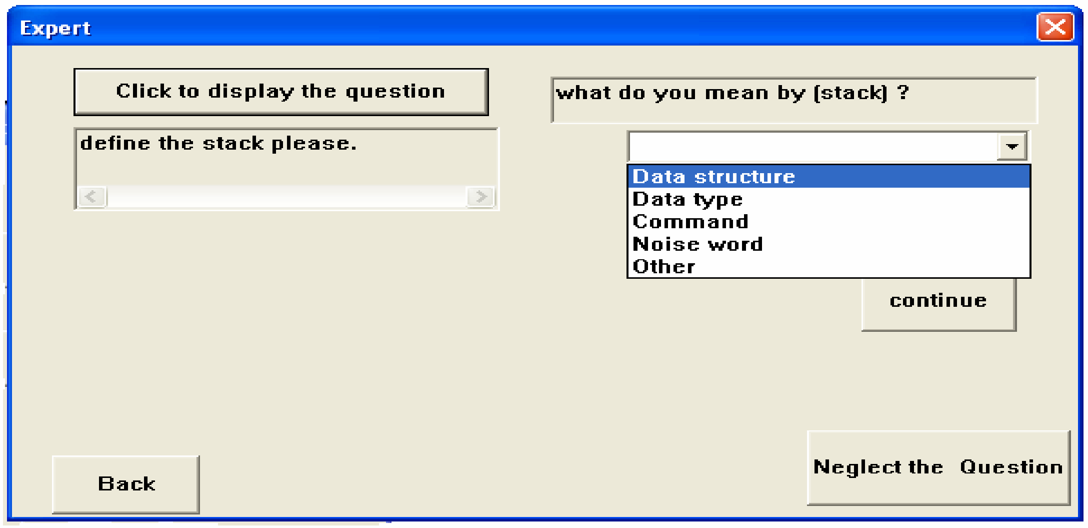
You already answer the user with this answer Data structure FILO. The user need more details please

Fig 3.5 Expert, the user wants more details

Then the expert will explain more about the question and this new answer will be saved in addition to the old one.

3.7.1.2 Expert

When the user chooses **Expert** from the main menu in fig 3.2 the system will ask the user to enter a password to check if user is authorized or not, if the password is correct. The expert click on "*Click to display the question*" bottom, the following window will appear to the end expert as shown in fig 3.6:



define the stack please.

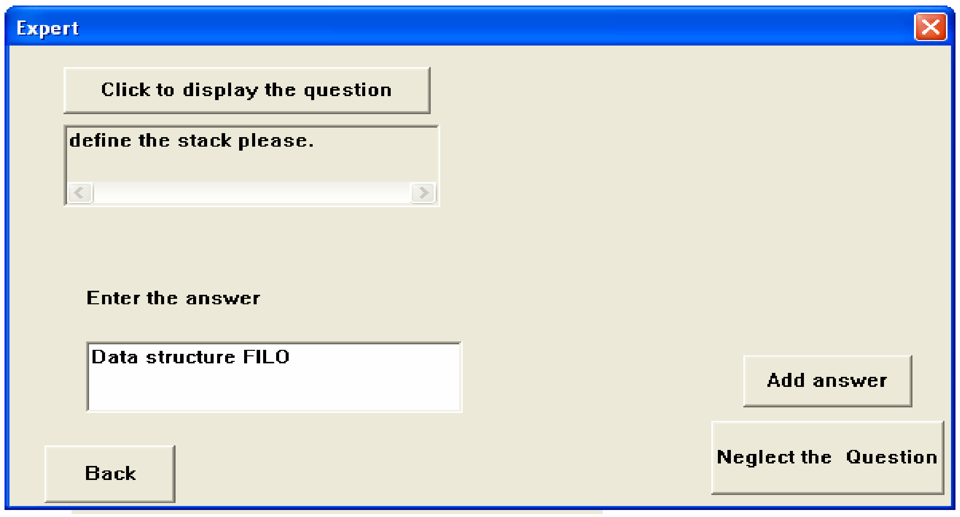
what do you mean by (stack) ?

Data structure
Data type
Command
Noise word
Other

Fig 3.6 Expert, first phase of learning

The first phase of learning will be implemented by learning each unknown word "*stack*" in our example we choose "*data structure*" , and then continue in learning any other unknown word "*define*" we will choose "*Command*".

Then the second phase of learning begins by learning the answer for the question and then click "*Add answer*" such as shown in fig 3.7:



define the stack please.

Data structure FILO

Fig 3.7 Expert, second phase of learning

If the end user enters unreasonable question e.g. “Where is Baghdad?” The expert has the right to neglect the question by clicking on “*Neglect the question*” bottom as shown in fig 3.8:

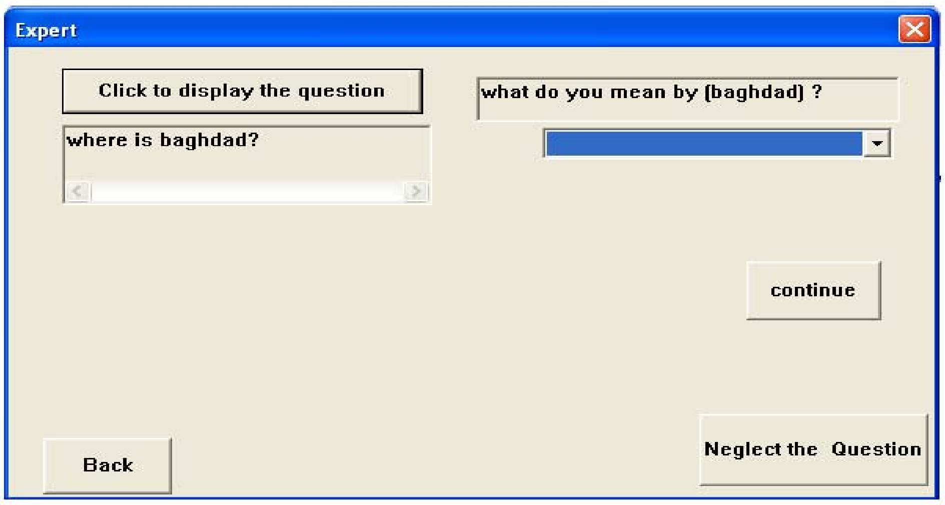

Expert
Click to display the question
where is baghdad?
what do you mean by (baghdad) ?
continue
Back
Neglect the Question

Fig 3.8 Expert, Neglect the question

3.7.2 Case Study

We will take some examples to illustrate how the system is implemented

<u>Example 1:</u>

<u>Q:</u> What is Stack?

During the Process Query module all capital letters will be changed to small letter to be case insensitive, convert the sentence into list of string L, filter any noise word, add any unknown word to the unknown word list Uwl, and save the question and the unknown word list in question.idb database. As shown in fig. 3.9 -a-.

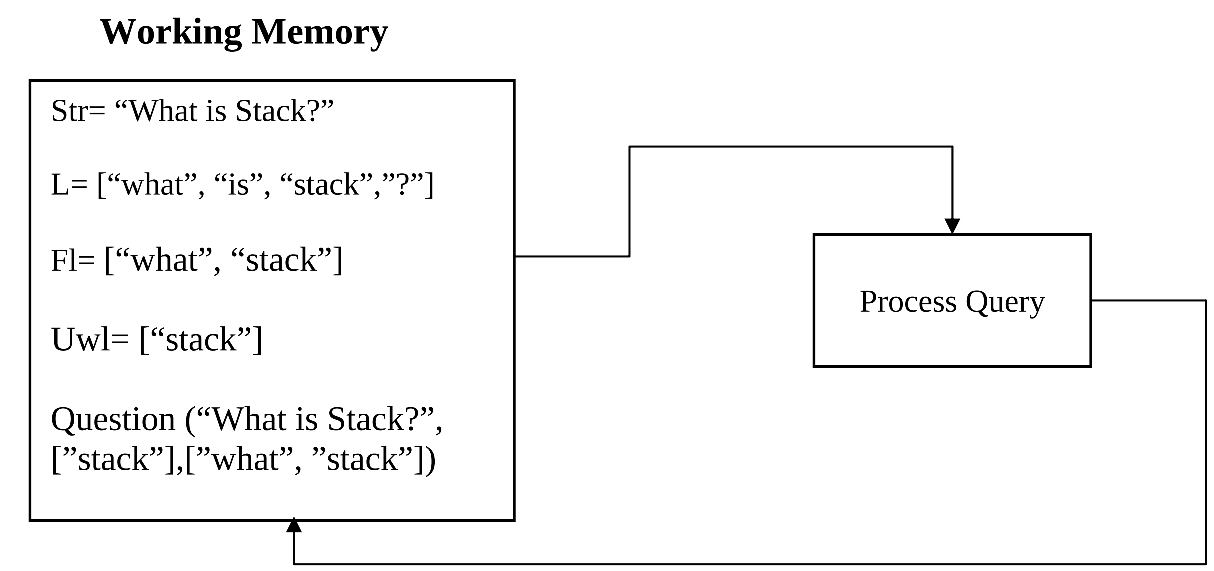

Working Memory
Str= “What is Stack?”
L= [“what”, “is”, “stack”,”?”]
Fl= [“what”, “stack”]
Uwl= [“stack”]
Question (“What is Stack?”, [”stack”],[”what”, ”stack”])
Process Query

Fig 3.9 Learning Process (Example 1) -a-

During Learning Engine the unknown words in the unknown word list will be determine to which field it belongs, by saving it in the database, the answer for the question entered by expert and this answer will be saved in the ques_ans.edb database, and the question will be deleted from question.idb. As shown in fig. 3.9 -b-

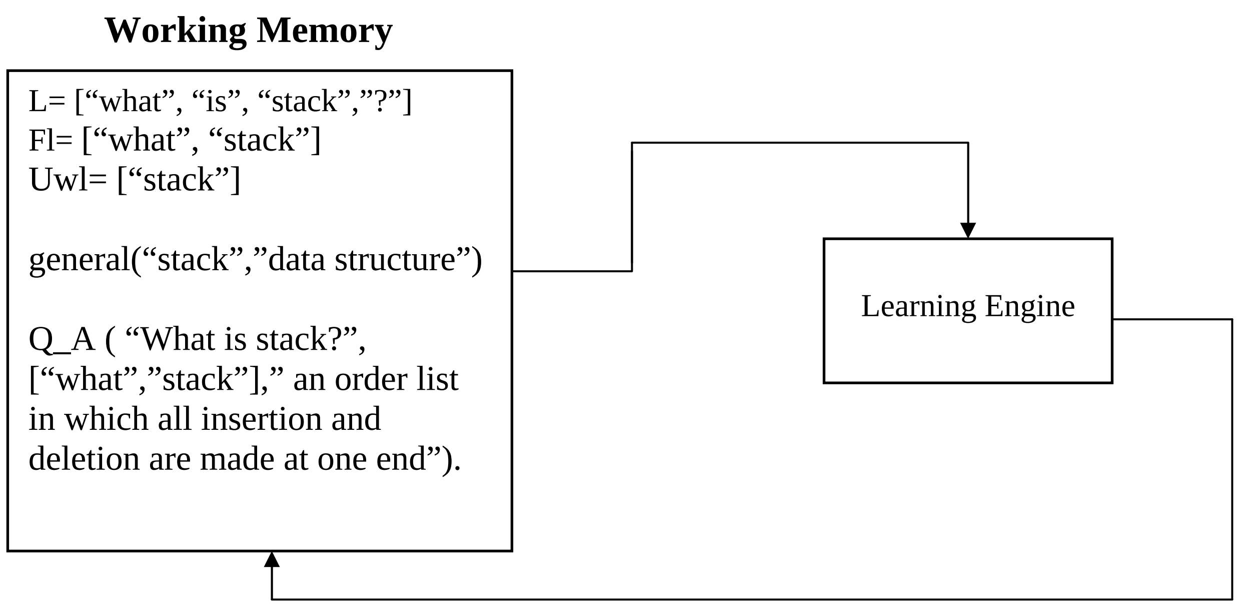

Working Memory
L= [“what”, “is”, “stack”,”?”]
Fl= [“what”, “stack”]
Uwl= [“stack”]
general(“stack”,”data structure”)
Q_A (“What is stack?”, [“what”,”stack”],” an order list in which all insertion and deletion are made at one end”).
Learning Engine

Fig 3.9 Learning Process (Example 1) -b-

During the Performance Engine a message will be displayed for the end user if the answer is OK or not if not then the question will be save in question.idb database and mark it as incomplete answer, as shown in fig 3.9 -c-

Working Memory

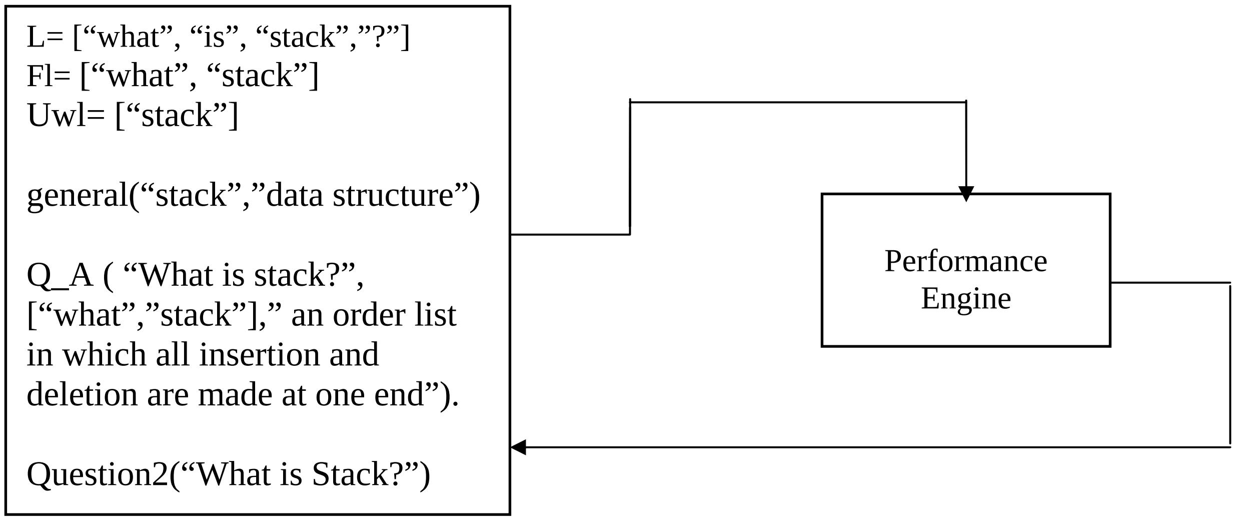

L= [“what”, “is”, “stack”,”?”]
Fl= [“what”, “stack”]
Uwl= [“stack”]
general(“stack”,”data structure”)
Q_A (“What is stack?”, [“what”,”stack”],” an order list in which all insertion and deletion are made at one end”).
Question2(“What is Stack?”)
Performance Engine

Fig 3.9 Learning Process (Example 1) -c-

<u>Example 2:</u>

<u>Q:</u> Define the Rear please.

During the Process Query module all capital letters will be changed to small letter to be case insensitive, convert the sentence into list of string L, filter any noise word, add any unknown word to the unknown word list Uwl, and save the question and the unknown word list in question.idb database. As shown in fig. 3.10 -a-.

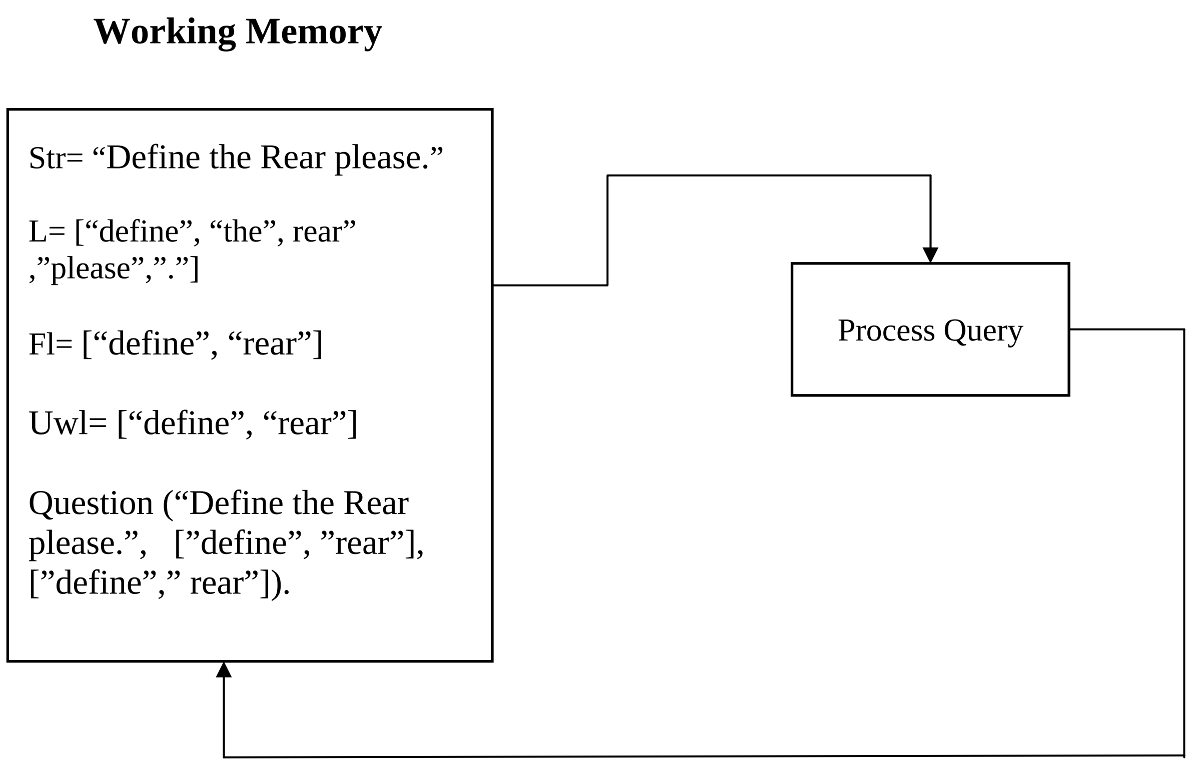

Working Memory
Str= “Define the Rear please.”
L= [“define”, “the”, rear” ,”please”,”.”]
Fl= [“define”, “rear”]
Uwl= [“define”, “rear”]
Question (“Define the Rear please.”, [”define”, ”rear”], [”define”,” rear”]).
Process Query

Fig 3.10 Learning Process (Example 2) -a-

During Learning Engine the unknown words in the unknown word list will be determine to which field it belongs by saving it in the database, in this example the unknown word “Rear” is not belong to any of the mentioned fields so, a new field will be added , the answer for the question entered by expert and this answer will be saved in the ques_ans.edb database, and the question will be deleted from question.idb. As shown in fig. 3.10 -b-

Working Memory

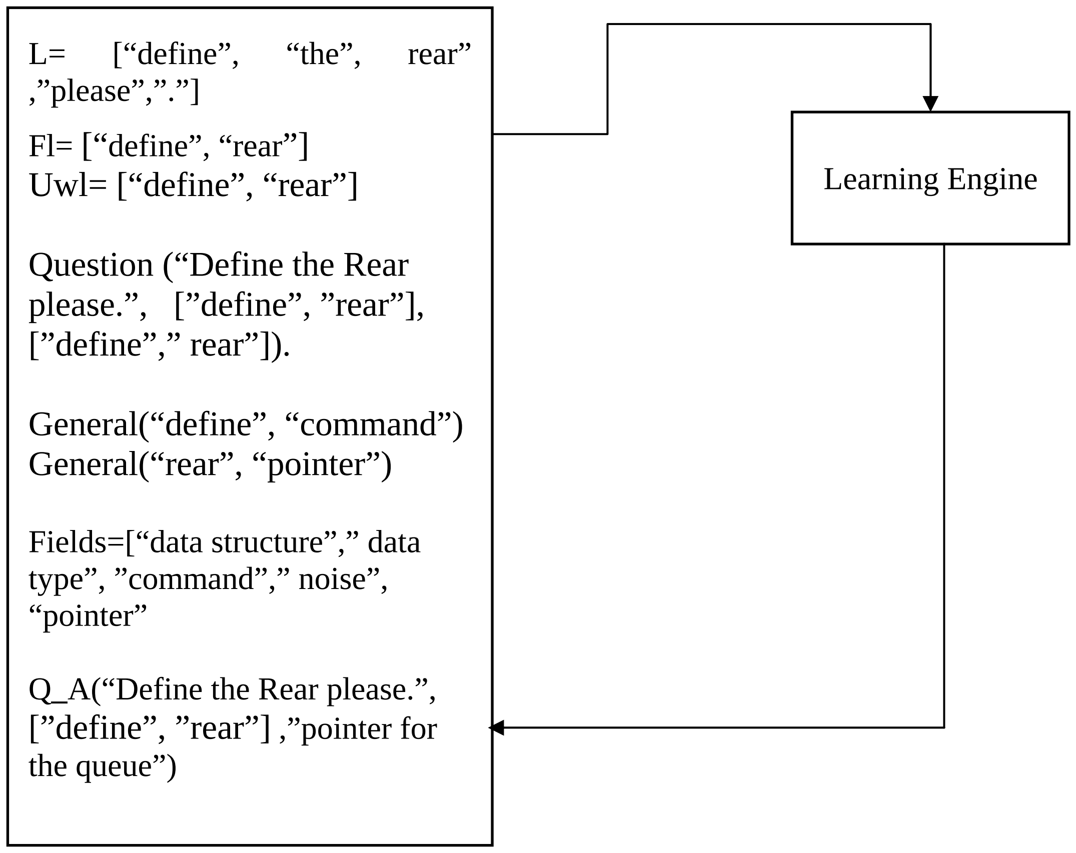

L= [“define”, “the”, rear” ,”please”,”.”]
Fl= [“define”, “rear”]
Uwl= [“define”, “rear”]
Question (“Define the Rear please.”, [”define”, ”rear”], [”define”,” rear”]).
General(“define”, “command”)
General(“rear”, “pointer”)
Fields=[“data structure”,” data type”, ”command”,” noise”, “pointer”
Q_A(“Define the Rear please.”, [”define”, ”rear”] ,”pointer for the queue”)
Learning Engine

Fig 3.10 Learning Process (Example 2) -b-

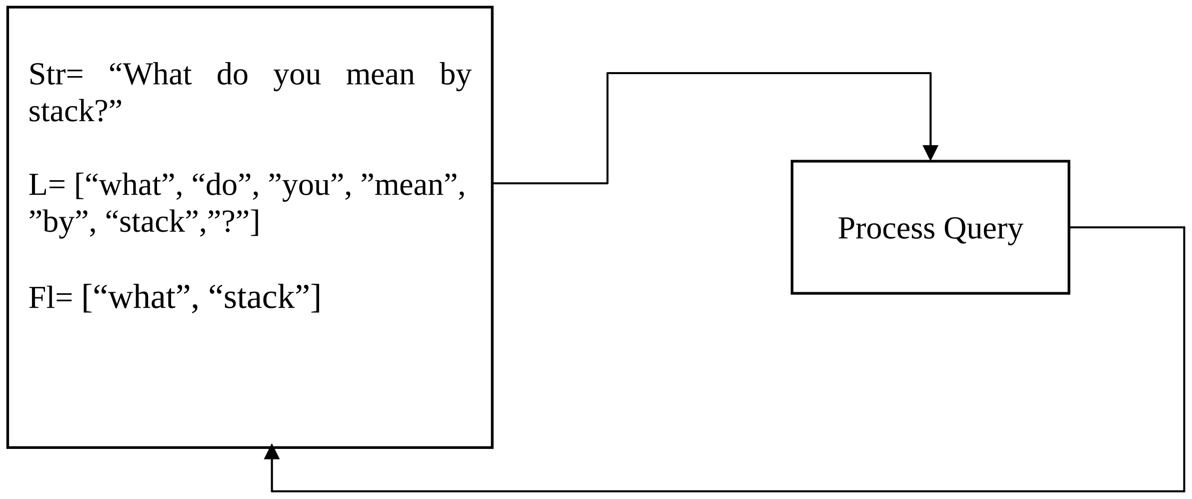
During the performance engine a message will be displayed for the end user if the answer is OK or not if the answer is Ok for the user the answer is marked as complete answer and no changes in the working memory will happen.

<u>Example 3:</u>

<u>Q:</u> What do you mean by stack?

In this example the system will immediately answer the end user because it is already learn it previously in example 1; the learning engine model will not run in this example.

Working Memory



Str= "What do you mean by stack?"
L= ["what", "do", "you", "mean", "by", "stack","?"]
Fl= ["what", "stack"]
Process Query

Fig 3.11 Learning Process (Example 3)

Example 4:

Q: Where is Baghdad?

During the process query the question is formulated as illustrated in the previous examples can be shown in fig. 3.12

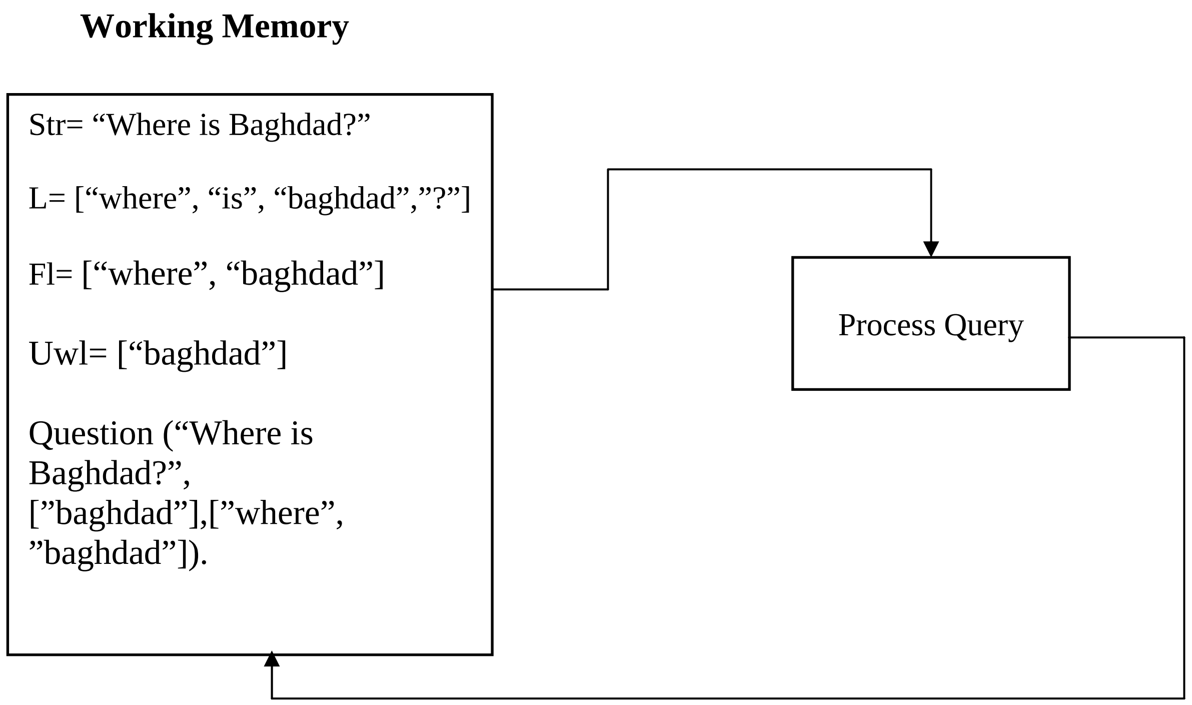

Working Memory
Str= “Where is Baghdad?”
L= [“where”, “is”, “baghdad”,”?”]
Fl= [“where”, “baghdad”]
Uwl= [“baghdad”]
Question (“Where is Baghdad?”, [”baghdad”],[”where”, ”baghdad”]).
Process Query

Fig 3.12 Learning Process (Example 4) -a-

In this example the question is not of the expert concern so the expert will neglect the question immediately and the question will be saved in the neglected questions file neg_q_f.idb with the date the end user ask the question, the date is saved so that the this neglected questions will be deleted after a predefined period e.g. seven days. As shown in fig. 3.12 -b-

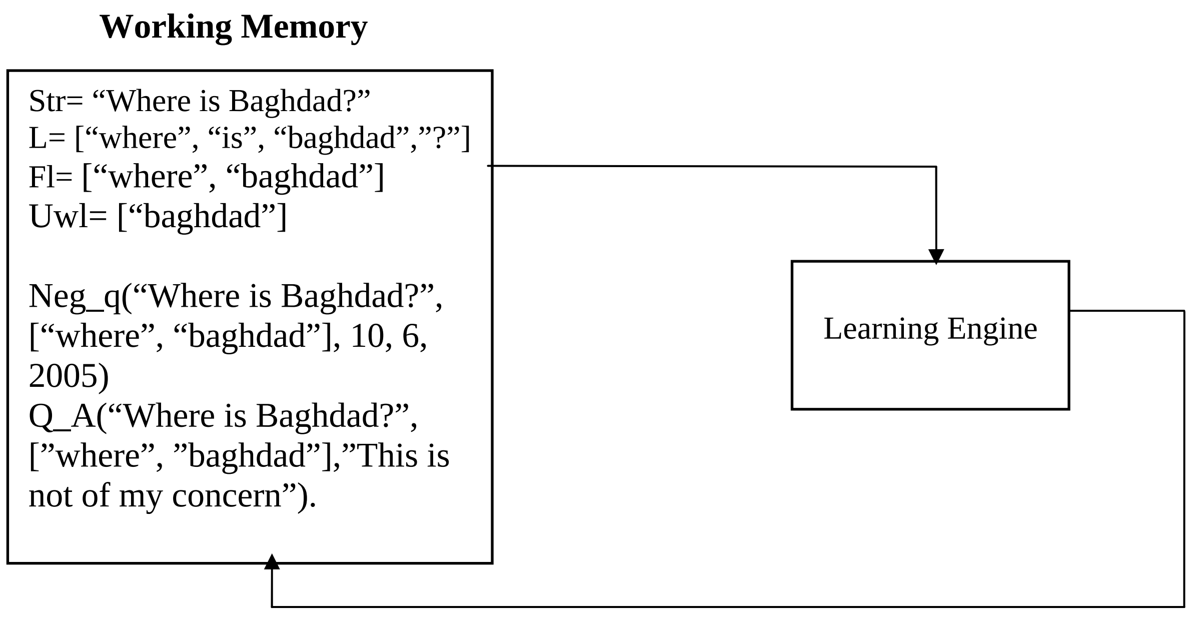

Working Memory
Str= “Where is Baghdad?”
L= [“where”, “is”, “baghdad”,”?”]
Fl= [“where”, “baghdad”]
Uwl= [“baghdad”]
Neg_q(“Where is Baghdad?”, [“where”, “baghdad”], 10, 6, 2005)
Q_A(“Where is Baghdad?”, [”where”, ”baghdad”],”This is not of my concern”).
Learning Engine

Fig 3.12 Learning Process (Example 4) -b-

Chapter Four

Discussion and Conclusions

4.1 Introduction

In this chapter the development environment, the conclusions of this work are given along with suggested recommendations for future work.

4.2. Discussion

This system is an attempt to develop an adaptive learning system with natural language understanding. Its features and characteristics can be stated as follows:

- Since the learning system used fast search B+ tree indexing method which is quickly look up information stored in external databases, so one can conclude that using this system to construct large knowledge bases will not cause any space or time problem in constructing and updating knowledge point of views.
- The system uses a combination between two learning strategies Rote learning since it acquires new facts or skills by transforming and augmenting existing knowledge that bears strong similarity to the desired new concept or skill into a form effectively useful in the new situation, and the second learning strategy is Learning by Instruction since it acquires knowledge from an expert that the learning transform the knowledge from the input language to an internally usable representation, and that the new information be integrated with prior knowledge for effective use.

- The system uses two types of database, internal and external databases. Prolog internal database provide fast data retrieving since the principle of searching techniques is the same of relational database. While the advantage of using Prolog external database becomes evident when there is a need to write systems that use large amounts of data. Moving the data to external files reduces the storage space which is valuable processing memory.
- The visual interface design is simplified due to using visual tools in designing menus and help ability. Visual interface make use of external memory size, which provides the property of operating on large amounts of data.
- The system provides a security system (depends on passwords) that prevents the end user from altering human expert knowledge.

4.3. Conclusions

- By making a few changes the program can run on different domains, i.e. by replacing the initial knowledge base, the dictionary and the keywords.
- Any type of inference will provide some useful knowledge that is worth remembering for future use.
- The Inferential Learning Theory depicts learning as a goal-guided process of deriving the desired knowledge by using input and background knowledge and knowledge transmutations as operators.
- The system is easy to use, because of its friendly interface (which draws knowledge base that enables knowledge monitoring during construction process).

4.4. Suggestions for Future Work

- Implement the learning system using another symbolic learning methods e.g. learning by induction or deduction.
- Develop a system in a new form that work in different domains.
- Implement it using Arabic natural language.

References

1. Kazakov, Dimitar, Ph.D. Thesis "Natural Language Processing Application of machine Learning", Czech Technical University, Prague, 1999.

2. Michalski, R.S., "Principles of Machine Learning and Inference", 2002.

3. Michalski, R. S., and Ram, A., - "Learning as Goal-Driven Inference" in Goal Driven Learning, MIT Press/Bradford Books, Cambridge MA, (1995). http://www.mli.gmu.edu/papers/rsMaR.chapter.ps

4. Eric Brill, Ph.D. Thesis, A Corpus Based Approach to Language Learning, Department of Computer and Information Science, University of Pennsylvania 1993.

5. Sunglian William. Kung, M.Sc. Thesis, "Exploiting Equity Momentum with Symbolic Machine Learning", University of Oxford, 1998.

6. Cynthia Ann Thompson, "Semantic Lexicon Acquisition for Learning Natural Language Interfaces", University of Texas, 1998

7. Fakhri Sawsan, M.Sc. Thesis, Machine Learning a Multistrategy Approach, National Computer Center, 2001.

8. Cardie, Claire, and Mooney, Raymond J., "Machine Learning Special issue on Natural Language Learning", 1999.
http://www.cs.utexas.edu/users/ml/mlj-nll/

9. Rich, Elaine & Knight, Kevin, Machine Learning, 1994
http://www.cse.dmu.ac.uk/~jennyc/machin_L_full.htm

10. Michalski, R., "Concept Learning", Encyclopedia of Artificial Intelligence, 1990.

11. Michalski S. Rysard, Carbonell J.G.,Mitchell Tom M., “Machine Learning an Artificial Intelligence Approach”, Kaufman Publishers Inc.,Los Altos, CA, 1983

12. Nilsson, Nils J., “Introduction to Machine Learning”,1996.

13. Mitchell, Tom, “Machine Learning”, McGraw-Hill, 1997.

14. Gheorghe, Tecuci, ”machine learning: Rote learning” Learning Agents Laboratory, Computer Science Department, George Mason University, 1998.

15. Rivest, Ron and Singh, Mona, Machine Learning Lecture, 1994.

16. H. Chen, P. Buntin, L. She, S. Sutjahjo, C. Sommer, D. Neely, “Expert Prediction, Symbolic Learning, and Neural Network: An experiment on Greyhound Racing”, 1992.

17. Mooney Raymond J., “Symbolic Learning for Natural-Language Processing: Integrating Information Extraction and Querying”, 2001.

18. Lugar G. E., Stubblefield William A., Artificial Intelligence, 1998.

19. Charniack, E., and McDermott, D., "Introduction to Artificial Intelligence", 1985.

20. Darlington Keith, “The essence of Expert Systems”, Pearson Education, England, 2000.

21. D. Schuurmans, “Machine Learning course notes”, University of Waterloo, 1999.

22. Ebatali “Natural Language Processing“, http://cogsci.ucsd.edu/%7Ebatali/108b/lectures/natlang.txt, 1999

23. E. Rich and K. Knight ,Paul R.Cohen and Edward A Feigenbaum, “Review Machine Learning”, 1994.

http://www.scism.sbu.ac.uk/inmandw/review/ml/review/rev8632.html

24. Gazdar Gerald, Mellish Chris, "Natural Language Processing In Prolog", 1989.

25. Roth Dan," Machine Learning and Natural Language", 2000.

26. Horowitz Ellis, Sahin Startaj, "Fundamental of Data Structure in Pascal", 1984.

27. Muaid Esraa, M.Sc. Thesis, Database Communication using Arabic Natural Language, Department of Computer Science, Al-Nahrain University,2001

خلاصة

إن هذه الدراسة المقدمة ليست سوى محاولة لبناء منظومة تعليمية لفهم اللغة الطبيعية و الطريقة المستخدمة هي طريقة التعليم الرمزي و مجال الدراسة المختار هو تعلم موضوع هياكل البيانات.

إن تقنية التعليم المستخدمة هي طريقة التعلم الرمزي و ذلك بجمع إستراتيجيتين من التعليم و هما طريقة التعليم بتكرار الحفظ و التعليم بواسطة توجيه التعليمات. حيث إن طرق التعليم الرمزي قد طورت لإنشاء نظم تتطلب استخلاص المعلومات من مستندات اللغة الطبيعية و للإجابة عن استفسارات معينة من قاعدة البيانات.

النظام المقترح يتكون من عدة مركبات, مركب معالجة الاستفسار (مصدر المعلومات) متمثلا بالاستفسار المقدم من قبل المستخدم النهائي, مركب التعلّم ينجز عملية التعلّم و ينتج معرفة جديدة, مركب محرّك الأداء يتأكد من فائدة المعرفة الجديدة المكتسبة من مركب التعلّم.

و نفذ هذا النظام باستخدام لغة برولوك الصورية Visual Prolog 5.1 لبناء المنظومة و واجهاتها حيث إن هذه اللغة توفر بيئة التطوير البصرية.

جمهورية العراق
وزارة التعليم العالي
جامعة النهرين
كلية العلوم

نظام تعليمي رمزي لفهم اللغة الطبيعية

رسالة مقدمة إلى كلية العلوم جامعة النهرين كجزء من متطلبات نيل شهادة الماجستير في علوم الحاسبات

من قبل

زينة عبد الرزاق موسى الجصاني
بكالوريوس
2002

المشرفون

د.مؤيد عبد الرزاق فاضل **د.طه سعدون باشاغا**

2005 1426